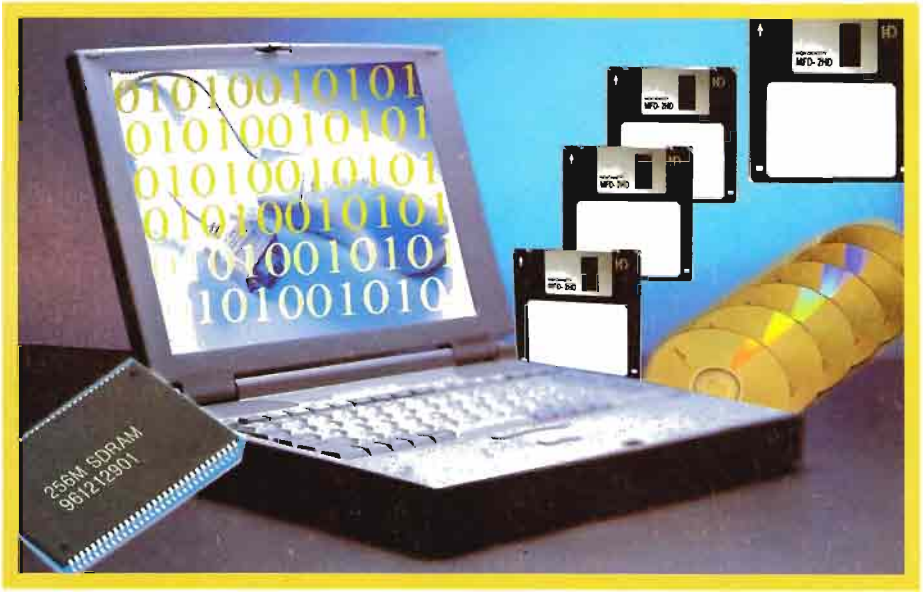
مبانی کامپیوتر و برنامه نویسی

جعفر تنها مهدی یوسف‌خانی



درسنامه

دانشگاه پیام نور

مبانی کامپیوتر و برنامه نویسی

(رشتهٔ مهندسی کامپیوتر)

جعفر تنها مهدی یوسف‌خانی

گروه کامپیوتر
(۱۱/د)

دانشگاه پیام‌نور
۱۱۷۰

سرشناسه	:	تنها، جعفر -
عنوان و پدید آور	:	مبانی کامپیوتر و برنامه نویسی (رشته کامپیوتر) / مؤلف: جعفر تنها، مهدی یوسف خانی .
مشخصات نشر	:	تهران: دانشگاه پیام‌نور، ۱۳۸۴.
مشخصات ظاهری	:	۴۱۵ص .
فروست	:	دانشگاه پیام نور؛ ۱۱۷۰. گروه کامپیوتر؛ ۱۱ / د
شابک	:	978 - 964 - 387 - 194 - 9
وضعیت فهرست نویسی	:	فیپا .
یادداشت	:	کتابنامه: ص . ۴۱۵ .
موضوع	:	۱. آموزش از راه دور. ایران .
موضوع	:	۲. کامپیوتر ها - آموزش برنامه ای . ۳. پاسکال (زبان برنامه نویسی کامپیوتر) – آموزش برنامه ای .
شناسه افزوده	:	الف. یوسف خانی، مهدی . ب. دانشگاه پیام نور. ج. عنوان .
رده بندی کنگره	:	QA۷۶/۷۳/پ۲ت۸۵۳
رده بندی دیویی	:	۰۰۵/۱۳۳
شمارهٔ کتابشناسی ملی	:	م۸۴-۲۱۶۰۰

دانشگاه پیام‌نور

مبانی کامپیوتر و برنامه نویسی

مؤلف: جعفر تنها- مهدی یوسف خانی

ویراستار علمی: دکتر احمد فراهی

تهیه و تولید: مدیریت تولید مواد و تجهیزات آموزشی

لیتوگرافی، چاپ و صحافی: انتشارات دانشگاه پیام‌نور

شمارگان: ۳۰۰۰۰ نسخه

نوبت و تاریخ چاپ: چاپ اول شهریور ۱۳۸۵، چاپ هشتم دی ۱۳۸۸

شابک: ۹ - ۱۹۴ - ۳۸۷ - ۹۶۴ - ۹۷۸

ISBN: 978 - 964 - 387 - 194 - 9

بسم الله الرحمن الرحیم

پیشگفتار ناشر

کتابهای دانشگاه پیام نور حسب مورد و با توجه به شرایط مختلف به صورت درسنامه، آزمایشی، قطعی، متون آزمایشگاهی، فرادرسی، و کمک‌درسی چاپ می‌شود. کتاب **درسنامه** (د) نخستین ثمرهٔ کوششهای علمی صاحب اثر است که براساس نیازهای درسی دانشجویان و سرفصلهای مصوب تهیه می‌شود و پس از داوری علمی در گروههای آموزشی، بدون طراحی آموزشی و ویرایش چاپ می‌شود. با تجدیدنظر صاحب اثر و دریافت بازخوردها و اصلاح نارساییها، درسنامه با طراحی آموزشی، ویرایش، و طراحی فنی ـ هنری به صورت **آزمایشی** (آ) چاپ می‌شود. با دریافت نظرهای اصلاحی، صاحب اثر در کتاب تجدید نظر می‌کند و کتاب به صورت **قطعی** (ق) چاپ می‌شود. در صورت ضرورت، در کتابهای چاپ قطعی نیز تجدید نظرهای اساسی به عمل می‌آید یا متناسب با پیشرفت علوم و فناوری بازنویسی می‌شوند. **متون آزمایشگاهی** (م) متونی است که دانشجویان با استفاده از آن و راهنمایی مربیان کارهای عملی آزمایشگاهی را انجام می‌دهند. کتابهای **فرادرسی** (ف) و **کمک‌درسی** (ک) به منظور غنی‌تر کردن منابع درسی دانشگاهی تهیه می‌شوند. کتابهای فرادرسی با تأیید معاونت پژوهشی و کتابهای کمک‌درسی با تأیید شورای انتشارات تهیه می‌شوند.

مدیریت تدوین

فهرست مطالب

عنوان صفحه

پیشگفتار

خداوند منان را شکر می‌گوییم که با اعطای نعمت حیات و عنایت او، توفیق آن را یافتیم تا کتابی را تحت عنوان مبانی کامپیوتر و برنامه‌نویسی به دانشجویان علاقمند تقدیم کنیم. با توجه به نیاز مبرم دانشجویان رشته مهندسی کامپیوتر برای یادگیری یک زبان برنامه‌نویسی پایه، بر آن شدیم، که تجربه چندین ساله خود در زمینه برنامه‌نویسی را در قالب کتابی در اختیار دانشجویان عزیز قرار دهیم. وجه تمایز این کتاب از سایر منابع، خودآموز بودن مطالب کتاب می‌باشد. برای یادگیری یک زبان برنامه‌نویسی پایه، ترجیح دادیم که زبان ساخت‌یافته پاسکال (Pascal) را در این کتاب مورد بحث و بررسی قرار دهیم. دلایل زیادی برای انتخاب این زبان برای یادگیری می‌توان بیان کرد، از آن جمله سادگی یادگیری، ساخت‌یافته بودن، کاربرد وسیع آن در نوشتن نرم‌افزارهای تجاری و غیره می‌باشد.

این کتاب شامل ۱۵ فصل است که بطور جامع و کامل در اختیار دانشجویان عزیز قرار داده شده است. فصل اول این کتاب شامل آشنایی با کامپیوتر و مباحثی مقدماتی در زمینه علم کامپیوتر می‌باشد که یکی از فصول مهم و بنیادی محسوب می‌شود و می‌تواند ایده و نمایی کلی از علم کامپیوتر در ذهن تداعی نماید. فصول دوم و سوم مباحثی در باب الگوریتمها، استفاده از آرایه‌ها در الگوریتمها و روش حل مسائل را به دانشجو می‌آموزد و یاد می‌دهد که چگونه الگوریتمها به صورت شماتیک رسم کنند.

از فصل چهارم تا نهم سعی کردیم که علائم، قواعد و دستورالعمل‌های موجود در یک برنامه به زبان پاسکال را به همراه انواع مثال‌ها توضیح دهیم. در این فصول دانشجو در

کل با ساختار برنامه در زبان پاسکال آشنا می‌شود. در پایان هر فصل تعدادی تمرین و پروژه برنامه‌نویسی قرار دادیم تا خواننده، بعد از اتمام مطالعه هر فصل بتواند خود را محک بزند.

از فصل دهم تا فصل چهاردهم داده‌های مرکب یا اصطلاحاً ساختمان داده‌ها در زبان پاسکال را به همراه استفاده از برنامه‌نویسی پیمانه‌ای، بررسی کردیم که توصیه می‌شود خوانندگان این فصول را با حوصله بیشتری مطالعه بفرمایند، تا درک صحیحی از برنامه‌نویسی ساخت‌یافته پیدا کنند.

فصل آخر کتاب را برای بررسی زمان اجرای الگوریتم‌های مختلف اختصاص دادیم، تا خواننده بتواند انواع الگوریتم‌ها، برای یک مسئله را مورد بحث و بررسی قرار دهد و تحلیل الگوریتم نماید.

در اینجا لازم است که از آقایان غلامرضا حلمی‌پسند و حمیدرضا شریفی که در ویرایش ادبی این اثر ما را یاری کردند، کمال تشکر را بنماییم.

سخن را با تشکر از همکاران عزیز، دانش‌پژوهان و دانشجویانی که با ارسال یادداشت، نظرات خود را به منظور بهتر و پربار کردن این کتاب برای مولفین ارسال می‌کنند تا در چاپ‌های دیگر مورد استفاده قرار گیرد، به پایان رسانده و از درگاه خداوند منان مسئلت داریم، این خدمت خالصانه را در راه رضای خود تلقی فرماید و ما را یاری نموده و نیرویی مرحمت فرماید تا بتوانیم در خدمت به دانش‌پژوهان به پاره‌ای از آنچه آرزو داریم، تحقق بخشیم.

مهدی یوسفخانی - جعفر تنها

فصل ۱

آشنایی با کامپیوتر

هدفهای کلی

- شناخت کامپیوترهای نسل قدیم و امروزی
- شناخت سخت‌افزارهای لازم برای کامپیوترهای شخصی
- بررسی نرم‌افزارها و انواع آن

هدفهای رفتاری

دانشجو پس از مطالعه این فصل باید بتواند:

- کامپیوترهای نسل جدید را با کامپیوترهای نسل قدیم مقایسه کند.
- سخت‌افزارهای لازم برای کامپیوترهای شخصی را بشناسد.
- انواع حافظه، مزایا و معایب آنها را شناخته و با هم مقایسه نماید.
- سیستم عامل و انواع آن را مقایسه نماید.
- نرم‌افزار و زبانهای برنامه‌نویسی را تعریف کند.

مقدمه

بشر از دیر باز دنبال ابزاری می‌گشت که بتواند محاسبات روزمره خود را با آن انجام دهد، بخصوص از وقتی که تجارت گسترده بین کشورها مطرح شد نیاز به ابزاری برای انجام محاسبات و ذخیره اطلاعات بیشتر احساس شد. این مسئله در مسائل دیگر نیز مطرح بود. (در جنبه‌های فنی، علمی و غیره) لذا بشر اولین بار از چرتکه برای انجام محاسبات خود استفاده کرد و بعدها ماشینی بنام کامپیوتر (Computer) پا به عرصه نهاد و توانست رضایت بشر را در انجام محاسبات و ذخیره اطلاعات جلب نماید. از سال ۱۹۴۰ تاکنون کامپیوترها به شکل عجیبی شیوه زندگی و کار ما را تغییر داده‌اند. امروزه تقریباً می‌توان گفت زندگی بدون استفاده از کامپیوتر امکانپذیر نیست. امروزه فیش‌های حقوقی، صورتحسابها، انواع گزارشات، انواع نمودارها، تهیه بلیط‌های هواپیماها و قطارها و غیره نیاز به کامپیوتر را بیش از پیش روشن‌تر می‌سازد. کامپیوترها همچنین در انجام اعمال بانکی، خریدهای روزمره، نوشتن کتابها و بسیاری از اعمال روزمره بشر کمک می‌کنند. گرچه در اذهان عمومی خلاف این موضوع جاری است، اما کامپیوتر نمی‌تواند مانند بشر استدلال کند، در واقع، کامپیوتر ماشینی است که محاسبات را با سرعت بالا و دقتی زیاد انجام می‌دهد.

برای انجام کارهای مفید با کامپیوتر باید برنامه‌ای برای آن نوشته شود. برنامه به مجموعه‌ای از دستورات و دستورالعمل‌ها گفته می‌شود که هدف خاصی را دنبال می‌کنند. برنامه‌ها معمولاً تحت یک زبان برنامه‌نویسی خاصی، نظیر پاسکال نوشته می‌شوند.

۱-۱- کامپیوترهای قدیمی و امروزه

در این بخش اشاره مختصری به تاریخچه کامپیوترها می کنیم. اولین کامپیوتر الکترونیکی در اواخر سال ۱۹۳۰ میلادی توسط دکتر جان آتاناسوف (Atanasoff) در دانشگاه ایالت آیوا (Iowa) طراحی شد. آتاناسوف کامپیوتر خود را برای کمک به محاسبات ریاضی طراحی کرد.

اولین کامپیوتر بزرگ (Super Computer) همه منظوره دیجیتال الکترونیک، تحت عنوان ENIAC در سال ۱۹۴۶ میلادی در دانشگاه پنسیلوانیا

ساخته شد. این کامپیوتر با سرمایه ارتش آمریکا طراحی شد. وزن این کامپیوتر ۳۰ تن و ابعاد آن ۵۰×۳۰ فوت بود. این کامپیوتر برای محاسبه جدول پرتابه‌ها، پیش‌گویی وضع آب و هوا و محاسبات انرژی اتمی بکار می‌رفت.

در کامپیوترهای اولیه از لامپهای خلاء بعنوان عنصر الکترونیکی پایه استفاده می‌کردند. در این ماشین‌ها ۱۹۰۰۰ لامپ خلاء استفاده شده بود و برای انرژی مصرفی لامپ‌ها و همچنین دستگاههای تهویه و خنک‌کننده ماشین حدود ۱۳۰ kw انرژی الکتریکی مصرف می‌شد. این ماشین‌ها دارای حجم زیادی بودند و سطحی را معادل ۹۰۱۵ مترمربع اشغال می‌کردند. این کامپیوترها به کامپیوترهای نسل اول معروف شدند. پیشرفت تکنولوژی در طراحی و ساخت اجزاء الکترونیکی باعث ایجاد نسل جدیدی از کامپیوتر بنام کامپیوترهای نسل دوم شد. که به میزان قابل توجهی کوچکتر و ارزان‌تر از نسل قبلی بودند. در این نسل از کامپیوترها ترانزیستور به بازار ارائه شده و آنها را در کامپیوترهای این دوره بکار بردند. همچنین از حلقه‌های کوچک مغناطیسی (Magnetic Core) بعنوان حافظه در این ماشین‌ها استفاده شد.

بعد از کامپیوترهای نسل دوم با پیشرفت الکترونیک و دیجیتال، کامپیوترهای جدید و عمدتاً با مزایایی از قبیل حجم کوچکتر، سرعت پردازش بالا، حجم ذخیره اطلاعات بیشتر و ارزان قیمت به بازار ارائه شد.

کامپیوترهای امروزی با بکارگیری ریزپردازنده (Microproccessor) به کامپیوترهای نسل چهارم معروفند. البته نسل‌های جدید دیگر کامپیوترها نیز به بازار ارائه می‌شود. (ماشین‌های هوشمند و رباتها)

در کامپیوترهای امروزی سرعت پردازش بسیار بالا، حجم اجزاء سخت‌افزاری بسیار کوچک، حجم حافظه بالا و غیره آنها را از نسل های دیگر متمایز می‌سازد.

در حالت کلی کامپیوتر از دو جزء اصلی سخت‌افزار (Hardware) و نرم‌افزار (Software) تشکیل می‌شود. منظور از سخت‌افزار، بخش فیزیکی و اجزاء الکترونیکی کامپیوتر می‌باشد. کاربر (User) برای استفاده از کامپیوتر نیاز به یک رابط به نام نرم‌افزار دارد، لذا نرم‌افزار رابط بین کاربر و سخت‌افزار

می‌باشد و بدون آن نمی‌توان از کامپیوتر استفاده کرد.

۲-۱- سخت‌افزار کامپیوتر

در حالت کلی روال موجود در کامپیوتر را بصورت زیر می‌توان ترسیم کرد:

شکل ۱-۱ سیستم کامپیوتری

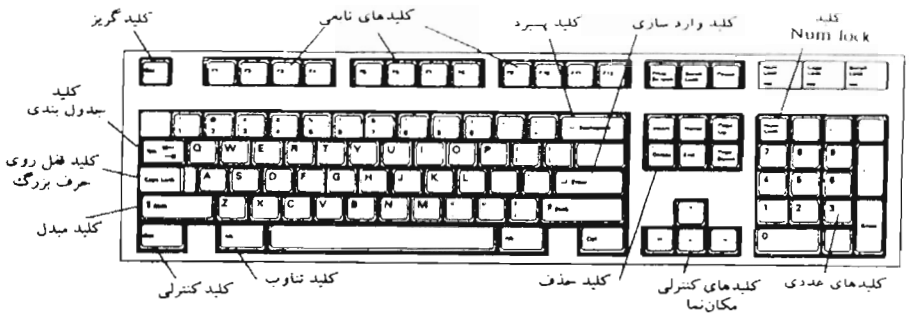
در شکل ۱-۱، همانطور که ملاحظه می‌کنید کامپیوتر داده‌ها را دریافت کرده سپس آنها را پردازش نموده و در نهایت خروجی لازم را تولید می‌نماید.

برای انجام هر سه عمل فوق یعنی ورودی، پردازش و خروجی قطعات سخت‌افزاری مورد نیاز است. در این بخش توضیح مختصری در مورد قطعات ارائه می‌دهیم.

کامپیوترهای امروزی معمولاً از قطعات زیر تشکیل می‌شوند:

- دستگاههای ورودی
- حافظه‌های جانبی
- حافظه‌های اصلی
- واحد پردازشگر مرکزی
- دستگاههای خروجی
- دستگاههای ورودی

بوسیله دستگاههای ورودی داده‌ها وارد کامپیوتر می‌شوند از دستگاههای ورودی می‌توان به‌صفحه کلید(keyboard)، موس(mouse)، قلم نوری(light pen) و غیره اشاره کرد. صفحه کلید و موس از مشهورترین نوع دستگاههای ورودی هستند. به شکل ۲-۱ توجه کنید.



شکل ۲-۱ صفحه کلید

- حافظه

حافظه یکی از اجزاء اصلی هر کامپیوتر می‌باشد که برای ذخیره داده‌ها بکار می‌رود. حافظه یک کامپیوتر از محل‌های پشت سر هم بنام سلول‌های حافظه (memory cells) تشکیل شده است. برای ذخیره و بازیابی اطلاعات، کامپیوتر باید آدرس هریک از سلولهای حافظه را بشناسد. بنابراین هر کدام از سلول‌های حافظه دارای آدرس منحصر بفردی می‌باشد که محل آن را در حافظه مشخص می‌کند. اغلب کامپیوترها دارای میلیون‌ها سلول حافظه هستند، که هر یک آدرس خاص به خود را دارند و داده‌ها در سلول‌ها ذخیره می‌شوند.

هر سلول حافظه شامل گروهی از واحدهای کوچکتر بنام بایت (Byte) می‌باشد و هر بایت از ۸ واحد کوچکتر بنام بیت (Bit) تشکیل می‌شود. یک بایت میزان حافظه‌ای است که برای ذخیره کردن یک کاراکتر موردنیاز است. واحدهای دیگر حافظه بصورت زیر می‌باشند:

2^{10} Byte = 1 kB , 2^{10} kB = 1 MB , 2^{10} MB = 1 GB، 8 bit = 1 Byte

کلیه داده‌ها با هر حجم در حافظه با الگوهای خاصی از صفر و یک ذخیره می‌شوند. برای ذخیره کردن یک مقدار، کامپیوتر به هر بیت از سلول انتخابی حافظه، مقدار 0 یا 1 می‌دهد و محتویات قبلی سلول را در حین پردازش از بین می‌برد و مقدار جدید را جایگزین آن می‌نماید.

- حافظه‌های اصلی

در حالت کلی دو نوع حافظه وجود دارد، حافظه‌های اصلی و حافظه‌های جانبی. داده‌ها، برنامه‌ها و نتایج در حافظه اصلی (بطور موقت) ذخیره می‌شوند. اغلب کامپیوترها دو نوع حافظه اصلی دارند: حافظه‌های با دسترسی تصادفی (RAM)[1] که داده‌ها و برنامه‌ها را بصورت موقت ذخیره می‌کند و حافظه فقط خواندنی (ROM)[2] که داده‌ها و اطلاعات را بصورت دائم ذخیره می‌کند. وقتی کامپیوتر روشن می‌شود برنامه‌های لازم روی RAM قرار می‌گیرد یا اصطلاحاً لود (Load) می‌شود. این حافظه فرار است و به محض خاموش شدن اطلاعات آن پاک می‌شود. از طرف دیگر حافظه ROM اطلاعات را بصورت پایدار در خود ذخیره می‌کند. اطلاعات این حافظه فقط خواندنی است. چون حافظه فرار نیست لذا با خاموش شدن کامپیوتر اطلاعات از آن پاک نمی‌شود. در ROM دستورالعمل‌هایی ذخیره می‌شوند که به محض روشن شدن کامپیوتر، برای راه‌اندازی سیستم به آنها نیاز است. معمولاً ظرفیت حافظه RAM خیلی بیشتر از ROM است.

- حافظه‌های جانبی

نوع دوم حافظه‌ها، حافظه‌های جانبی می‌باشند. حافظه‌های جانبی برای ذخیره اطلاعات بطور پایدار بکار می‌روند. انواع حافظه‌های جانبی وجود دارد که از مشهورترین آنها می‌توان به دیسک سخت (Hard Disk)، فلاپی دیسک (Floppy Disk)، نوار (Tape) و سی‌دی (CD ROM) می‌توان اشاره کرد. از بین حافظه‌های جانبی، دیسک سخت کاربرد بیشتری نسبت به بقیه دارد از خواص دیسک‌های سخت می‌توان به داشتن ظرفیت بالا اشاره کرد.

- واحد پردازش مرکزی (CPU)

این واحد برای پردازش داده‌ها و برنامه‌ها بکار می‌رود. داده‌ها یا برنامه از حافظه اصلی به این واحد ارسال می‌شوند و بعد از پردازش در این واحد

---

۱. Random Access Memory

۲. Read Only Memory

دوباره به حافظه اصلی برگردانده می‌شود. CPUها غالباً از سه واحد: محاسبه و منطق، کنترل و حافظه تشکیل می‌شوند. داده‌ها با هدایت واحد کنترل به واحد محاسبه و منطق ارسال می‌شوند و پس از انجام محاسبات لازم در واحد محاسبه و منطق، دوباره با هدایت واحد کنترل به حافظه برگردانده می‌شوند. حافظه واقع در CPU به بافر معروف است و ظرفیت چندانی ندارد.

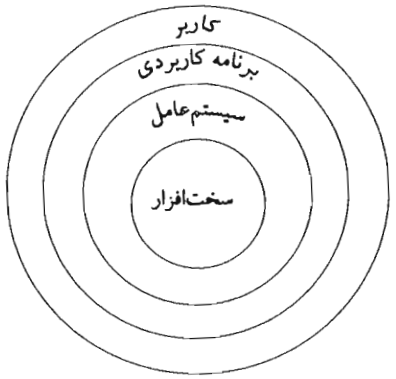
- دستگاههای خروجی

غالباً برای مشاهده نتایج پردازش روی داده‌ها نیاز به سخت‌افزارهایی را احساس می‌کنیم. این سخت‌افزارها را دستگاههای خروجی می‌نامیم. از مشهورترین نوع دستگاههای خروجی می‌توان به صفحه نمایش (Monitor)، چاپگر (Printer) و غیره اشاره کرد.

۳-۱- نرم‌افزار (Software)

نرم‌افزار یکی از بخش‌های اساسی کامپیوتر به شمار می‌آید، که در واقع سخت‌افزار را بکار می‌گیرد. بعبارت دیگر رابط بین کاربر و سخت‌افزار را نرم‌افزار می‌نامند. نرم‌افزار در حقیقت روح و جان یک کامپیوتر است، که به سخت‌افزار هویت می‌بخشد. نرم‌افزارها انواع مختلفی دارند، که مشهورترین آنها نرم‌افزارهای سیستمی و کاربردی را می‌توان نام برد.

سیستم عامل (OS: Operating System) مشهورترین نوع نرم‌افزارهای سیستمی می‌باشد. که مدیریت منابع سیستمی را بر عهده دارد. سیستم‌عامل، همچنین ارتباط بین کاربر و اجزاء سخت‌افزاری و نرم‌افزاری دیگر را برقرار می‌کند. بعد از روشن شدن کامپیوتر سیستم عامل اولین نرم‌افزاری است که در حافظه RAM لود می‌شود و بدون آن نمی‌توان از سایر نرم‌افزارهای کامپیوتر استفاده نمود. شکل ۳-۱ موقعیت سیستم عامل را نسبت به اجزای دیگر سیستم نشان می‌دهد:



شکل ۳-۱ سیستم‌عامل پل ارتباطی بین کاربر و اجزای دیگر سیستم

سیستم عامل روی حافظه‌های جانبی ذخیره می‌شود و به محض روشن شدن کامپیوتر کار خود را آغاز می‌کند. سیستم عامل وظایف متعددی دارد که به چند نمونه از آنها در اینجا اشاره می‌کنیم.

زمان‌بندی وقت CPU، تقسیم‌بندی حافظه و تخصیص آن به نرم‌افزارهای مختلف، به اشتراک گذاشتن حافظه و مدیریت سخت‌افزار و غیره از وظایف بسیار مهم سیستم‌عامل می‌باشد.

سیستم‌عامل‌های مختلفی وجود دارند، که هر کدام از آنها محصول شرکت‌های کامپیوتری معتبر می‌باشد. از متداولترین آنها می‌توان به: Ms-Dos , CP/M , Unix , Linux , Windows و غیره اشاره کرد.

امروزه معمولاًدرکامپیوترهای شخصی ازمحصولات شرکت میکروسافت استفاده می‌کنند. محصولات جدید میکروسافت در زمینه سیستم‌عامل، سیستم‌عامل جدید Windows با ویرایش‌های جدید XP , ۲۰۰۰ می‌باشد.

توجه: کاربران می‌توانند ویرایش‌های جدید سیستم‌عامل Windows (۲۰۰۰ , XP , ۲۰۰۳) را در کامپیوترهای شخصی خود نصب کنند.

نرم‌افزار توسط زبانهای برنامه‌نویسی (Programming Language) نوشته می‌شوند. زبانهای برنامه‌نویسی، یک سیستم ارتباطی هستند که توسط آنها می‌توان دستورات لازم را به ماشین انتقال داد.

هر زبان برنامه‌نویسی به مجموعه‌ای از علایم، قواعد و دستورالعمل‌ها گفته می‌شود که امکان ارتباط با کامپیوتر را جهت بیان کاری یا حل مسئله‌ای فراهم می‌کند.

انواع زبانهای مختلفی موجود است که هر کدام خواص مختلفی را دارند. در حالت کلی زبانهای برنامه‌نویسی را به سه دسته زیر تقسیم‌بندی می‌کنند:

- زبانهای سطح بالا
- زبانهای سطح پایین
- زبانهای سطح میانی

زبانهای سطح بالا، به زبانهایی گفته می‌شود که به زبان گفتاری نزدیک باشند. درک چنین زبانهایی بسیار راحت می‌باشد. تولید زبانهای سطح بالا از اواسط ۱۹۵۰ آغاز گردید. متداولترین زبانهای سطح بالا عبارتند از: FORTRAN , Pascal , Basic , PL/1 , Cobol و غیره.

زبانهای سطح بالا برای اجرا شدن در کامپیوتر نیاز به یک مترجم برای تبدیل زبان سطح بالا به زبان ماشین می‌باشند. این نرم‌افزار را کامپایلر (Compiler) می‌نامند. هر زبان برای خود کامپایلر خاص خود را دارد.

کار اصلی زبانهای برنامه‌نویسی، نوشتن برنامه (Program) می‌باشد. هر برنامه مجموعه‌ای از دستورالعمل‌های زبان را در خود دارد. برنامه نوشته شده توسط یک زبان را برنامه منبع (Source Program) می‌نامند. کامپایلر برنامه نوشته در یک زبان سطح بالا را به برنامه مقصد (Object Program) تبدیل می‌کند. به شکل ۴-۱ توجه کنید:

شکل ۴-۱ عملکرد کامپایلر

زبانهای سطح پایین زبانهایی هستند، که به زبان ماشین نزدیک‌تر هستند. کارکردن با این زبانها براحتی و سادگی زبانهای سطح بالا نیست ولی به علت نزدیکی به سخت‌افزار ماشین، برنامه‌های نوشته شده به این زبانها سرعت بالایی در زمان اجرا دارند. اسمبلی یکی از انواع زبانهای سطح پایین می‌باشد. هر ماشین با توجه به نوع سخت‌افزار زبان اسمبلی مخصوص به خود دارد.

زبانهای سطح میانی، زبانهایی مابین زبانهای سطح بالا و سطح پایین هستند این زبانها نیز مانند زبانهای سطح بالا نیاز به کامپایلر برای ترجمه دارند.

در این کتاب زبان پاسکال (Pascal) را برای آموزش و نوشتن برنامه‌ها انتخاب کردیم. این زبان که به افتخار بلز پاسکال دانشمند فرانسوی قرن هفدهم میلادی، پاسکال نامگذاری شده است، در اواخر سال ۱۹۶۰ و اوایل ۱۹۷۰ توسط پروفسور نیکلاس ویرث در انستیتو فنی فدرال سوئیس مطرح گردید. این زبان از قدرت بالایی در انجام امور علمی و تجاری برخوردار است و در بسیاری از کالج‌های دنیا جهت آموزش برنامه‌نویسی تدریس می‌گردد. این زبان توسط سازمان استاندارد ملی آمریکا در سال ۱۹۸۳ به صورت استاندارد درآمد.

فصل ۲

الگوریتمها

هدفهای کلی

- تعریف دقیق یک مسئله
- روش حل آن
- راه‌حلهای مختلف

هدفهای رفتاری

دانشجو پس از مطالعه این فصل باید بتواند:

- مسئله خود را دقیقاً درک کند.
- راه‌حلهای مختلف برای آن بیابد.
- آنرا به صورت ریاضی مدل کند.
- فلوچارت آن را رسم کند.

مثال ۱-۲ الگوریتمی بنویسید که دو عدد از ورودی دریافت کرده مجموع دو عدد را محاسبه و چاپ نماید.

در مسئله بالا داده‌های مسئله دو عدد صحیح مثلا a , b می‌باشند و هدف مسئله، به دست آوردن مجموع (sum) دو عدد است و در نهایت چاپ مجموع در خروجی .

معمولا در نوشتن الگوریتم ،از شروع برای انجام کار الگوریتم استفاده می‌کنند و برای هر مرحله شماره‌ای را اختصاص می‌دهند :

خروجی‌ها	انجام محاسبات	ورودیها
مجموع دو عدد	جمع دو عدد	a , b

0- شروع

۱- a, b را بخوان.

۲- مجموع a , b را محاسبه و در sum قرار بده.

۳- sum را در خروجی چاپ کن

۴- پایان

مثال۲-۲ الگوریتمی بنویسید که سه عدد از ورودی دریافت کرده مجموع و میانگین سه عدد را محاسبه و چاپ کند.

خروجی‌ها	انجام محاسبات	ورودیها
چاپ مجموع	محاسبه مجموع	a
چاپ میانگین	محاسبه میانگین	b
		c

الگوریتم بصورت زیر می‌باشد:

شروع

۱- سه عدد از ورودی بخوان.

۲- مجموع سه عدد را محاسبه و در sum قرار بده.

۳- sum را بر سه تقسیم کرده،در ave قرار بده.

۴- ave , sum را در خروجی چاپ کن.

۵- پایان.

نکته: نتیجه انجام محاسبات روی ورودیها باید داخل متغیرها قرار گیرد.

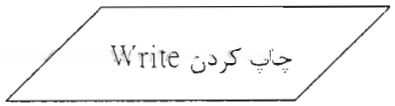
مثلا مجموع دو عدد، داخل یک متغیر جدید بنام sum قرار می‌گیرد.

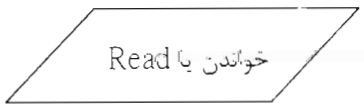
همانطور که در مثال‌ها مشاهده کردید الگوریتم برای حل مسئله، در حقیقت روش حل مسئله می‌باشد. که با ترتیب مشخص و مراحل معین انجام می‌گیرد.

معمولا درک یک الگوریتم با شکل راحتتر از نوشتن آن بصورت متن می‌باشد. لذا الگوریتم را با فلوچارت(flowchart)نمایش می‌دهند. فلوچارت از شکل‌های زیر تشکیل می‌شود.

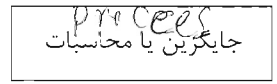
- علامت‌های شروع و پایان: که معمولا از یک بیضی استفاده می‌کنند:

- علامتهای ورودی و خروجی: که معمولا از متوازی‌الاضلاع استفاده می‌شود:





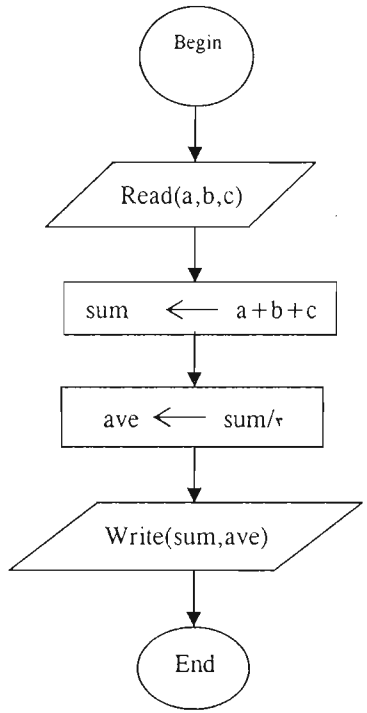
- علامتهای محاسباتی و جایگزینی: برای نمایش دستورات جایگزینی و محاسباتی از مستطیل استفاده می‌کنند:



- علامت شرط: برای نمایش شرط از لوزی استفاده می‌شود:
- علامت اتصال: برای اتصال شکل‌های مختلف بهم از فلش‌های جهت‌دار استفاده می‌کنند.

نکته: برای اینکه با دستورات زبان پاسکال آشنا شویم در داخل فلوچارت‌ها از دستورات پاسکال استفاده می‌کنیم.

مثال ۳ـ۲ فلوچارت مثال ۱۰۲ را رسم نمائید.

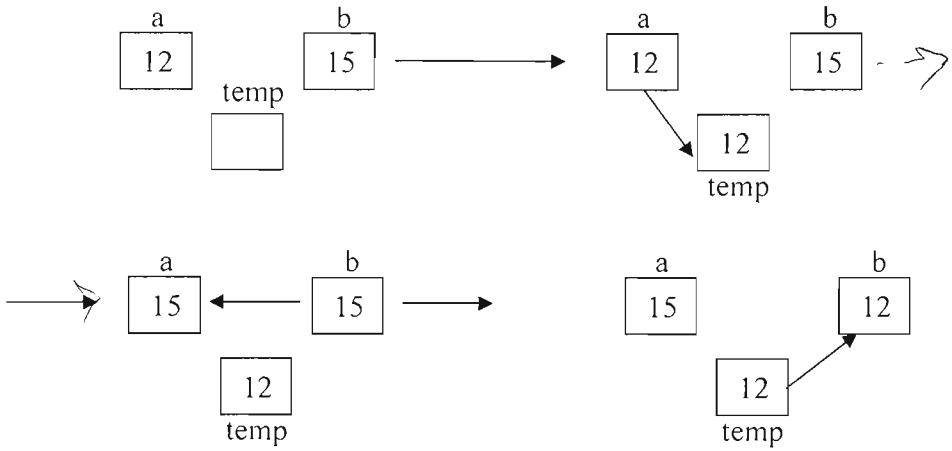


به منظور اجرای درستی الگوریتم معمولا آن را با تعدادی ورودی تست می‌کنند.

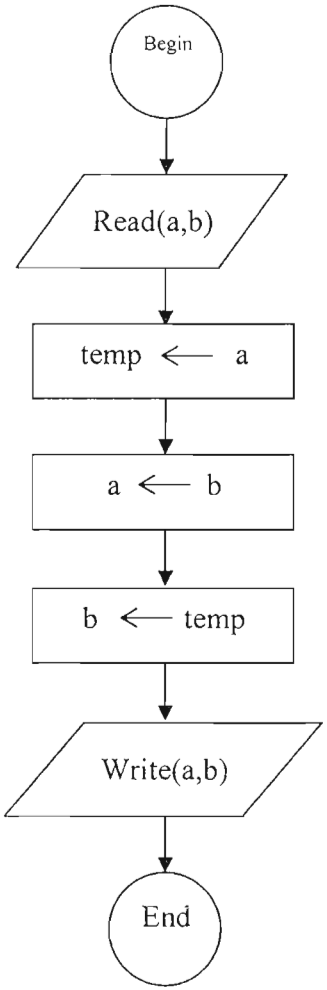
A	b	c	sum	ave
12	3	6	21	7.00

مثال۴-۲ فلوچارتی رسم نمائید که دو عدد از ورودی دریافت کرده، سپس محتویات دو عدد را با هم جابجا نماید.

برای حل این مسئله a , b را دو متغیر که در آنها دو عدد خوانده شده، قرار می‌گیرند در نظر می‌گیریم. سپس با استفاده از یک متغیر کمکی محتویات این دو عدد را جابجا می‌کنیم مثلا :



فلوچارت مسئله بالا بصورت زیر خواهد بود:



تمرین

۱ــ فلوچارتی رسم نمائید که طول و عرض مستطیل را از ورودی دریافت کرده محیط و مساحت آنرا محاسبه و چاپ کند.

۲ــ فلوچارتی رسم نمائید که شعاع دایره‌ای را از ورودی دریافت کرده، محیط و مساحت آنرا محاسبه و چاپ نماید.

۳ــ فلوچارتی رسم کنید که سه عدد first, second , third را از ورودی دریافت کرده، محتویات آنها را جابجا نموده، حاصل را در خروجی چاپ کند.

٤ــ فلوچارتی رسم نمائید که دو عدد از ورودی دریافت کرده، سپس محتویات دو عدد را بدون استفاده از متغیر کمکی جابجا کند.

۵ــ فلوچارتی رسم نمائید که عددی (درجه حرارت برحسب سانتیگراد) را از ورودی دریافت کرده سپس آنرا به درجه فارنهایت تبدیل کند.

۳ـ۲ دستورالعمل‌های شرطی

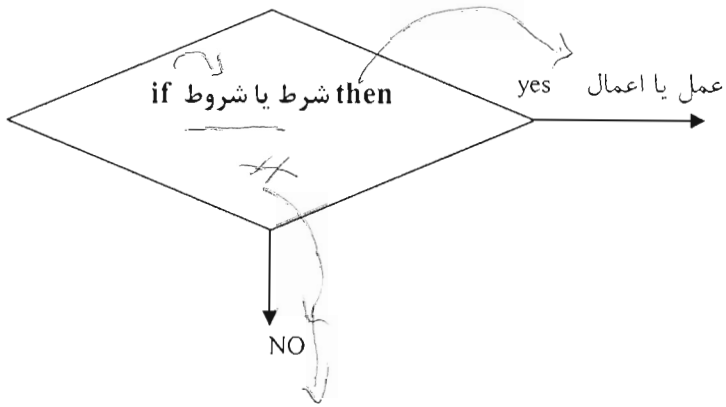
در حل بسیاری از مسائل یا تقریباً تمام مسائل نیاز به استفاده از شروط جزء، نیازهای اساسی محسوب می‌شود. همانطور که ما خودمان در زندگی روزمره با این شرط‌ها سرکار داریم. بطور مثال اگر هوا ابری باشد ممکن است چنین سخن بگوییم:

اگر هوا بارانی باشد سپس چتری برمی‌دارم.

در غیر اینصورت چتر برنمی‌دارم.

غالباً دستورات شرطی از دو قسمت تشکیل می‌شود، یکی در صورتی که شرط برقرار باشد عملی یا اعمالی انجام می‌شود و دیگری زمانی که شرط برقرار نباشد ممکن است عمل یا اعمال دیگری انجام پذیرد.

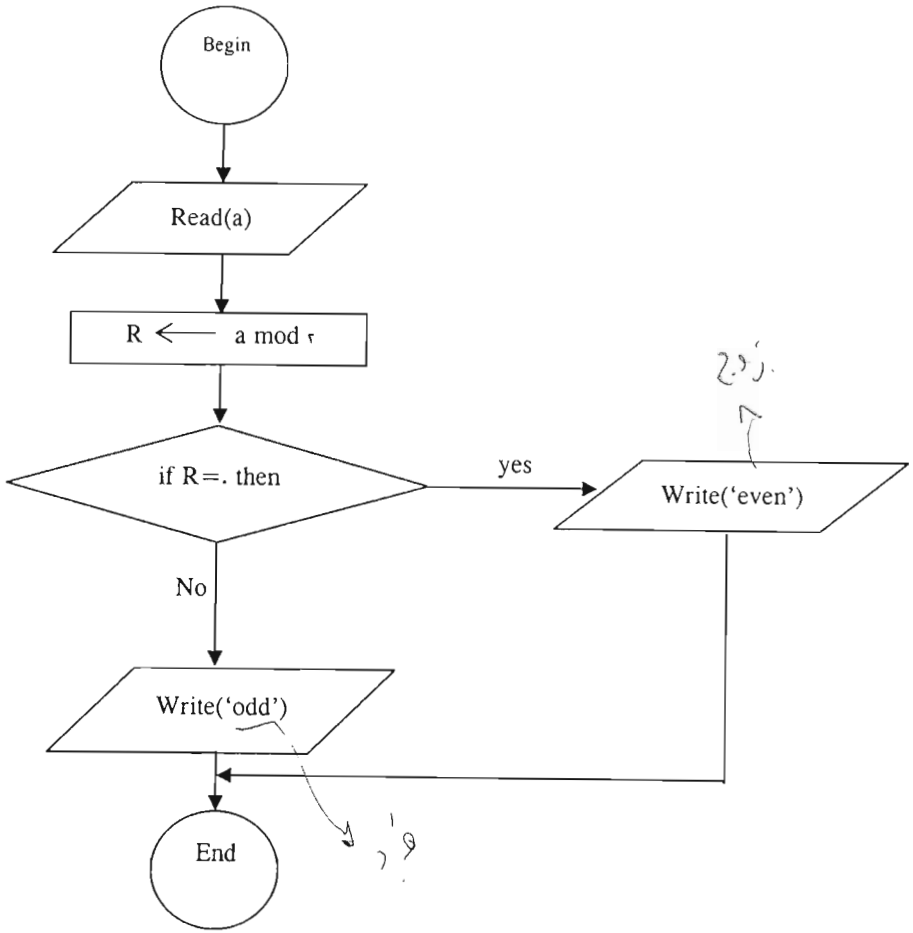
در حالت کلی شرط را بصورت زیر نمایش می‌دهند:



عمل یا اعمال بعدی

مثال ۵ ۲ فلوچارتی رسم نمائید که عددی را از ورودی دریافت کرده، فرد یا زوج بودن آن را تشخیص دهد.

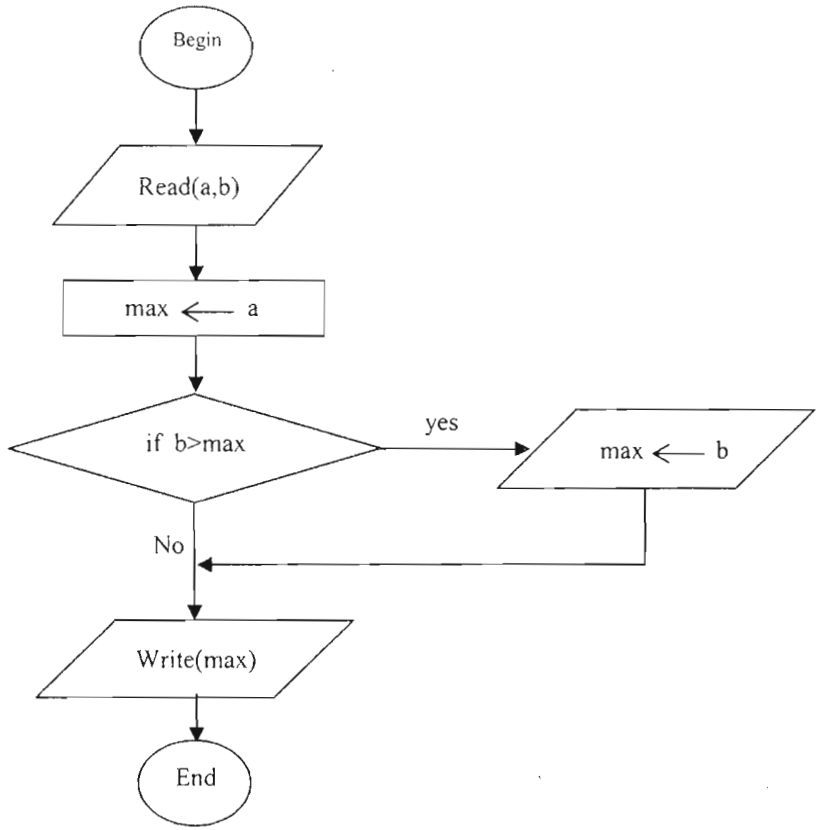
در مسئله بالا عدد خوانده شده را بر ۲ تقسیم می‌کنیم، اگر باقیمانده عدد بر ۲ برابر صفر بود، عدد زوج است ، در غیر اینصورت عدد فرد می‌باشد. برای محاسبه باقیمانده در پاسکال از mod استفاده می‌کنند.



در فلوچارت بالا در صورتی که عدد زوج باشد قسمت yes شرط اجرا می‌شود و در خروجی پیغام زوج بودن چاپ می‌شود و بعداً الگوریتم پایان می‌یابد. در صورتی که عدد فرد باشد، پیغام فرد بودن در خروجی نمایش داده می‌شود.

نکته: در کنترل اجرای یک فلوچارت با ورودیهای مختلف باید خروجی مناسب تولید شود و دو نهایت باید به End ختم شود در ضمن هر فلوچارت فقط یک End دارد.

مثال ۶-۲ فلوچارتی رسم کنید که دو عدد از ورودی دریافت کرده بزرگترین عدد را پیدا کرده در خروجی چاپ نماید.

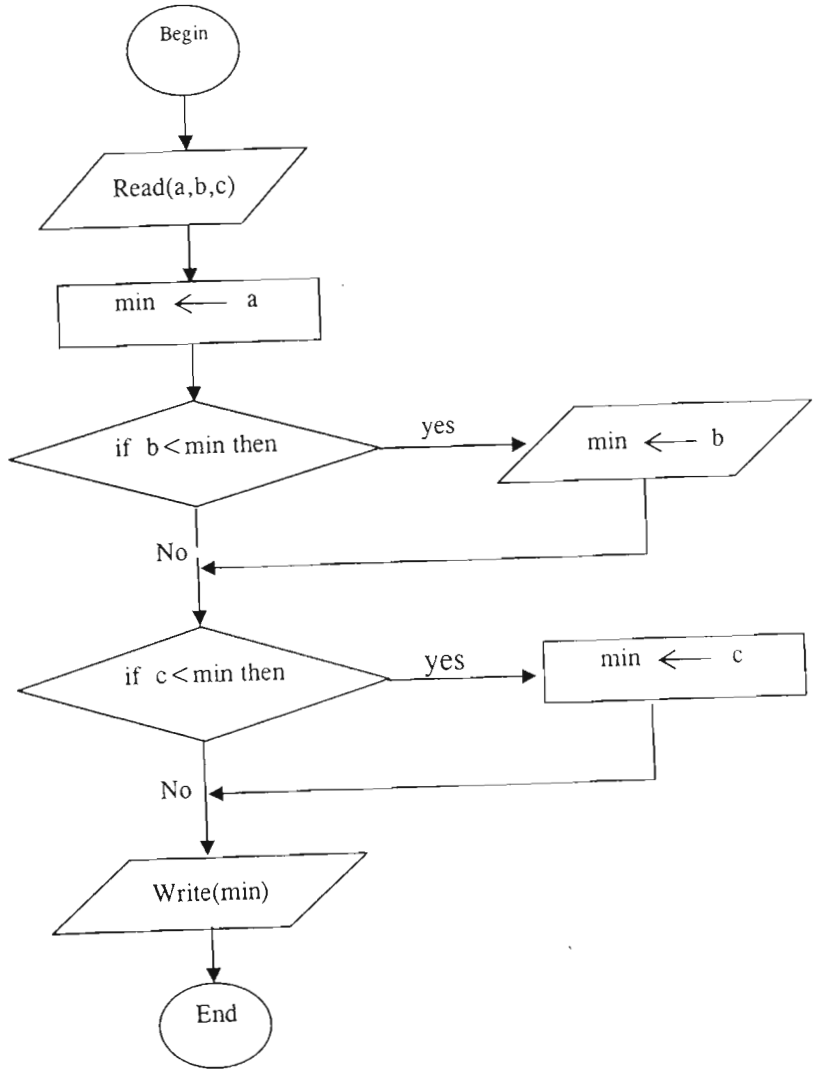


در فلوچارت بالا نخست اولین ورودی را max در نظر می‌گیریم، سپس ورودی دوم را با max مقایسه می‌کنیم اگر بزرگتر از max بود، دومی را در max قرار می‌دهیم.

نمونه اجرای فلوچارت بالا بصورت زیر می‌باشد.

	a	b	max	خروجی
۱	۱۲	۱۵		
۲			۱۲	
۳			۱۵	
٤			۱۵	
				۱۵

مثال ۷-۲ فلوچارتی رسم نمائید که سه عدد از ورودی دریافت کرده، کوچکترین عدد را یافته در خروجی چاپ نماید:



نمونه اجرای فلوچارت بالا بصورت زیر می‌باشد:

خروجی	M	c	b	a	
		۱۷	۱۱	۱۲	۱
	۱۲				۲
	۱۱				۳
	۱۱				٤
	۱۱				٥
۱۱					

تمرین

۱- فلوچارتی رسم کنید که عددی را از ورودی دریافت کرده، قدر مطلق عدد را در خروجی چاپ کند.

۲- فلوچارتی رسم نمائید که عددی از ورودی دریافت کرده مثبت، منفی یا صفر بودن عدد را تشخیص داده، در خروجی با پیغام مناسب چاپ کند.

۳- فلوچارتی رسم نمائید که عددی را از ورودی دریافت کرده، بخشپذیری آن بر ۳ و ۵ را بررسی نماید.

۴ـ فلوچارتی رسم نمائید که دو عدد و یک عملگر را از ورودی دریافت کرده کار یک ماشین حساب ساده را شبیه‌سازی نماید.

۵- فلوچارتی رسم نمائید که سه عدد از ورودی دریافت کرده، کوچکترین و بزرگترین عدد بین سه عدد خوانده شده را یافته در خروجی چاپ نماید.

۶ـ فلوچارتی رسم نمائید که ضرایب یک معادله درجه دوم را از ورودی دریافت کرده، ریشه‌های آن را محاسبه در خروجی چاپ کند.

۲ـ۲ حلقه‌ها

در حل بسیاری از مسائل با عملیاتی روبرو می‌شویم ، که نیاز به تکرار دارند و عمل تکرار آنها به تعداد مشخصی انجام می‌گیرد. فرض کنید، بخواهیم میانگین ۱۰۰ عدد را محاسبه کنیم، در اینصورت منطقی بنظر نمی‌رسد که ۱۰۰ متغیر مختلف را از ورودی دریافت کنیم سپس آنها را جمع کنیم.

در چنین مسائلی از حلقه‌ها استفاده می‌کنند. حلقه‌ها انواع مختلفی دارند که در این بخش به دو نوع آنها اشاره می‌کنیم.

- **حلقه‌های نوع اول (به حلقه‌های for معروفند)**

در این نوع حلقه‌ها تعداد تکرار مشخص می‌باشد این حلقه از اجزاء زیر تشکیل می‌شود:

۱ـ اندیس حلقه

۲ـ مقدار اولیه برای اندیس حلقه

۳- مقدار افزاینده برای اندیس حلقه (معمولا یک واحد در هر مرحله)

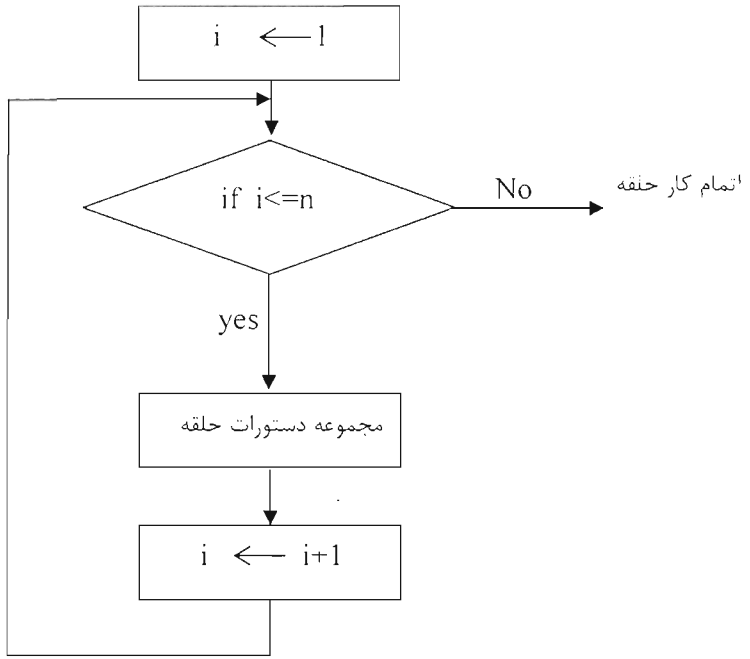
۴ـ مقدار نهایی (تعداد تکرا حلقه)

۵ـ شرطی برای کنترل تعداد تکرار حلقه

این حلقه‌ها را غالباً با فلوچارت بصورت زیر نمایش می‌دهند:

i ◄——— اندیس حلقه

n ◄——— مقدار نهایی

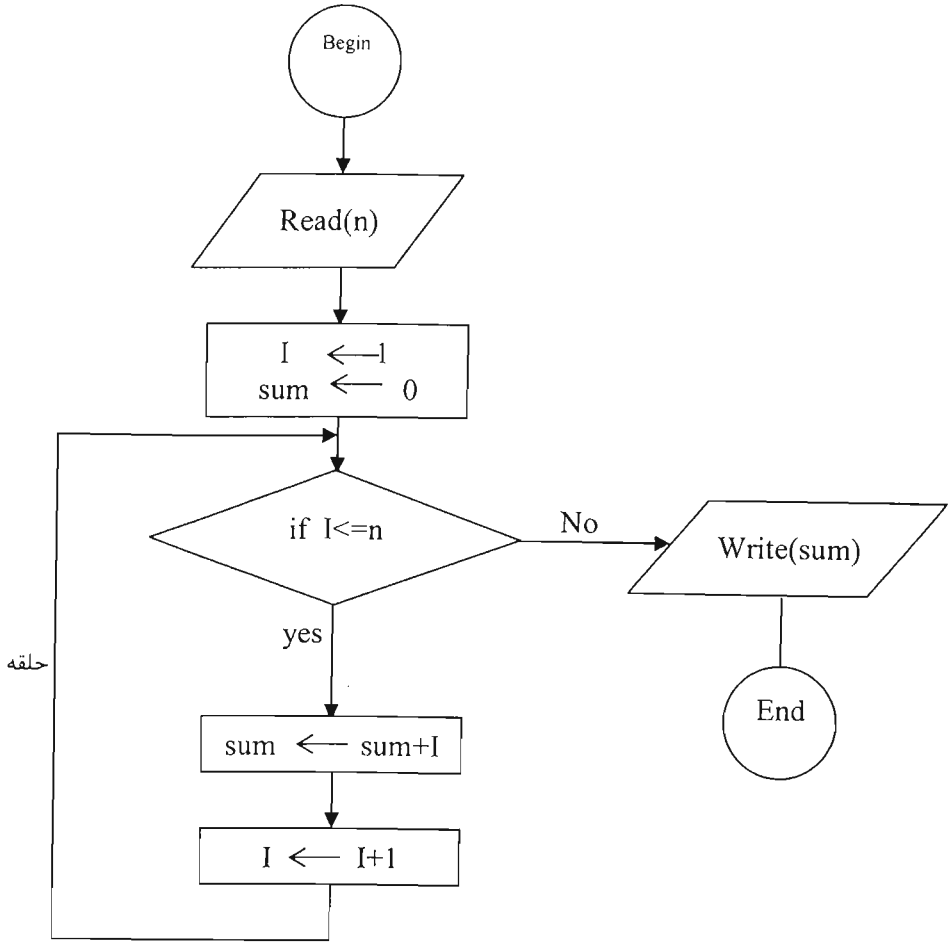


در این حلقه‌ها ابتدا مقدار اولیه اندیس (که غالباً در پاسکال از یک شروع می‌شود) را در متغیر مربوطه قرار می‌دهیم، سپس آن را با مقدار نهایی مقایسه می‌کنیم. اگر کمتر بود، مجموعه دستورات لازم که قرار است، داخل حلقه اجرا شود را مشخص می‌کنیم و بعد از اتمام اجرای دستورات مقدار اندیس حلقه را یک واحد افزایش داده، دوباره با مقدار نهایی مقایسه می‌کنیم. این روند تا زمانیکه مقدار اندیس حلقه به مقدار نهایی نرسیده باشد تکرار خواهد شد.

مثال ۲ ۸ فلوچارتی رسم نمائید که عدد n را از ورودی دریافت کرده، مجموع اعداد از یک تا n را محاسبه کند.

i ⟵ اندیس حلقه

n ⟵ مقدار نهایی

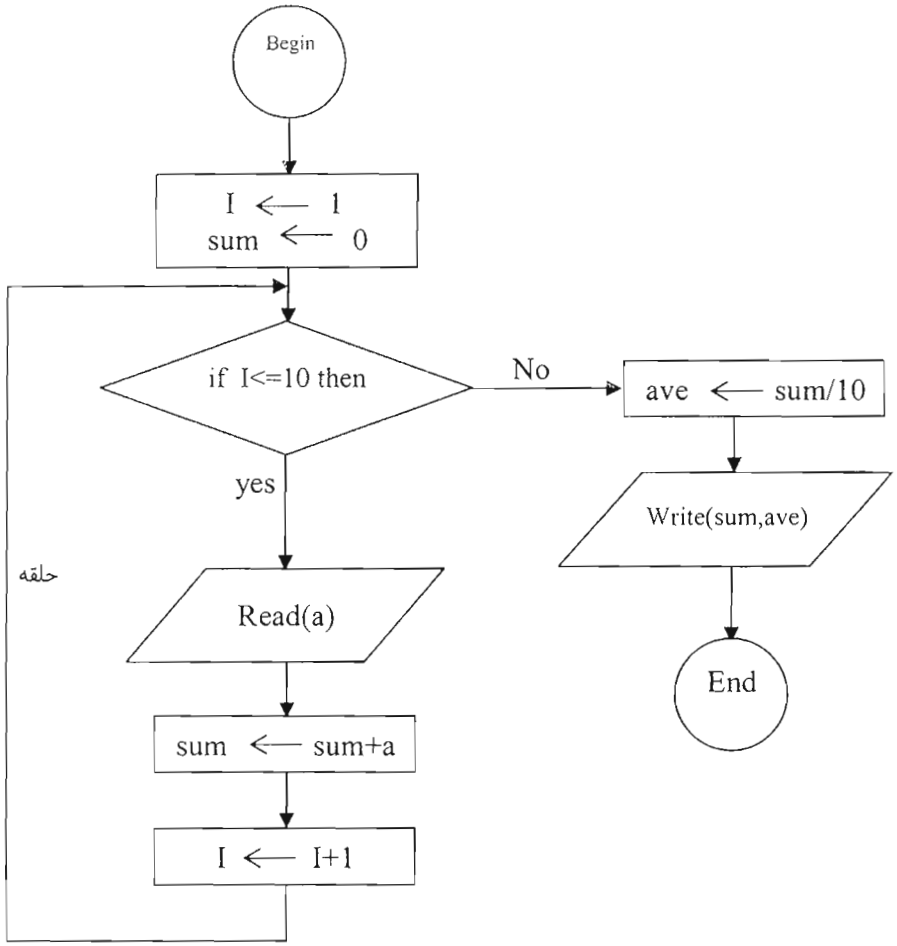


در مثال بالا مجموع اعداد از یک تا N ، که مقدار N را از ورودی می‌خوانیم محاسبه می‌شود.

نمونه اجرای فلوچارت بالا بصورت زیر است:

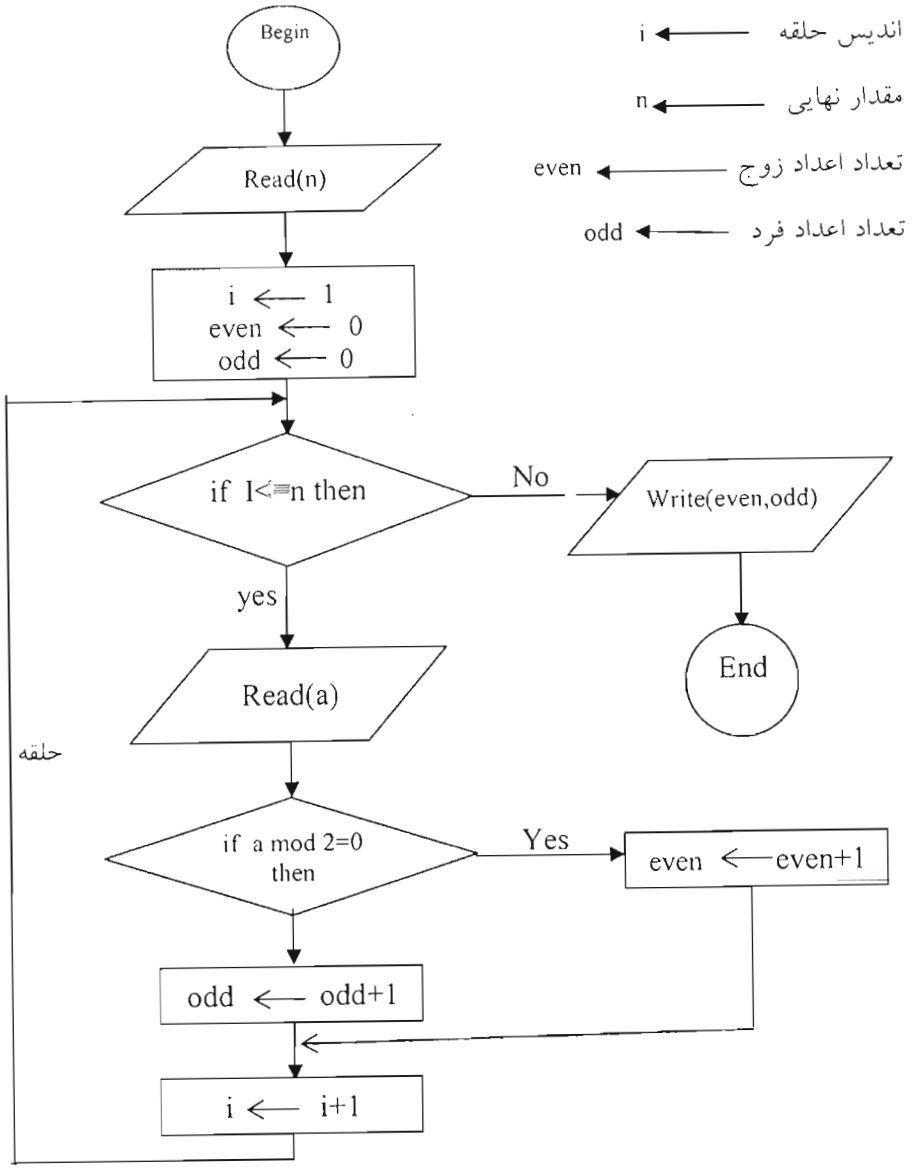
	N	I	sum	خروجی
1	5	1	0	
2		2	1	
3		3	3	
4		4	6	
5		5	10	
6		6	15	
7				15

مثال ۲ ۹ فلوچارتی رسم نمائید که ۱۰ عدد از ورودی دریافت کرده، مجموع و میانگین ۱۰ عدد را محاسبه و چاپ کند.



در فلوچارت بالا ۱۰ عدد مختلف از ورودی خوانده می‌شود. عمل خواندن داخل حلقه تکرار می‌شود لذا ۱۰ ورودی مختلف خوانده می‌شود و مجموع آنها محاسبه می‌گردد.

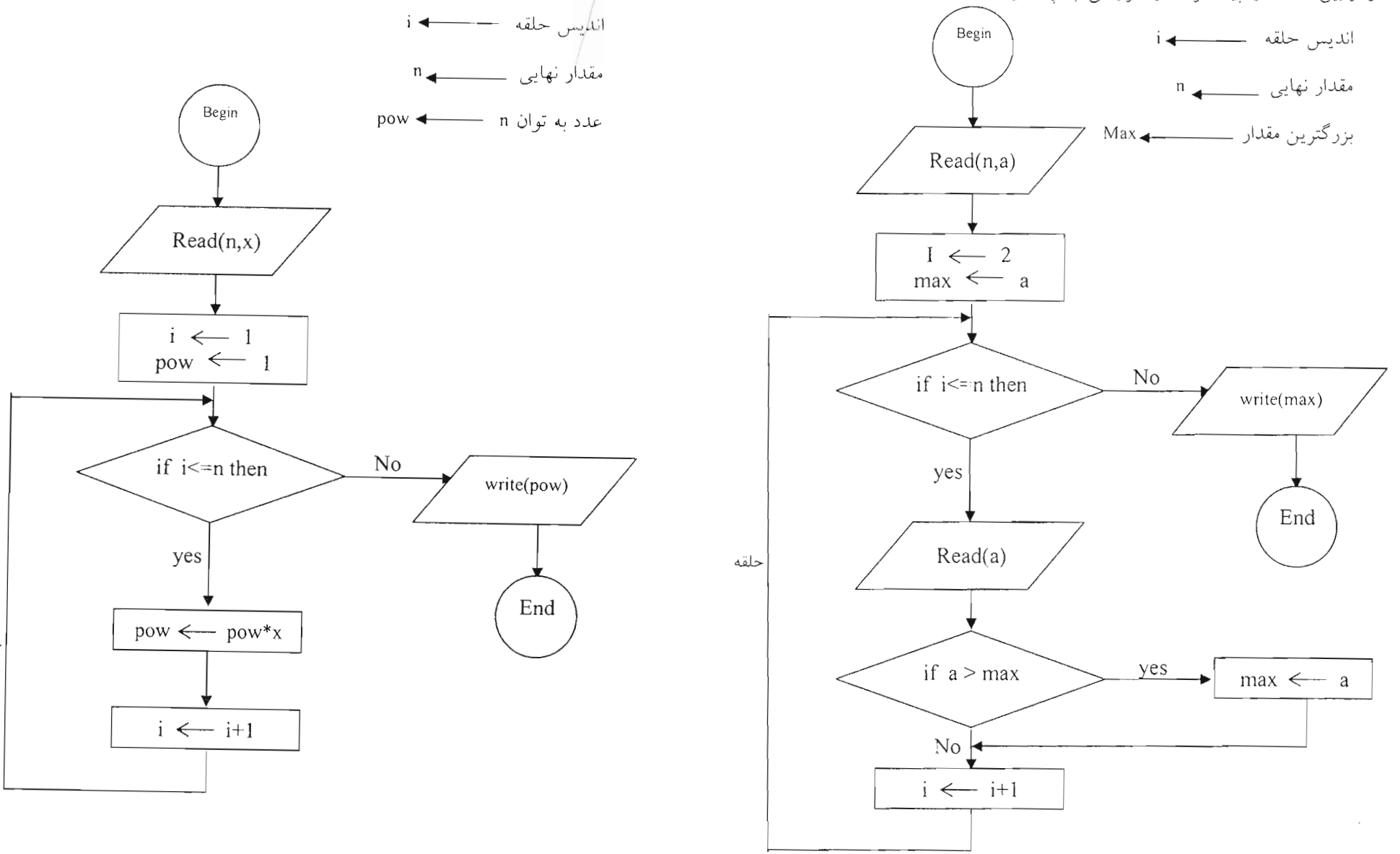
مثال ۱۰-۲ فلوچارتی رسم کنید که N عدد از ورودی دریافت کرده سپس تعداد اعداد زوج و فرد را شمرده، در خروجی چاپ نماید.



فلوچارت بالا را با ورودیهای زیر تست می‌کنیم:

	a	n	even	odd	I	خروجی
1		4	0	0	1	
2	5		0	1	2	
3	12		1	1	3	
5	17		1	2	4	
6	29		1	3	5	
7						1 , 3

مثال ۱۱-۲ فلوچارتی رسم کنید که n عدد از ورودی دریافت کرده، بزرگترین مقدار از بین n عدد را پیدا کرده در خروجی چاپ نماید.

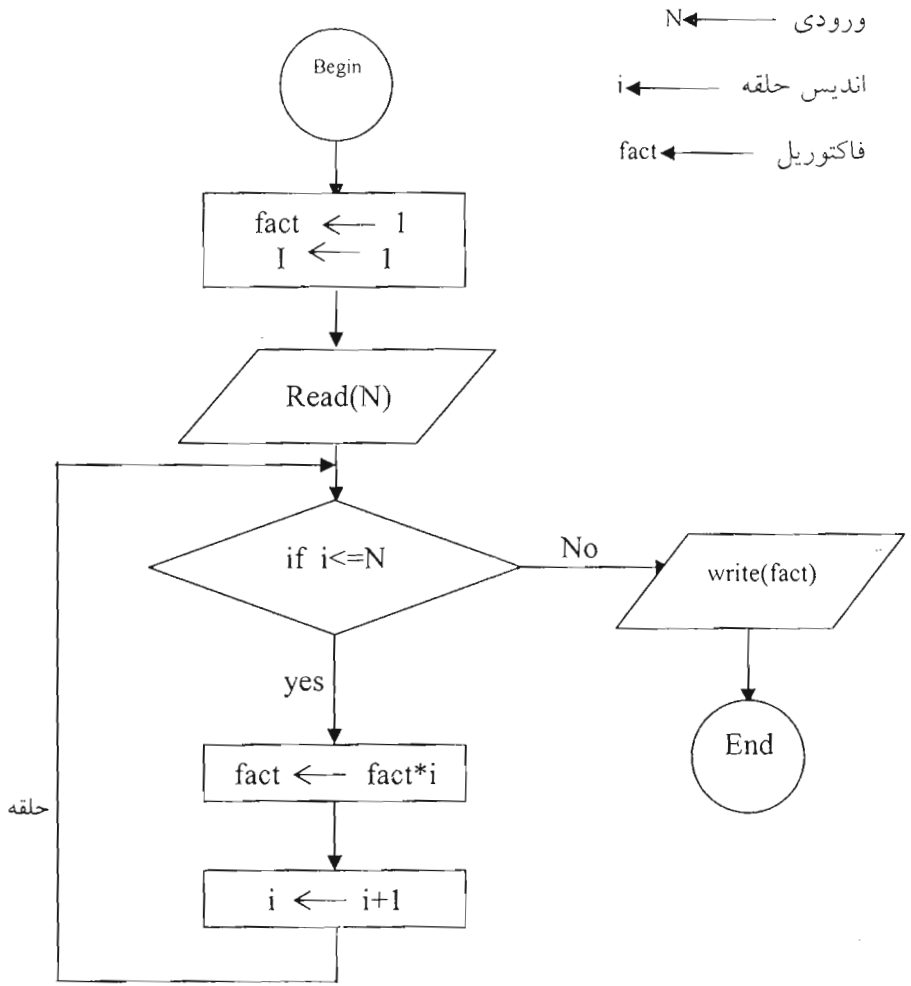


مثال ۱۲-۲ فلوچارتی رسم نمائید که x ، n ، دو عدد صحیح مثبت را از ورودی دریافت کرده سپس x به توان n را محاسبه کند.

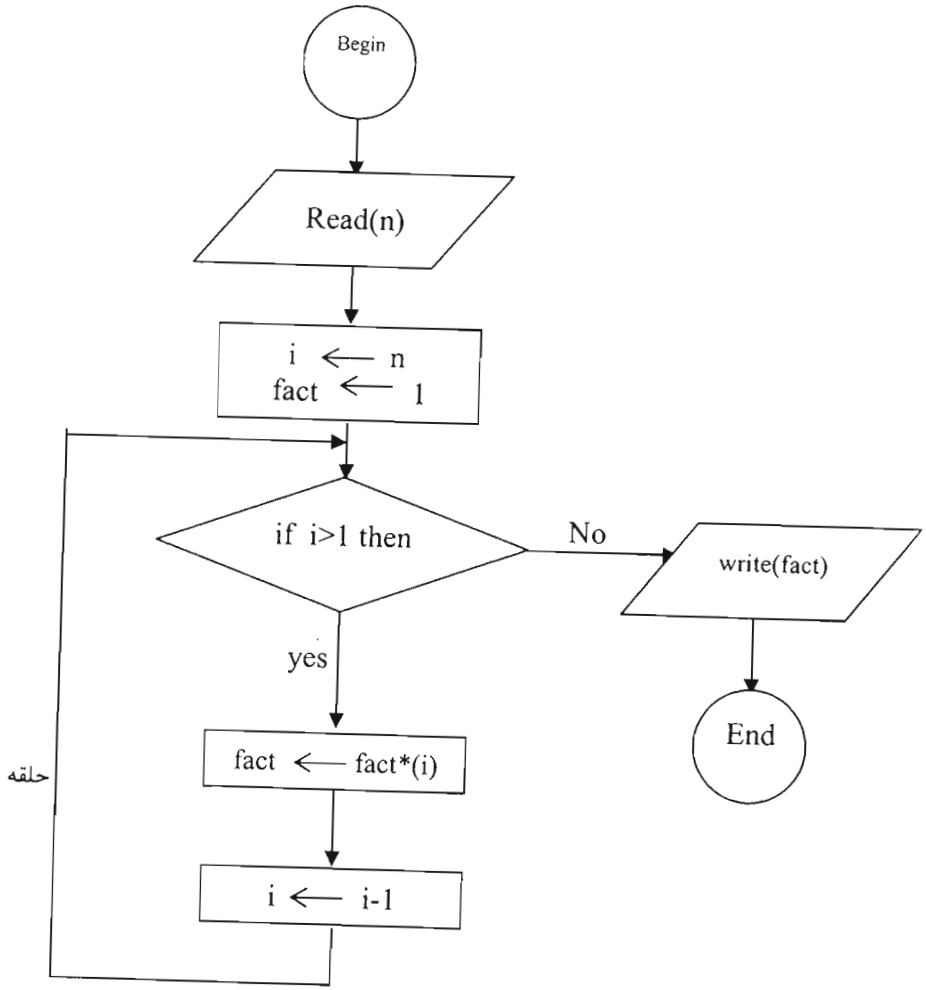
مثال ۱۳-۲ فلوچارتی رسم کنید که عدد N را از ورودی دریافت کرده، فاکتوریل آنرا محاسبه نماید.

فاکتوریل یک عدد در حالت کلی برابر است با:

$$N! = N \times (N-1) \times \ldots \times 1$$



فلوچارت بالا را می‌توان بهتر نوشت

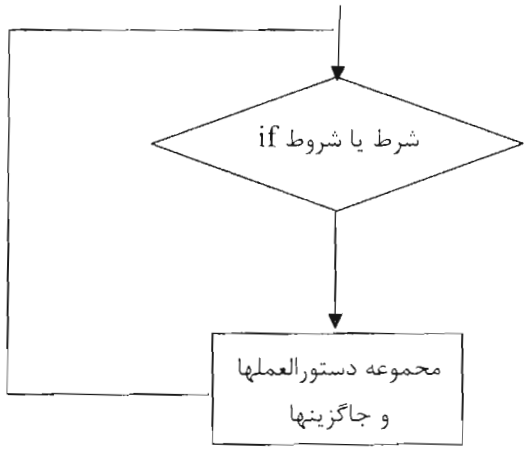


سوال: آیا می‌توان فلوچارت بالا را به شکل بهتری نوشت .

- **حلقه‌هایی که تعداد تکرار آنها مشخص نیست. (در پاسکال به حلقه while مشهورند.)**

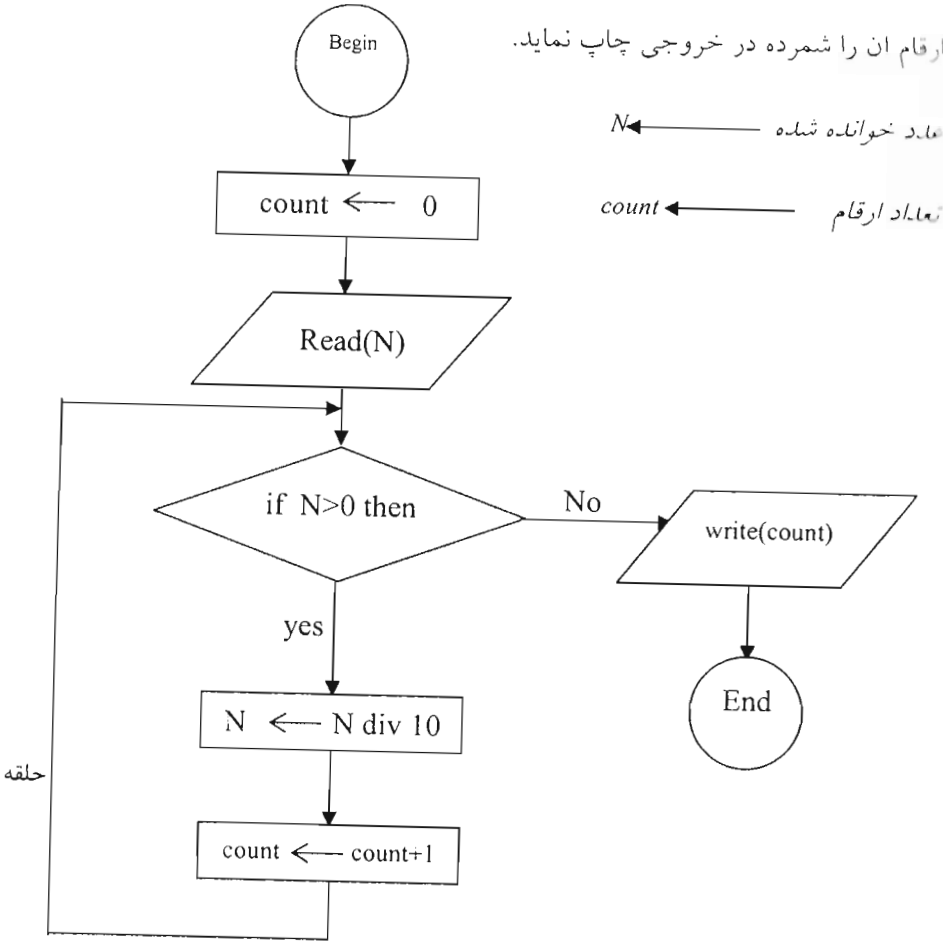
در این حلقه‌ها با توجه به ورودی، تعداد تکرار مشخص می‌شود. و دقیقاً نمی‌توان تعداد تکرار حلقه را بدون ورودی معین کرد. این حلقه ها فقط شامل شرطی هستند که تا زمانیکه برقرار باشد حلقه اجرا می‌شود.

در حالت کلی این نوع حلقه‌ها بصورت زیر نمایش داده می‌شوند:



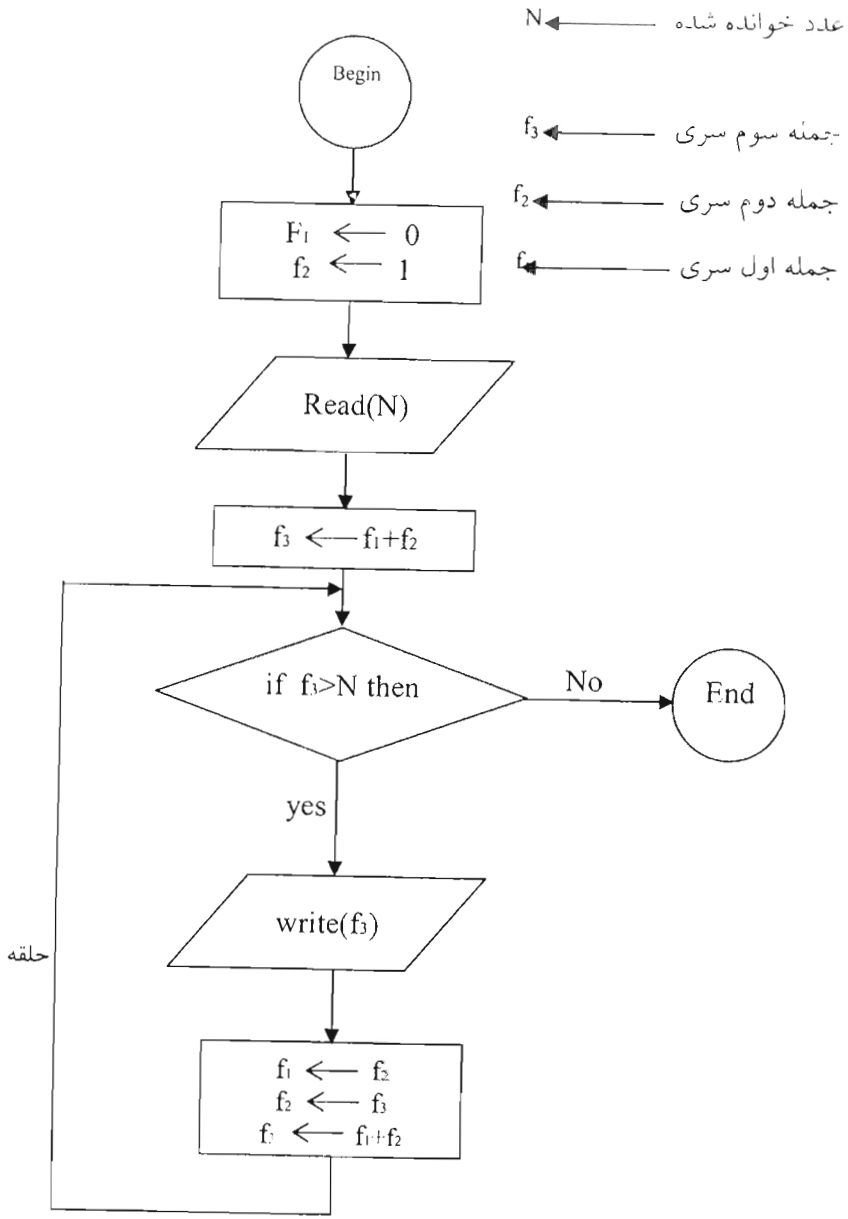
در این حلقه‌ها وجود اندیس الزامی نیست ولی می‌توان حلقه‌های نوع اول را هم با این نوع حلقه‌ها شبیه‌سازی کرد.

مثال: ۱۴ ۲ فلوچارتی رسم کنید که عددی را از ورودی دریافت کرده سپس تعداد ارقام ان را شمرده در خروجی چاپ نماید.



در فلوچارت بالا div عملگری برای تقسیم صحیح در پاسکال می‌باشد.

همانطور که ملاحظه می‌کنید تا زمانیکه N > 0 باشد عمل تقسیم ادامه پیدا می‌کند. و در عین حال تعداد ارقام عدد شمرده می‌شود.



خروجی فلوچارت بالا برای N=1023 بصورت زیر است:

	N	count	خروجی
۱	1023	0	
	102	1	
	10	2	
	1	3	
	0	4	
			4

مثـــال ۱۵-۲ فلوچـــارتی رســـم نمائیـــد کـــه عـــددی از ورودی دریافـــت کـــرده، ســـری فیبوناچی قبل از آنرا تولید نماید.

سری بصورت زیر می باشد :

0 1 1 2 3 5 8 13 ...

و در حالت کلی جملات سری بصورت:

$$f_k=f_{k-1}+f_{k-2}$$

مثال ۱۶-۲ فلوچارتی رسم نمائید که عددی از ورودی دریافت کرده، اول بودن عدد را بررسی نماید.

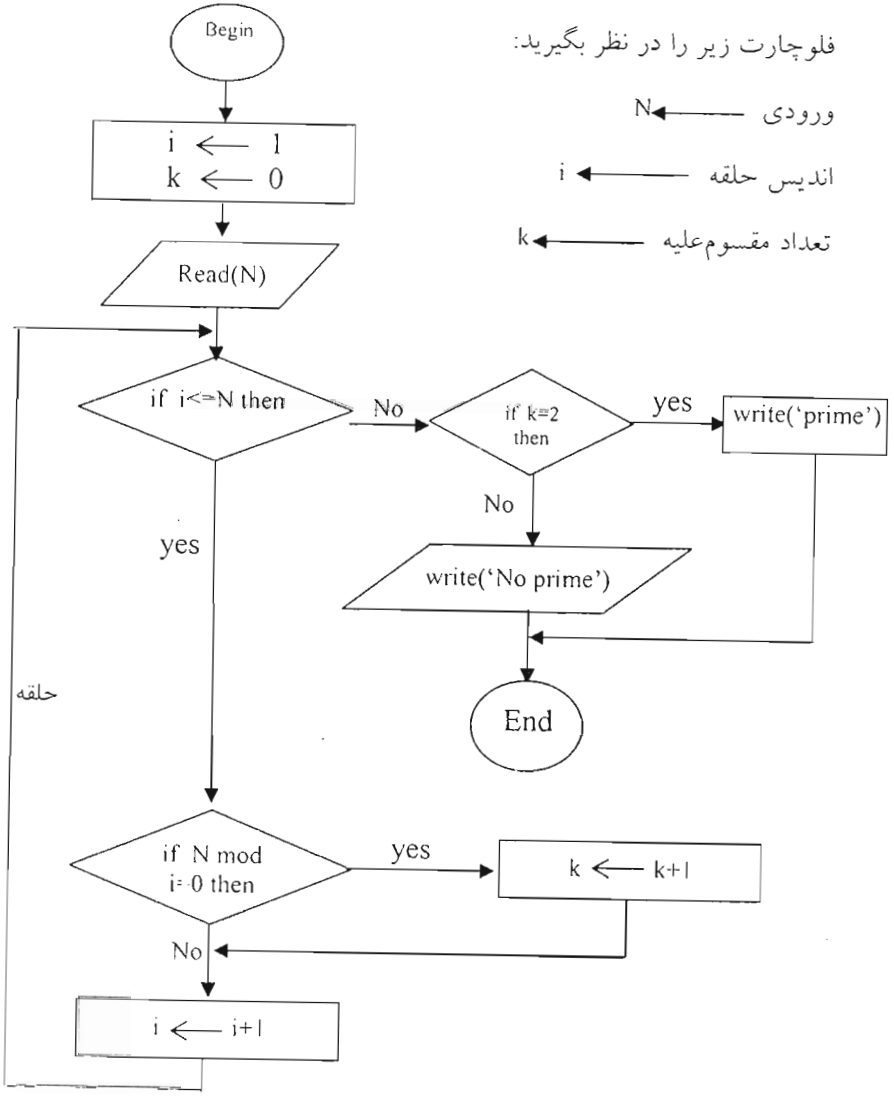
همانطور که می‌دانیم عدد اول، عددی است که دو مقسوم‌علیه بیشتر نداشته باشد.

فلوچارت زیر را در نظر بگیرید:

ورودی N

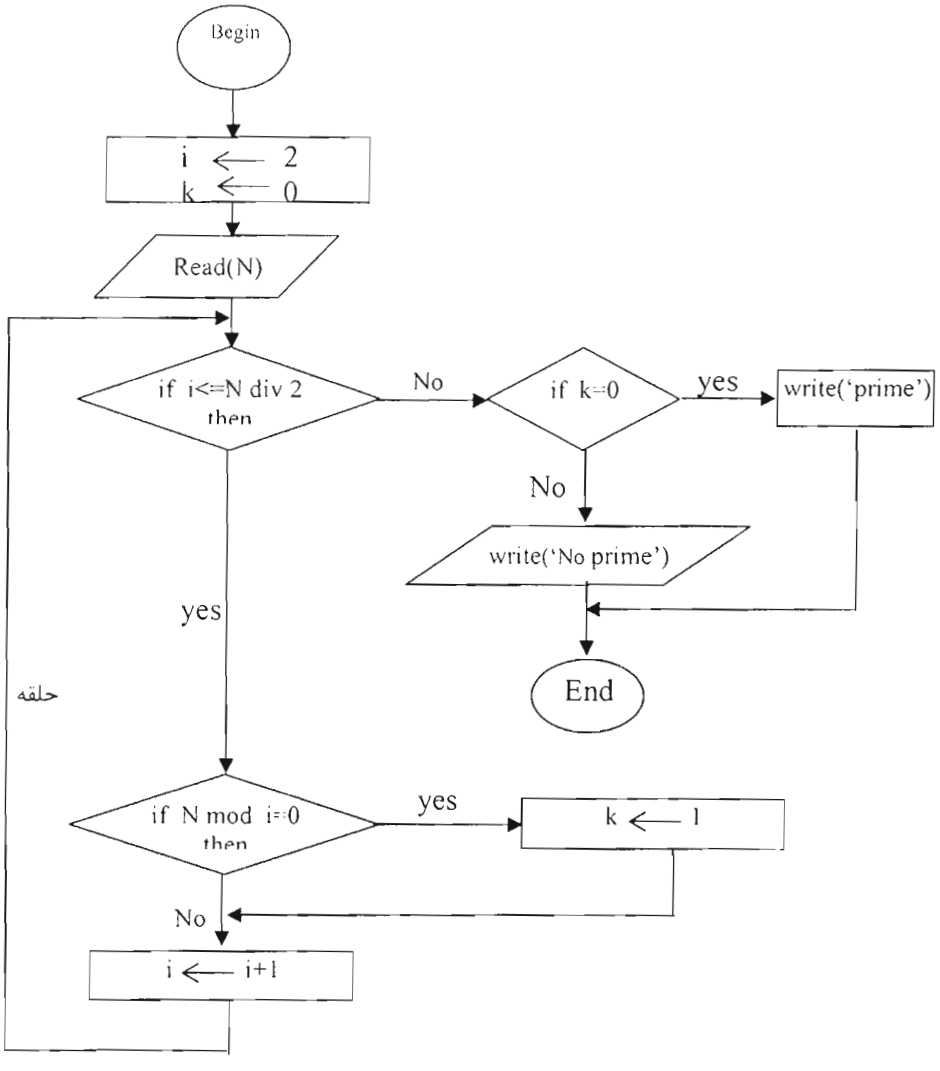
اندیس حلقه i

تعداد مقسوم‌علیه k



همانطور که ملاحظه می‌کنید فلوچارت بالا تعداد مقسوم‌علیه‌های عدد N را محاسبه می‌کند، اگر تعداد مقسوم‌علیه‌های عدد، دو تا باشد عدد اول است، در غیر اینصورت اول نیست.

فلوچارت بالا را می‌توان بصورت زیر بهتر کرد:



سوال: فلوچارتی رسم نمائید که کد آن از فلوچارت بالا بهتر باشد.

در فلوچارت بالا حلقه تقریباً به اندازه نصف فلوچارت قبلی تکرار می‌شود لذا زمان اجرای بهتری دارد.

تمرین

۱- فلوچارتی رسم نمائید که عددی از ورودی دریافت کرده، کامل بودن آنرا بررسی نماید. (عدد کامل، عددی است که مجموع مقسوم‌علیه‌های آن با خودش برابر باشد.)

۲- فلوچارتی رسم کنید که N را از ورودی دریافت کرده، N جمله سری فیبوناچی را تولید نماید.

۳- فلوچارتی رسم نمائید که X , N را از ورودی خوانده $X^N / N!$ را محاسبه کند.

۴- فلوچارتی رسم نمائید که N عدد از ورودی دریافت کرده، تعداد اعداد مثبت، منفی و صفر را شمرده در خروجی چاپ کند.

۵- فلوچارتی رسم نمائید که عددی را از ورودی دریافت کرده، تشخیص دهد که عدد خوانده شده جزء سری فیبرناچی هست یا نه؟

۶- فلوچارتی رسم نمائید که دو عدد M , N را از ورودی خوانده، بزرگترین مقسوم‌علیه مشترک دو عدد را محاسبه و چاپ کند.

۷- فلوچارتی رسم نمائید که N عدد از ورودی دریافت کرده، بزرگترین مقدار از بین N عدد و تعداد تکرار آن را محاسبه و چاپ نماید.

۸- فلوچارتی رسم نمائید که عددی را از ورودی دریافت کرده، بررسی کند که عدد خوانده شده، مربع کامل است یا نه؟

۹- فلوچارتی رسم کنید که عددی از ورودی خوانده، آن را به مبنای ۲ ببرد.

۵ـ۲ حلقه‌های تودرتو

الگوریتم‌هایی که تا حال بکار بردیم، فقط شامل یک حلقه بودند.

در صورتی که در بسیاری از مسائل ممکن است نیاز به استفاده از چند حلقه در داخل هم داشته باشیم. در این نوع حلقه‌ها باید دقت بیشتری به خرج دهیم، تا مشکلی پیش نیاید. اگر از حلقه‌های نوع اول بصورت تودرتو استفاده کنیم در اینصورت برای هر حلقه شرط نهایی و اندیس اولیه جداگانه باید تعریف کنیم .

در حلقه‌های تودرتو به ازای یکبار تکرار حلقه اولیه، حلقه داخلی به اندازه مقدار نهایی خود تکرار می‌شود. در کل اگر حلقه اولیه n بار تکرار شود و حلقه داخلی m بار، در اینصورت کل حلقه $n \cdot m$ بار تکرار خواهد شد.

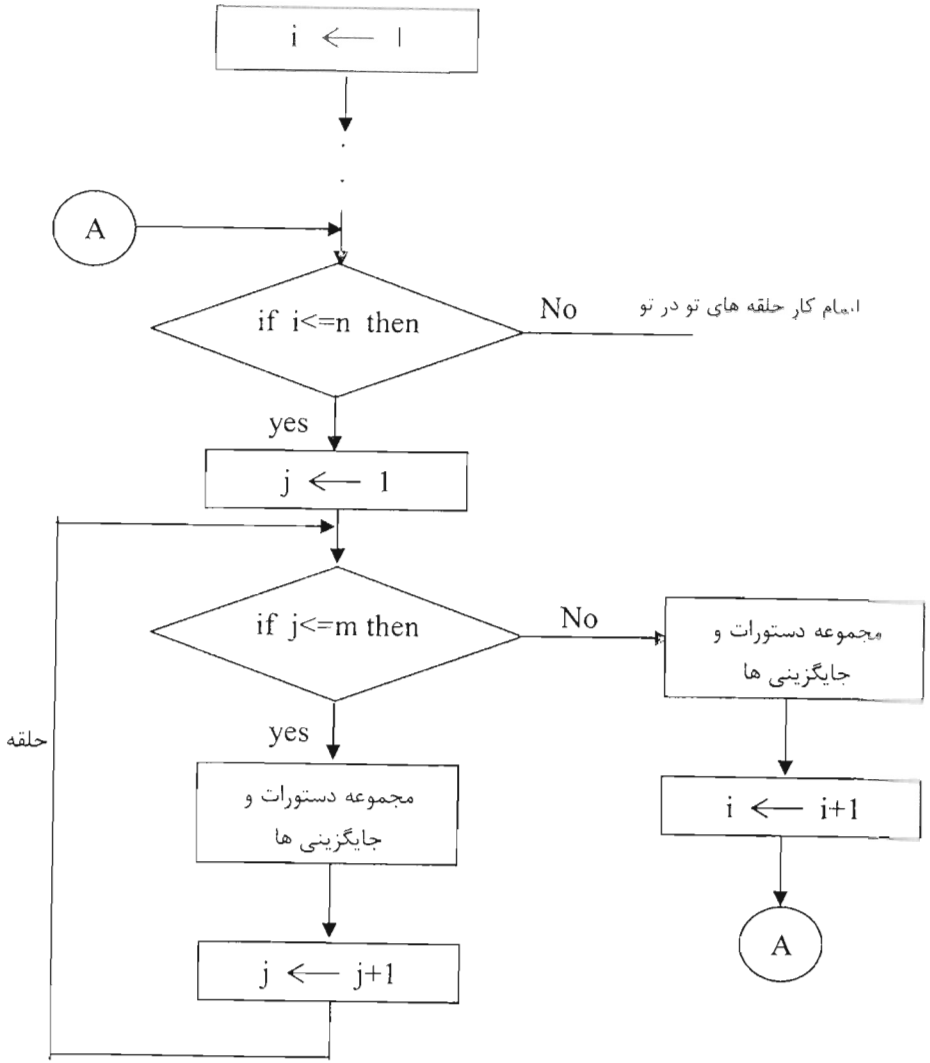
فلوچارت حلقه‌های تودرتو را می‌توان بصورت زیر نشان داد:

اندیس حلقه اول ← i

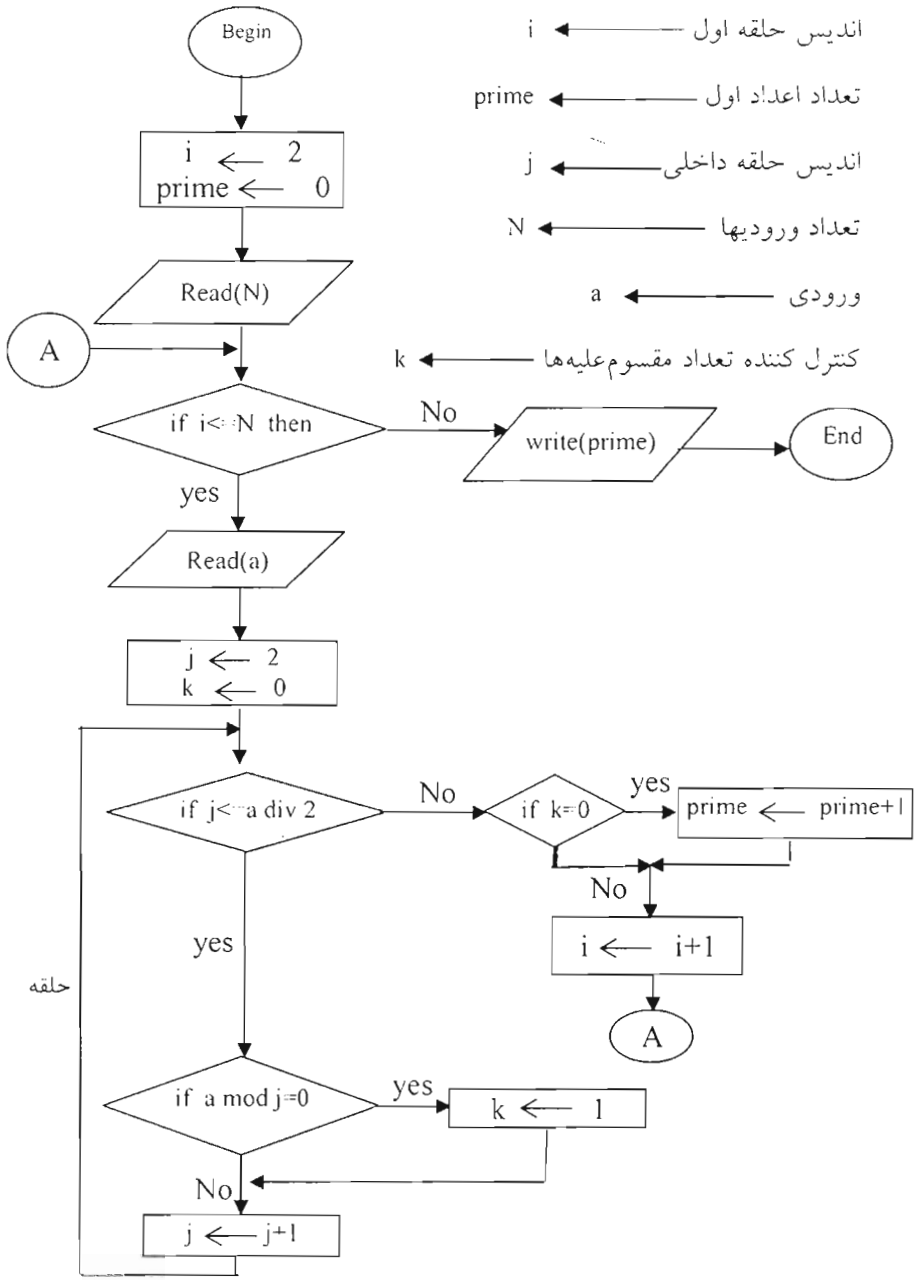
اندیس حلقه داخلی ← j

مقدار نهایی حلقه اول ← n

مقدار نهایی حلقه داخلی ← m



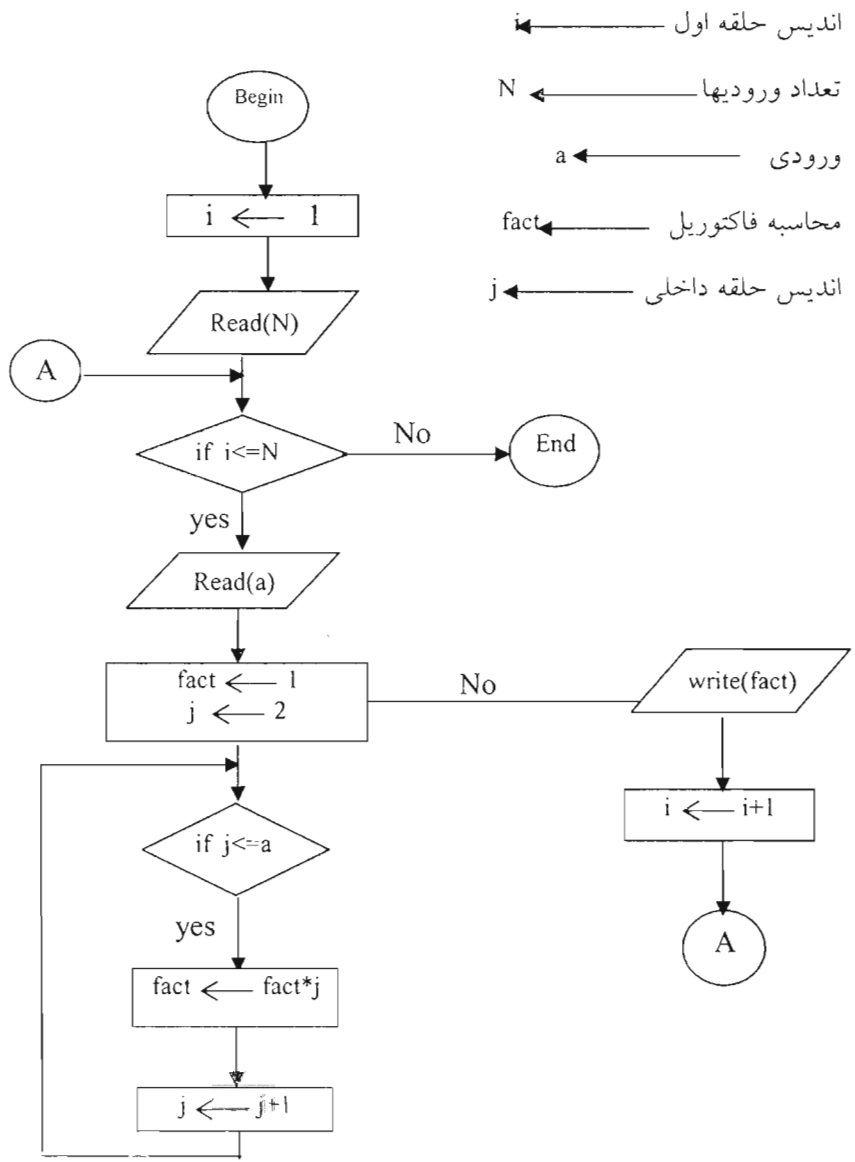
مثال ۱۷-۲ فلوچارتی رسم نمائید که N عدد از ورودی دریافت کرده تعداد اعداد اول را شمرده در خروجی چاپ نماید.



فلوچارت بالا را برای N=2 و اعداد 7 , 6 اجرا می‌کنیم:

	N a I j k prime	خروجی
1	2 7 1 0 0 0	
2	1 2 0 0	
3	1 3 0 0	
4	1 4 0 1	
5	6 2 2 1 1	
6	2 3 1 1	
7	2 4 1 1	
8	3	
		1

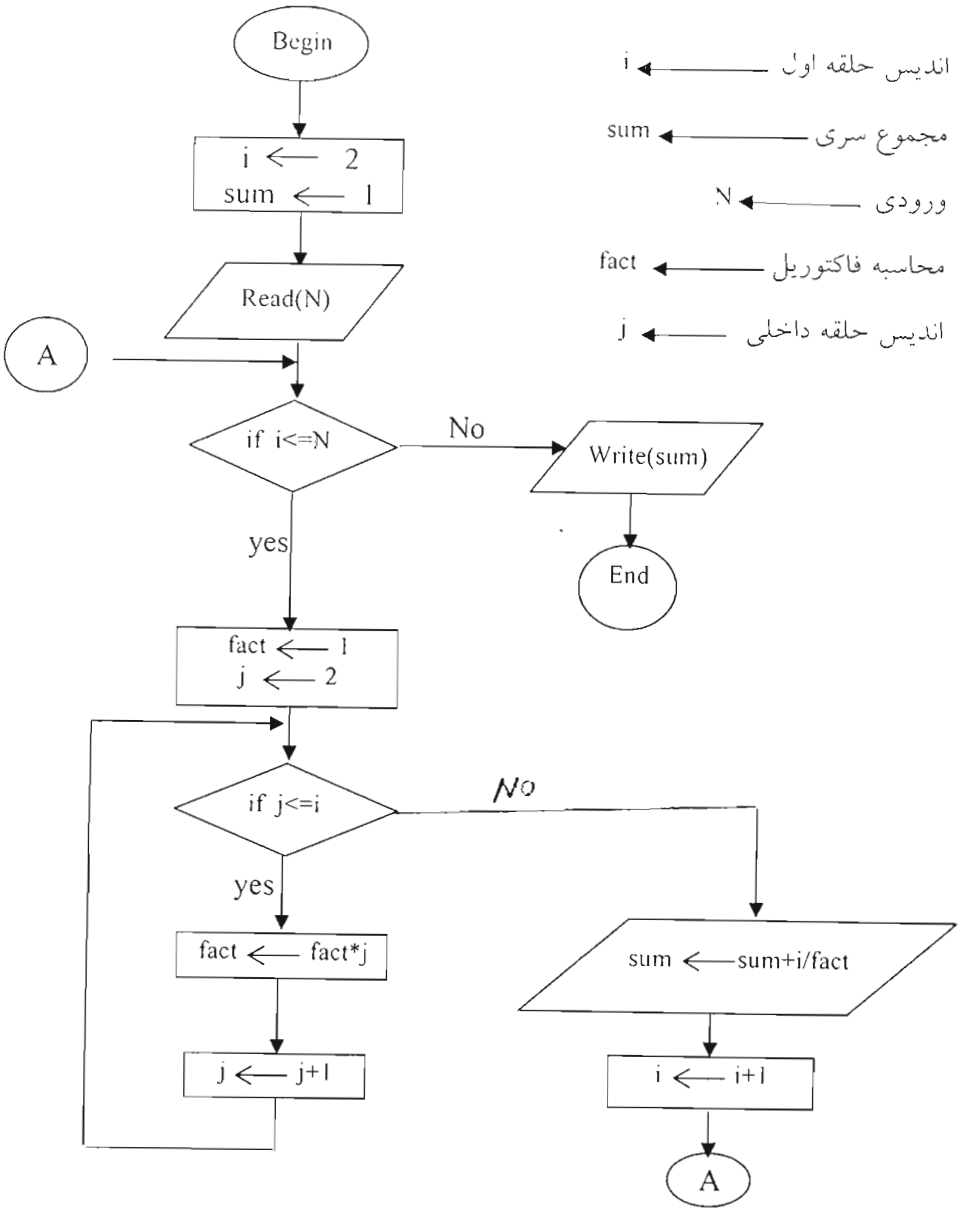
مثال ۱۸-۲ فلوچارتی رسم نمائید که N عدد از ورودی دریافت کرده، فاکتوریل هر کدام از اعداد را محاسبه و در خروجی چاپ نماید.



در حلقه تودرتو بالا، حلقه اول فقط ورودیها را کنترل می‌کند و حلقه دوم فاکتوریل هر کدام از ورودیها را محاسبه و چاپ می‌نماید.

مثال ۱۹-۲ فلوچارتی رسم نمائید که N را از ورودی دریافت کرده، مجموع سری زیر را محاسبه نماید:

$$S = 1 + \frac{2}{2!} + \frac{3}{3!} + \ldots + \frac{N}{N!}$$



سوال: آیا می‌توان فلوچارت بالا را بدون استفاده از حلقه تودرتو نوشت.

۶ـ۲ تمرینات آخر فصل

۱- فلوچارتی رسم نمائید که N عدد از ورودی دریافت کرده تعداد اعداد اول و کامل را شمرده در خروجی چاپ نماید.

۲- فلوچارتی رسم نمائید که X , N را از ورودی خوانده مقدار سری زیر را محاسبه کند:

$$S = 1 + \frac{x^2}{2!} + \frac{x^4}{4!} + \ldots + \frac{x^n}{N!}$$

(N زوج است)

۳- فلوچارتی رسم نمائید که N عدد از ورودی خوانده، تعداد اعدادی که جزء سری فیبوناچی می‌باشند را شمرده در خروجی نمایش دهد.

۴- فلوچارتی رسم نمائید که عددی را از ورودی دریافت کرده مقلوب عدد را محاسبه و در خروجی چاپ کند.

۵ـ فلوچارتی رسم کنید که تاریخ تولد شخصی را از ورودی خوانده، سن شخص را با تاریخ روز، محاسبه نموده در خروجی چاپ کند.

۶ـ سکه‌های رایج کشور که عبارتند از:

250,100,50,20,10,5,2,1 ریالی

را در نظر بگیرید.

فلوچارتی رسم نمائید که، یک اسکناس n ریالی را از ورودی دریافت کرده، با توجه به سکه‌های رایج ،آن را با حداقل تعداد سکه ها خرد کند.

۷- فلوچارتی رسم نمائید که M, N (m>n) را از ورودی دریافت کرده سری فیبوناچی بین N ,M را تولید کرده، در خروجی چاپ کند.

۸- فلوچارتی رسم نمائید که عددی را از ورودی دریافت کرده، صفرهای عدد را حذف نموده و حاصل را در خروجی چاپ کند.

مثال: 10240 ⟶ 124

۹- فلوچارتی رسم نمائید که یک جمله را از ورودی دریافت کرده، سپس تعداد کلمه‌های آن را بشمارد.

۱۰ـ فلوچارتی رسم نمائید که اطلاعات حداکثر ۱۰۰ دانشجو که عبارتند از:

تعداد درس

نمره هر درس

تعداد واحد هر درس

را از ورودی خوانده سپس تعداد دانشجویانی که معدل آنها بیشتر از ۱۷ می‌باشد را شمرده در خروجی چاپ کند.

فصل ۳

کاربرد آرایه‌ها در الگوریتمها

هدفهای کلی

- استفاده از آرایه در الگوریتمها
- استفاده از فلوچارت برای آرایه‌ها

هدفهای رفتاری

دانشجو پس از مطالعه این فصل باید بتواند:

- مفهوم آرایه را درک کند.
- برای متغیرهای زیاد، آرایه در نظر بگیرد.
- در الگوریتمها از آرایه استفاده کند.
- در فلوچارت از آرایه استفاده کند.

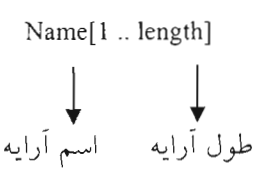
مقدمه

در مسائلی که تا حال بررسی کردیم، توسط یک متغیر تعدادی ورودی را دریافت می‌کردیم و اعمال لازم را روی ورودیها انجام می‌دادیم ولی همیشه با این مسائل روبرو نیستیم. فرض کنید بخواهیم اطلاعات ۱۰۰ کارمند را از ورودی بخوانیم و سپس آنها را مرتب کنیم، در اینصورت باید ورودیها را در جایی از حافظه ذخیره کنیم. در زبانهای برنامه‌نویسی معمولا از آرایه برای ذخیره اطلاعات در حافظه استفاده می‌کنند. در آرایه‌ها ما با توجه به تعداد ورودیها، طول آن را مشخص می‌کنیم. سپس داده‌ها را خوانده در آن قرار می‌دهیم.

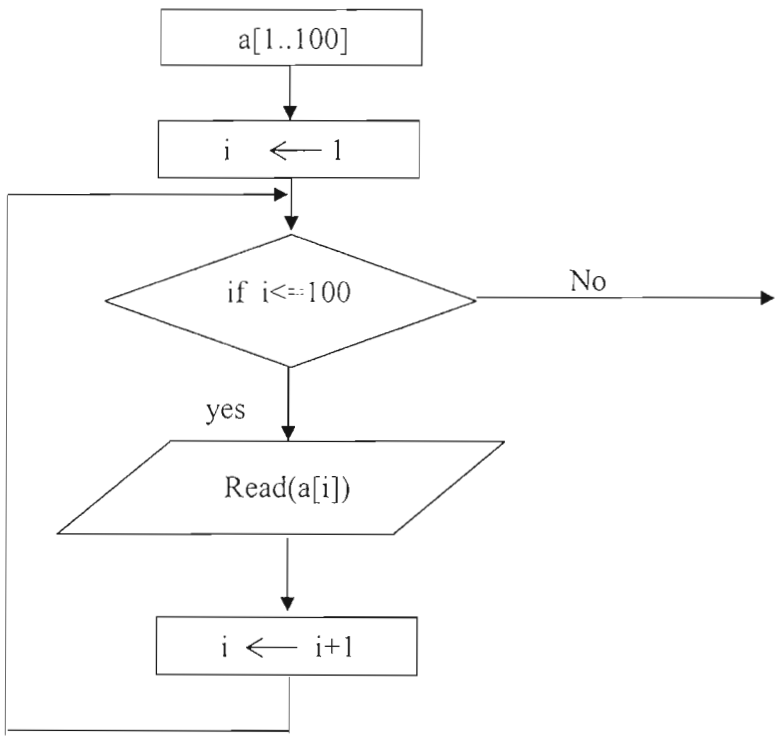
۱ـ۳ تعریف آرایه

خانه‌های پشت سر هم از حافظه، که همنوع بوده و توسط یک اسم معرفی می‌شوند، آرایه نام دارد. نحوه دسترسی به هر یک از اعضاء آرایه، از طریق اندیس آرایه امکانپذیر است . برای تعریف آرایه ابتدا طول آرایه که در حقیقت تعداد خانه‌های آن را مشخص می‌کند، معین می‌کنیم سپس نوع خانه‌ها باید معین شوند.

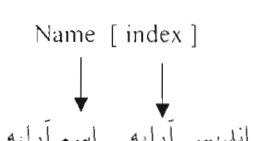
در فلوچارت‌ها آرایه‌ها را بصورت زیر نمایش می‌دهیم:



برای خواندن یک آرایه از ورودی از حلقه‌ها استفاده می‌کنیم. فلوچارت خواندن آرایه ورودی بصورت زیر می‌باشد:

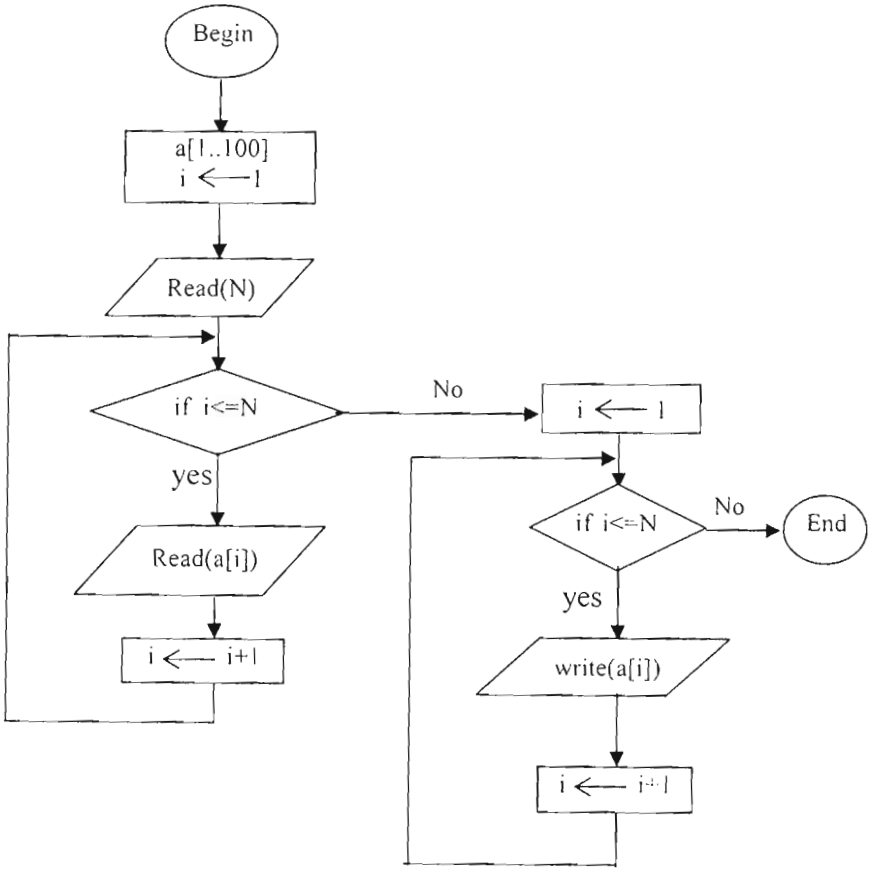


با توجه به فلوچارت بالا برای دسترسی به عنصر i ام آرایه در حالت کلی بصورت:



عمل می‌کنند.

مثال ۱-۳ فلوچارتی رسم کنید که یک آرایه حداکثر ۱۰۰ عنصری را از ورودی دریافت کرده، سپس آن را خروجی نمایش دهد.



نمونه اجرای الگوریتم بالا:

a

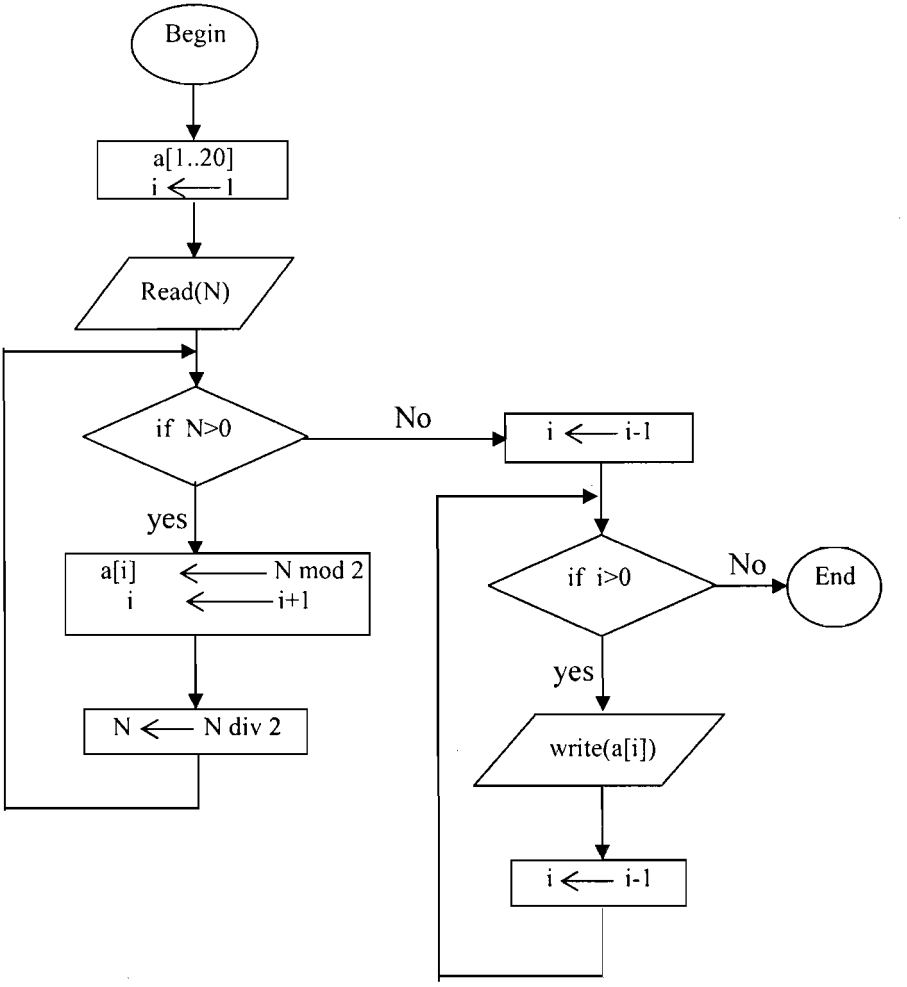
N=6

17	2	19	22	77	32	...
1	2	3	4	5	6	

عناصری که از ورودی خوانده می‌شود بترتیب در خانه‌های حافظه قرار می‌گیرند.

مثال ۲-۳ فلوچارتی رسم کنید که عددی را از ورودی دریافت کرده آن را به مبنای ۲ ببرد.

برای نوشتن فلوچارت، عدد خوانده شده را تا زمانیکه بر ۲ بخشپذیر است، تقسیم می‌کنیم و در هر مرحله باقیمانده را در آرایه قرار می‌دهیم، سپس آرایه را از آخر به اول چاپ می‌کنیم.

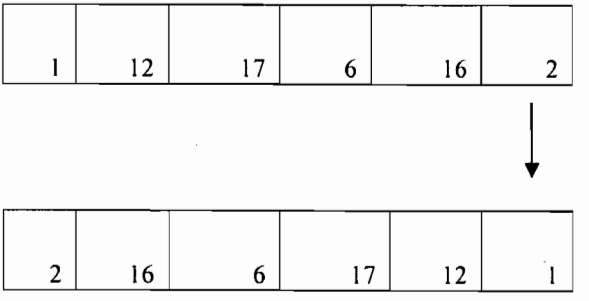


خروجی برنامه بالا با عدد 7 بصورت زیر است:

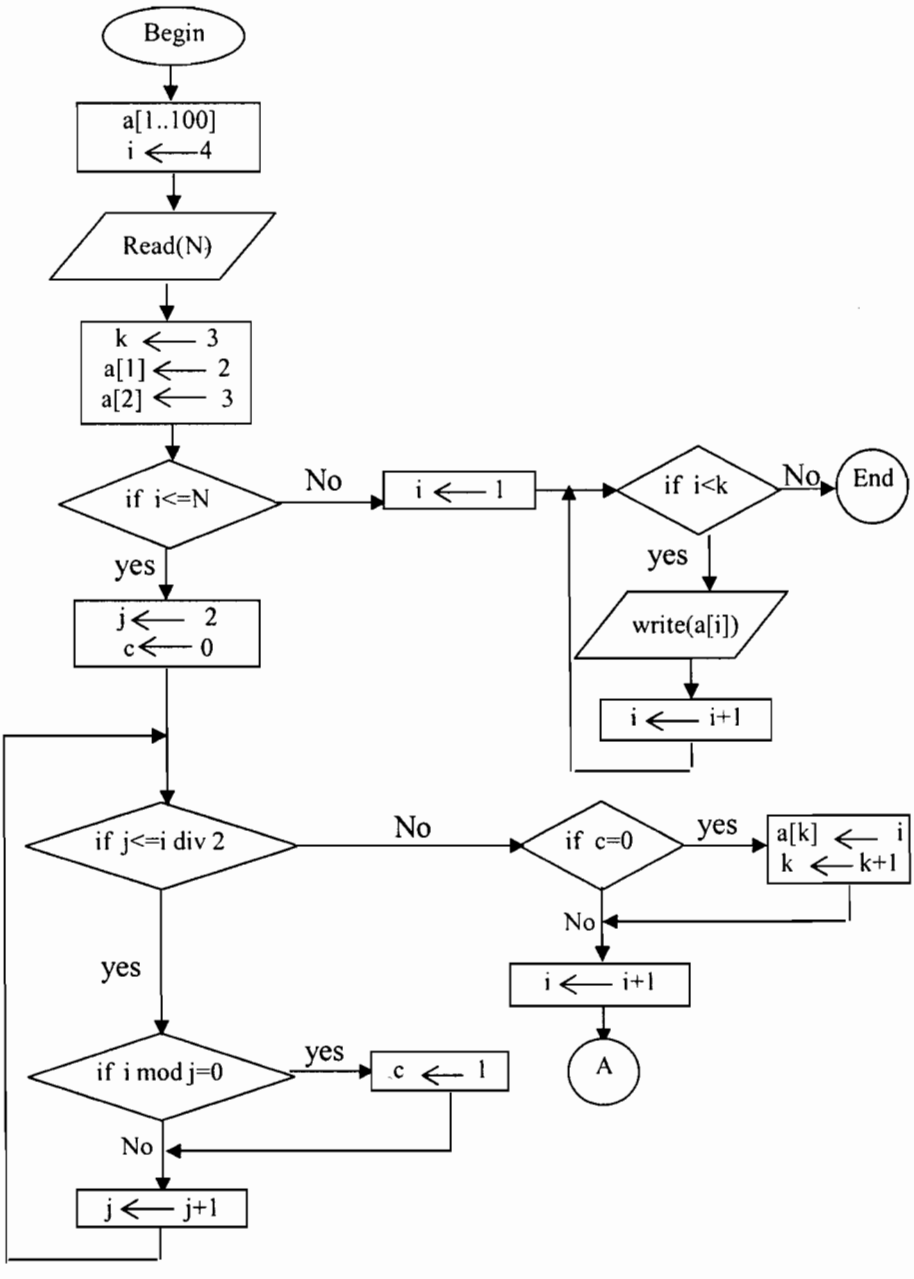
N	I	a[I]	خروجی
7	1	a[1]=1	
3	2	a[2]=1	
1	3	a[3]=1	
0	4		
			111

مثال ۳-۳ فلوچارتی رسم نمائید که یک آرایه ۱۰۰ عنصری را از ورودی دریافت کرده، سپس معکوس آن را به دست آورده، آرایه حاصل را در خروجی چاپ نماید.
فلوچارت را به عنوان تمرین ترسیم نمائید.

فلوچارت برای ۶ عنصر بصورت زیر عمل می‌کند:



مثال ۳-۴ فلوچارتی رسم نمائید که عددی از ورودی دریافت کرده سپس اعداد اول قبل از آن را تولید نموده، در یک آرایه قرار دهد.


Begin
a[1..100]
i ⟵ 4
Read(N)
k ⟵ 3
a[1] ⟵ 2
a[2] ⟵ 3
if i<=N
No
i ⟵ 1
if i<k
No
End
yes
write(a[i])
i ⟵ i+1
yes
j ⟵ 2
c ⟵ 0
if j<=i div 2
No
if c=0
yes
a[k] ⟵ i
k ⟵ k+1
No
yes
i ⟵ i+1
if i mod j=0
yes
c ⟵ 1
A
No
j ⟵ j+1

در فلوچارت بالا از حلقه‌های تودرتو برای اعداد اول قبل از N استفاده کردیم . در حلقه داخلی عدد اول تشخیص داده می‌شود و در صورت صحیح بودن شرط ، عنصر i که اول است در آرایه قرار می‌گیرد و در نهایت بعد از اتمام کار حلقه اولیه اعداد اول تولید شده، در خروجی نمایش داده می‌شوند.

۳-۲ جستجو و مرتب سازی (Search and Sort)

یکی از مسائلی که در بحث طراحی الگوریتم بسیار مهم است، بحث مرتب‌سازی و جستجو می‌باشد. منظور از جستجو اینست که یک مقداری را از یک لیست جستجو کنیم و منظور از مرتب‌سازی اینست که یک لیست مرتب از داده‌ها را تولید کنیم.

برای جستجو و مرتب‌سازی الگوریتم‌های مختلفی وجود دارد در زیر الگوریتم‌های اولیه، برای جستجو و مرتب‌سازی را بررسی می‌کنیم.

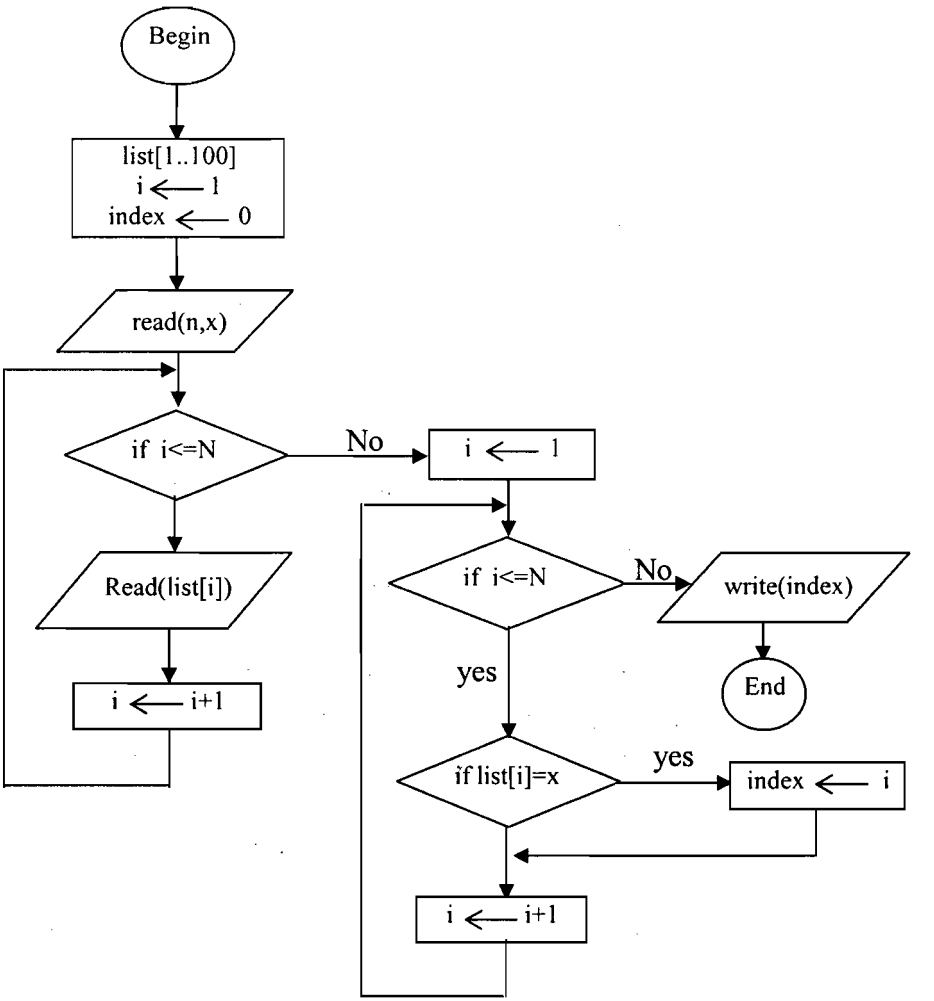
همانطور که در بالا اشاره کردیم منظور از جستجو، یافتن عنصری در یک لیست می‌باشد.

دو الگوریتم زیر غالباً برای جستجو بکار می‌روند:

- جستجوی خطی linear search
- جستجوی دودوئی binary search

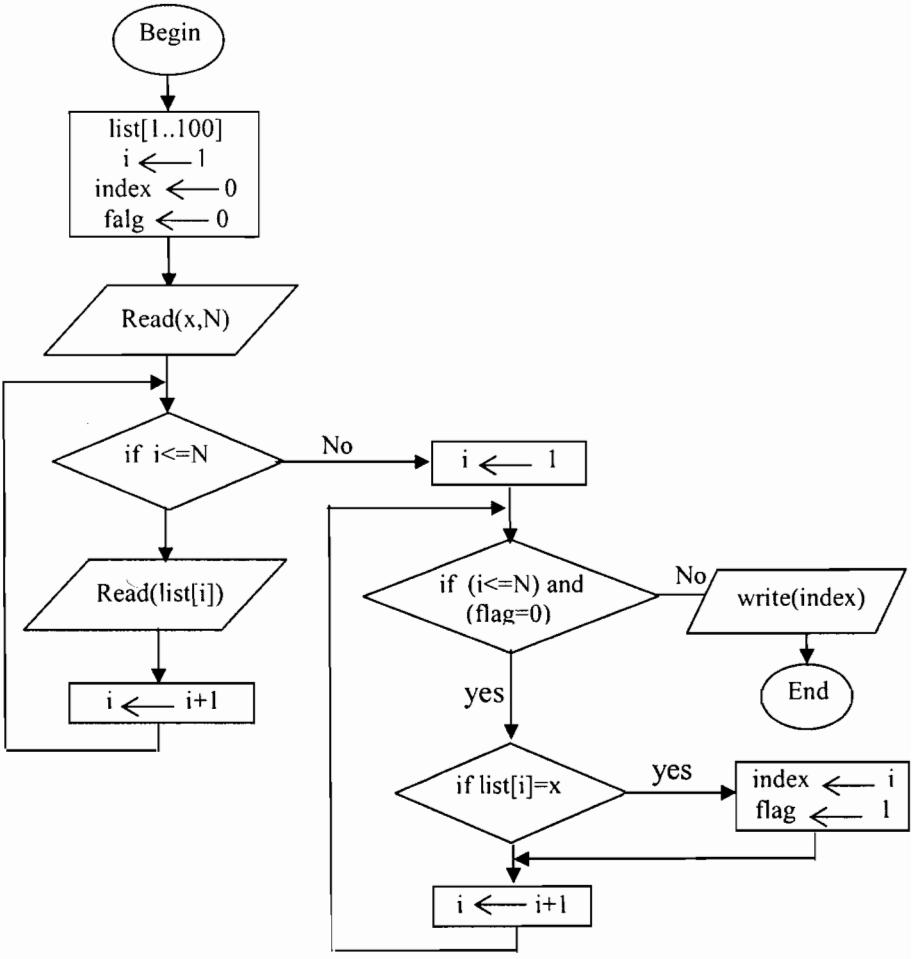
در جستجوی خطی عبارت مورد جستجو را به ترتیب با اولین، دومین و ... عنصرآرایه مقایسه می‌کنیم اگر عنصر مورد جستجو پیدا شد اندیس آن را نمایش می‌دهیم. فلوچارت جستجوی خطی بصورت زیر است:

مثال ۵-۳ فلوچارتی رسم کنید که یک آرایه حداکثر ۱۰۰ عنصری را از وردی دریافت کرده، سپس با خواندن عنصری از ورودی، عنصر جدید را در آرایه جستجو نماید.



در الگوریتم جستجوی خطی بالا عنصر مورد نظر X با عناصر آرایه list مقایسه می‌شود، اگر عنصر X در آرایه وجود داشته باشد، اندیس خانه مربوطه در خروجی چاپ می‌شود و اگر وجود نداشته باشد، مقدار صفر در خروجی چاپ می‌شود.

باید به مفهوم عمل جستجو بیشتر دقت کنیم. غالباً منظور از جستجو پیدا کردن محل اولین وقوع عنصر مورد نظر می‌باشد، لذا الگوریتم بالا را می‌توان بصورت زیر تصحیح کرد:



در فلوچارت بالا به محض پیدا کردن عنصر مورد جستجو، مقدار متغیر flag برابر یک قرار داده می‌شود و اینکار باعث پایان عمل جستجو می‌شود. زیرا شرط حلقه تا زمانی برقرار است که دو شرط بالا برقرار باشد.

در جستجوی دودوئی، لیست مورد جستجو مرتب می‌باشد. لذا برای جستجو اعمال زیر انجام می‌شود:

۱- عنصر X با عنصر وسط آرایه که اندیس آن برابر

middle ←——— (low+high)/2

مقایسه می‌شود.

۲ـ اگر x از عنصر وسط کوچکتر باشد، عنصر مورد نظر احتمالاً در قسمت بالای لیست قرار دارد. لذا آرایه با اندیس، جدید در نظر گرفته می‌شود و قسمت پایین لیست از فضای جستجو حذف می‌شود.

۳ـ اگر x از عنصر وسط آرایه بزرگتر باشد قسمت بالای لیست حذف می‌شود و فضای جستجو، قسمت پایین آرایه خواهد بود.

۴- اگر x برابر عنصر وسط باشد عمل جستجو خاتمه می‌پذیرد.

عملیات 2 , 3 را تا زمانیکه، عنصر مورد نظر پیدا نشده، تکرار می‌کنیم.(در صورت وجود عنصر در لیست)

فلوچارت جستجوی دودوئی را بعنوان تمرین به خوانده واگذار می‌کنیم.

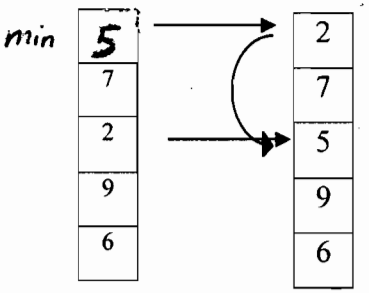
مرتب‌سازی بحث بعدی این فصل می‌باشد. برای مرتب کردن داده‌ها نیز الگوریتمهای مختلفی وجود دارد، که هر کدام مزایا و معایب خاص خود را دارد. در اینجا به یک الگوریتم از الگوریتمهای مرتب سازی اشاره می‌کنیم و بحث مفصل در این مورد را به فصلهای بعد واگذار می‌کنیم.

لیست زیر را در نظر بگیرید:

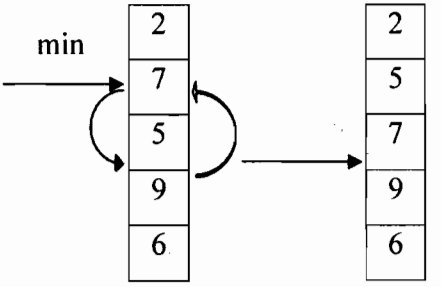
5	7	2	9	6

برای مرتب کردن لیست بالا در هر مرحله کوچکترین عنصر را پیدا کرده در خانه‌های بالای لیست قرار می‌دهیم. مراحل این الگوریتم بصورت زیر است:

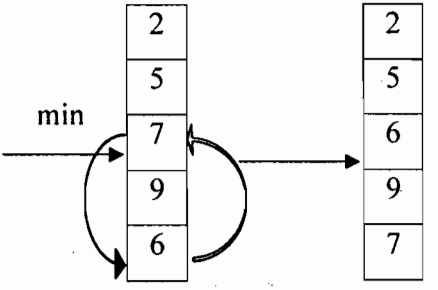
مرحله اول:



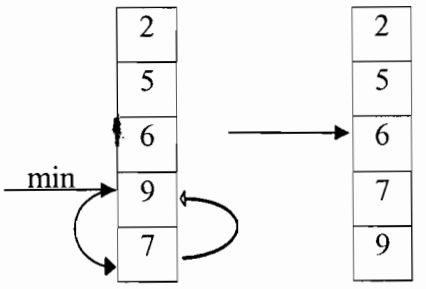
مرحله دوم :



مرحله سوم :



مرحله چهارم :



همانطور که ملاحظه می‌کنید لیست حاصل بصورت صعودی مرتب می‌شود.

نکته: برای مرتب سازی بصورت نزولی نیز الگوریتم بالا با تغییر پیدا کردن بیشترین مقدار در هر مرحله به جای کمترین مقدار عمل مرتب سازی نزولی را انجام می‌دهد.

الگوریتم بالا به الگوریتم مرتب‌سازی انتخابی (selection sort) معروف است . فلوچارت الگوریتم بالا را بعنوان تمرین به خواننده واگذار می‌کنیم.

تمرین

۱- فلوچارتی رسم نمائید که عددی از ورودی دریافت کرده، اعداد کامل قبل از خود را تولید و در یک آرایه قرار دهد.

۲ــ فلوچارتی رسم نمائید که یک آرایه حداکثر ۱۰۰ عنصری از ورودی دریافت کرده، عناصری از آن که اول هستند را با صفر کردن حذف نماید.

۳ــ فلوچارتی رسم نمائید که یک عدد حداکثر ۲۰ رقمی را توسط آرایه‌ای از ورودی دریافت نماید. سپس یک عدد تک رقمی را از ورودی خوانده در عدد ۲۰ رقمی ضرب نموده، حاصل را در خروجی چاپ نماید.

۴- فلوچارتی رسم نمائید که دو عدد حداکثر ۲۰ رقمی را از ورودی دریافت کرده در هم ضرب نماید.

۵- فلوچارتی رسم نمائید که دو آرایه مرتب حداکثر ۱۰۰ عنصر را از ورودی دریافت کرده، آنها را در آرایه سومی طوری قرار دهد، که آرایه سوم مرتب باشد.

۶- فلوچارتی رسم نمائید که یک عدد از ورودی دریافت کرده صفرهای عدد را حذف کرده، عدد حاصل را در خروجی چاپ نماید.

۷- فلوچارتی رسم نمائید که یک عدد اعشاری را از ورودی دریافت کرده، مقلوب عدد را محاسبه در خروجی چاپ نماید.

۸- فلوچارتی رسم نمائید که یک عدد از ورودی دریافت کرده آن را به عامل‌های اول تجزیه نماید.

فصل ۴

ساختار برنامه در زبان پاسکال

هدفهای کلی

- شناخت اجزای تشکیل دهنده یک برنامه
- شناخت ساختار یک برنامه در زبان پاسکال
- بررسی دستگاههای خروجی و دستورات لازم در زبان پاسکال برای تولید خروجی

هدفهای رفتاری

دانشجو پس از مطالعه این فصل باید بتواند:

- اجزاء لازم برای نوشتن برنامه در زبان پاسکال را بداند.
- یک شناسه صحیح در زبان پاسکال را تعریف کند.
- ساختار یک برنامه در زبان پاسکال و اعلانهای مربوط به برنامه را تعریف نماید.
- یک برنامه ساده به زبان پاسکال که فقط خاصیت خروجی دارد، بنویسد.

مقدمه

هر زبان برنامه‌نویسی ساختار مشخصی دارد که می‌بایست برنامه خود را در چهارچوب ساختار آن زبان بنویسیم ساختار زبان را می‌توان بصورت زیر توصیف کرد.

ساختار زبان عبارت است از نحوه بکار گیری دستورات، عبارات، شروط و ... برای رسیدن به هدف خاص. زبان پاسکال نیز از این قاعده مستثنی نیست و دارای ساختار مشخصی می‌باشد که در زیر بررسی می‌کنیم.

۱-٤- اجزای تشکیل‌دهنده یک برنامه

در کل یک زبان برنامه‌نویسی از یک سری علائم، قواعد، دستورالعمل ها و ... تشکیل می‌شود. لذا یک برنامه که در یک محیط برنامه نویسی نوشته می شود، باید این علائم، قواعد و دستورالعمل ها را در نظر داشته باشد.

یک برنامه به زبان پاسکال نیز باید علائم، قواعد و دستورالعمل های زبان برنامه‌نویسی پاسکال را رعایت کند. در ذیل به بررسی زبان برنامه‌نویسی پاسکال می‌پردازیم.

۱-۱-٤- کلمات ذخیره‌شده (Reserved Words)

کلمات ذخیره شده، کلماتی هستند که مترجم زبان آنها را می‌شناسد و معنای خاصی برای زبان دارند. مترجم زبان به محض مشاهده این کلمات اعمال خاصی را انجام می‌دهد. هر زبان دارای تعداد مشخصی کلمات ذخیره شده می‌باشد و این تعداد قابل افزایش توسط برنامه‌نویس نیست. برنامه‌نویس این کلمات را در حین نوشتن برنامه بکار می‌برد.

لیست کلمات ذخیره شده در پاسکال عبارتند از:

shr	mod	exports	and
asm	file	nil	string
array	for	not	then
begin	function	object	to
case	goto	of	type
concat	if	or	unit
constructor	implementation	packed	until
destructor	in	procedure	uses
div	inherited	program	var
do	inline	record	with
downto	interface	repeat	while
xor	else	label	set
end	library	shl	

توجه: نگران یادگیری کلمات ذخیره شده نباشید، به مرور آنها را یاد خواهید گرفت.

در زبان پاسکال فقط ۵۱ کلمه ذخیره شده وجود دارد که این تعداد در مقایسه با ۱۵۹ کلمه ذخیره شده که در زبان بیسک وجود دارد، قابل توجه است.

۲-۱-٤- شناسه‌ها (identifier)

شناسه که آن را با id نشان خواهیم داد در پاسکال برای نامگذاری ثابتها، تایپ‌ها، پروسیجروها، توابع، میدان‌های یک رکورد، برنامه و همچنین یونیت مورد استفاده قرار می‌گیرد. دو نوع id وجود دارد که عبارتند از:

الف) id های استاندارد: این نوع id ها از قبل در زبان پاسکال تعریف شده‌اند و در برنامه‌ها، معنای خاصی دارند مانند: write ,read , ...

ب) id های غیراستاندارد: این نوع id ها بوسیله کاربر بطور مجزا تعریف می‌شوند و اصطلاحاً به آنها userdefined گفته می‌شود.

id ها می‌توانند طولی از ۱ تا ٦۳ کاراکتر داشته باشند و اگر طول آنها بیشتر از ٦۳ کاراکتر باشد فقط ٦۳ کاراکتر اول در نظر گرفته می‌شود.

اسامی شناسه های غیر استاندارد از قواعد زیر پیروی می‌کنند:

۱. حروف a-z و A-Z
۲. ارقام ۹-۰
۳. کاراکتر اول نباید رقم باشد

٤. نمی توان از کلمات ذخیره شده استفاده کرد.
٥. از جای خالی (space) بین کاراکترها نمی‌توان استفاده کرد.
٦. از علامت under line (_) می‌توان بین حروف استفاده کرد.
بقیه کاراکترها در شناسه ها قابل استفاده نیستند.

مثال ۱-٤ شناسه‌های زیر را در نظر گرفته، سپس معتبر بودن و غیرمعتبر بودن آنها را تشخیص دهید.

شناسه صحیح	شناسه غلط	
Num۱	۱num	کاراکتر اول رقم
- num	End	کلمه ذخیره شده
Num_۱۲	t (۱)	استفاده از پرانتز
Number	num ۱	جای خالی
No	ch?	کاراکتر غیر مجاز

۲-٤- ساختار برنامه در زبان پاسکال

اجزاء اصلی یک برنامه به زبان پاسکال بصورت زیر می‌باشد:

الف) عنوان برنامه (program heading)
ب) قسمت تعاریف برنامه
ج) قسمت دستور العمل‌ها

الف) عنوان برنامه

عنوان برنامه که شامل نام برنامه و فرمانهای کامپایلر می‌باشد، اولین بخش از ساختار یک برنامه را تشکیل می‌دهد و محل قرار گرفتن نام برنامه به همراه پارامترهای ورودی و خروجی و همچنین فرمانهای کامپایلر که کنترل عمل کامپایل شدن برنامه را به عهده دارند، می‌باشد.

تذکر: عنوان برنامه در توربو پاسکال اختیاری می‌باشد و وجود آنها باعث هر چه روشن شدن وضعیت برنامه و شرایط ترجمه آن به زبان ماشین می‌باشد ولی در پاسکال استاندارد عنوان برنامه حتماً باید لحاظ شود.

نام برنامه بعد از کلمه ذخیره شده program ظاهر می‌شود و هدف اسم گذاری برنامه‌ای هست که می خواهیم بنویسیم. اسم برنامه یک شناسنامه محسوب می‌شود لذا از قواعد اسم گذاری شناسه ها تبعیت می کند. بعد از اسم برنامه، می‌توان پارامترهای عملیاتی برنامه را لحاظ کرد (مشخص کردن پارامترهای عملیاتی اختیاری است).

مثال ۲-۴

```
program  test;
program  test _one;
```

پارامترهای عملیاتی که بعد از اسم برنامه ظاهر می‌شوند، وضعیت ورودی و خروجی بودن را مشخص می‌کنند. اگر از کلمه ورودی استفاده گردد به معنای این است که برنامه فقط عمل ورود داده‌ها را انجام می‌دهد و اگر از کلمه خروجی استفاده گردد به معنای این است که برنامه عمل چاپ داده‌ها یا اطلاعات را انجام می‌دهد.

تذکر: غالباً اگر پارامترهای عملیاتی لحاظ شوند، هم ورودی و هم خروجی بودن در نظر گرفته می‌شود.

مثال ۳-۴

```
Program      test _Two (Input ,output) ;
Program      scan (Input) ;
Program      Print (out put) ;
```

بخش دومی که به عنوان برنامه باید در نظر گرفت عبارتست از دستورات کامپایلر که به مرور در مباحث بعدی، مورد بررسی قرار خواهند گرفت.

ب: قسمت تعاریف برنامه

این بخش از برنامه خود شامل ۴ قسمت زیر می‌تواند باشد:

۱. اعلان ثابت‌ها Constant Declaration
۲. اعلان انواع Type Declaration
۳. اعلان متغیرها Variable Declaration
۴. اعلان برچسب ها Label Declaration

این قسمت از برنامه به قسمت اعلان ها نیز مشهور است، چرا که در آن به اعلان نام‌ها و انواع مورد استفاده در قسمت‌های مختلف برنامه پرداخته می‌شود.

ج: قسمت دستورالعمل‌ها

در این قسمت از برنامه با استفاده از دستورالعمل‌ها و قواعد زبان پاسکال، مسئله مورد نظر به زبان پاسکال پیاده‌سازی می‌شود.

حل مسئله که به الگوریتم مسئله معروف است، توسط برنامه‌نویس ارائه می‌شود و با استفاده از دستورالعمل‌ها و قسمت‌های مختلف زبان پیاده‌سازی می‌شود. دستورات برنامه در داخل بلوکی که با کلمه ذخیره شده Begin شروع و به کلمه ذخیره شده End ختم می‌شود، قرار می‌گیرند. در ضمن End برنامه همواره به نقطه (.) ختم می‌شود.

ساختار کلی یک برنامه در زبان پاسکال بصورت زیر می‌باشد:

```
Program    اسم برنامه ( پارامترها );
تعاریف
.
.
Begin
دستور 1 ;
دستور 2 ;
.
.
End.
```

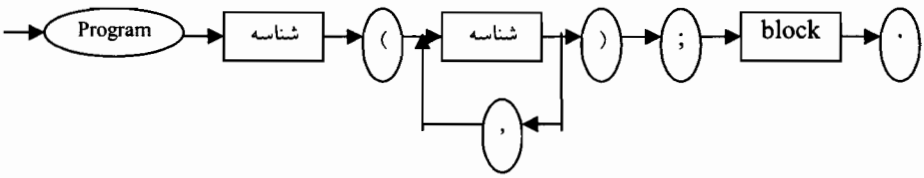
نکته: استفاده از کلمه کلیدی Program و اسم برنامه در توربو پاسکال ضروری نیست.

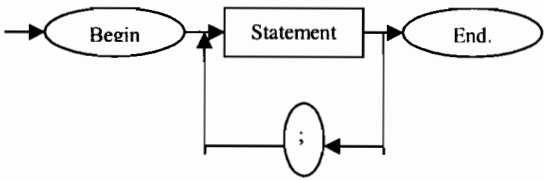
درحالت کلی می‌توان شکل یک برنامه در زبان پاسکال را بصورت زیر بیان کرد:

۱. استفاده از کلمه ذخیره شده Program و اسم برنامه (که می‌تواند بکار برده نشود)
۲. قسمت تعاریف شناسه‌ها
۳. بلوک اصلی برنامه که با Begin شروع و به End همراه (.) ختم می‌شود.
۴. هر دستور در پاسکال به (;) ختم می‌شود.

توجه داشته باشید که کلمات ذخیره شده به تنهایی به (;) ختم نمی‌شوند.

نمودارهای یک برنامه و بلوک بترتیب به صور ت زیر می‌باشد:





شکل ۱-۴ دیاگرامهای برنامه و بلوک

۳-۴- خروجی (Output)

محاسبه هر چه باشد، وقتی دستور اجرای آن را به کامپیوتر می‌دهیم، کامپیوتر باید نتایج را به ما بدهد. بنابراین هر کامپیوتر باید یک یا چند دستگاه خروجی داشته باشد تا از طریق آن بتواند نتیجه عملیاتی را که به کامپیوتر داده‌ایم، به ما نشان دهد. این دستگاهها را معمولاً دستگاههای خروجی (Output Device) می‌گویند.

مشهورترین دستگاههای خروجی عبارتنداز:

- صفحه نمایش Monitor
- چاپگر Printer
- ترمینال Terminal
- رسام Plotter

هر کدام از دستگاههای بالا به نحوی نتایج برنامه را برای کاربر به نمایش می‌گذارند.

در نتیجه یک زبان برنامه‌نویسی باید امکاناتی داشته باشد تا کامپیوتر را در ارائه نتایج محاسبات هدایت کند.

در زبان پاسکال دستورات زیادی برای این کار وجود دارد که یکی از آنها دستور Writeln می‌باشد. نحوه بکارگیری این دستور بصورت زیر می‌باشد:

```
Writeln ( ' statments ' )
```

عبارتی که قرار است چاپ شود بین علامت نقل قول قرار می‌گیرد و همه داخل پارانتز محصور می‌شوند.

مثال ۴-۴ برنامه زیر را در نظر بگیرید:

```
Program Print ( output ) ;
Begin
   Writeln ( 'Pascal  language' ) ;
   Writeln ( 'Hello' ) ;
End.
```

خروجی برنامه بالا بصورت زیر می‌باشد:

```
Pascal  language
Hello
```

سوالی که مطرح می شود این است که اگر بخواهیم یک کوتیشن در یک سطر چاپ کنیم، چه باید بکنیم؟ مثلاً اگر بخواهیم دستور زیر را بنویسیم:

```
Writeln ( ' program ' s  Report ' ) ;
```

چه اتفاقی رخ می دهد.

وقتی کامپایلر، علامت نقل قول بعد از Program را می خواند، طبیعتاً تصور می‌کند که به انتهای عبارت رسیده است لذا از عبارت شما اشکال می‌گیرد. چون بقیه عبارت را نمی تواند ترجمه کند. کاری که می‌توانیم انجام دهیم این است که از دو علامت نقل قول به صورت زیر استفاده کنیم:

```
Writeln ( ' program' ' s  Report ' ) ;
```

وقتی کامپایلر دومین علامت نقل قول را می خواند اینگونه استنباط می‌کند که عبارت بعد از Program همچنان ادامه دارد لذا عبارت مورد نظر را در خروجی چاپ می‌کند.

۴-۴- تمرینات

۱- کدام یک از شناسه‌های زیر در پاسکال مجاز هستند:

A1 , number 1 , n-m , 7X , A5.4 , *B2 , number_1

چه خطاهای کامپایلری در برنامه‌های زیر وجود دارد، مشخص کرده سپس آنها را تصحیح کنید:

الف)
```
Program    test_1;
Begin ;
        Writeln ( ' program' );
        Writeln ( ' Test ' ) ;
End.
```

ب)
```
Program    test 2 ;
Begin
        Writeln ( ' out put ' ) ,
        Writeln ( ' program Two ' ) ;
End.
```

ج)
```
Program      Begin( out put ) ;
Begin
        Writeln ( ' output ' ) ;
        Writeln (  program  three ) ;
End ;
```

۲- خروجی دستورات زیر را بدست آورید ؟

الف)
```
Writeln ( ' hello ' ) ;
Writeln ( ' world  ' ) ;
```

ب)
```
Writeln ( ''' *'' ) ' ) ;
Writeln ( ' ***      *' ) ;
Writeln ( '''    **  ''' ) ;
```

ج)
```
Writeln ( '12 + 7 = 19 ' ) ;
Writeln ( ' 47 + 3 ' ) ;
Writeln ( ' = 50 ' ) ;
Writeln ( ' 50 / 2 ' ) ;
```

۴-۵- تمرینات برنامه‌نویسی

۱- برنامه‌ای بنویسید که در سطرهای جداگانه اسم، فامیلی و شماره دانشجویی خود را وسط صفحه نمایش چاپ کند.

۲- برنامه‌ای بنویسید که با استفاده از علامت * یک مستطیل را در صفحه نمایش چاپ کند.

۳- برنامه‌ای بنویسید که عبارت Pascal's Book را در خروجی چاپ نماید.

۴- برنامه‌ای بنویسید که کاراکتر T را به صورت بزرگ در صفحه نمایش چاپ کند.

فصل ۵

انواع عملگرها و داده‌ها در زبان پاسکال

هدفهای کلی

- معرفی انواع عملگرها در زبان پاسکال
- شناخت انواع داده‌ها
- بررسی اولویت عملگرها
- معرفی دستورات جایگزینی در پاسکال

هدفهای رفتاری

دانشجو پس از مطالعه این فصل باید بتواند:

- انواع عملگرها در زبان پاسکال را بکار ببرد.
- انواع داده‌ها برای یک برنامه را تعریف کند.
- اولویت عملگرها در یک عبارت را تشخیص دهد.
- یک برنامه ساده با عملیات معمولی را بنویسد.

مقدمه

هر مسئله‌ای که مطرح می‌شود تا توسط کامپیوتر اجرا شود، نیاز به ورودی دارد و برنامه با استفاده از ورودیها، خروجی‌های لازم را تولید نموده و نمایش می‌دهد. ورودی‌های هر مسئله ممکن است با مسئله دیگر از نظر نوع، تعداد و ... متفاوت باشد.

در کل ورودیهای مسئله باید دارای نوع باشند. مثلاً یک عدد می‌تواند صحیح یا اعشاری باشد لذا برنامه قبل از نوشته شدن نیاز به تعیین نوع داده‌های ورودی خود می‌باشد. در طی اجرای برنامه با توجه به نوع مسئله اعمالی روی این داده ها انجام می‌شود که این اعمال توسط عملگرهای (Operator) مجاز در یک زبان انجام می‌شود. در این فصل انواع داده‌ها، عملگرهای مجاز در زبان پاسکال را بررسی می‌کنیم.

۱-۵- عملگرها

عملگرها نمادهایی هستند که برای انجام اعمال خاصی مورد استفاده قرار می‌گیرند.

عملگرها برای انجام اعمال خاصی روی عملوندها (Operands) بکار می‌روند. با توجه به نوع عملگر ممکن است یک یا دو عملوند وجود داشته باشد. عملگرها در زبان پاسکال از تنوع زیادی برخوردارند.

در پاسکال چهار دسته عملگر به نام‌های: محاسباتی، رابطه ای، منطقی و عملگرهای بیتی وجود دارند.

۱-۱-۵- عملگرهای محاسباتی

عملگرهای محاسباتی که در پاسکال مورد استفاده قرار می‌گیرند، در جدول ۱-۵ فهرست شده‌اند.

جدول ۱-۵ عملگرهای محاسباتی

ردیف	عملگر	نام	مثال
۱	+	جمع	x + y
۳	-	تفریق و منهای یکانی	x - y , -x
۳	*	ضرب	x * y
۴	/	تقسیم	x / y
۵	div	تقسیم	a div b
۶	mod	باقیمانده تقسیم	a mod b

عملگرهای ۱ تا ۴، عملگرهای آشنای ریاضی هستند. عملگر ۵، عملگر تقسیم بوده و شامل دو عملوند می‌باشد که برای تقسیم عدد صحیح بکار می‌رود. عملوندهای از ۱ تا ۴ می‌توانند هم صحیح باشند و هم اعشاری، ولی عملوندها در div و mod فقط می‌توانند صحیح باشند. (a و b صحیح و x و y حقیقی هستند)

عملگر شماره ۶ عملگر باقیمانده است و باقیمانده تقسیم دو عدد صحیح را محاسبه می‌نماید.

به مثال‌های زیر توجه کنید:

```
6   div 3 = 3        6   mod 3 = 0
8   div 3 = 1        6   mod 4 = 3
3   div 3 = 1        7   mod -3 = 1
-6  div 3 = -3       -7  mod -3 = -1
```

اولویت عملگرهای محاسباتی در جدول ۲-۵ نمایش داده شده است.

جدول ۲-۵ تقدم عملگرهای محاسباتی

بالاترین تقدم - (تفریق یکانی)
* / div mod
پایین‌ترین تقدم - +

عملگرهایی که در یک سطر ظاهر شده اند دارای تقدم مکانی نسبت به یکدیگر هستند و هر کدام از عملگرها که اول ظاهر شوند، زودتر اجرا خواهد شد.(از چپ)

۲-۱-۵- عملگرهای رابطه‌ای

عملگرهای رابطه‌ای برای تشخیص ارتباط بین عملوندها یا مقایسه آنها مورد استفاده قرار می‌گیرند (جدول ۳-۵).

جدول ۳-۵ عملگرهای رابطه‌ای

مثال	نام	عملگر
x > y	بزرگتر	>
x >=y	بزرگتر مساوی	>=
x < y	کوچکتر	<
x <= y	کوچکتر مساوی	<=
x = y	مساوی بودن	=
x <> y	نامساوی	<>

این عملگرها معمولاً در شرطها کاربرد دارند و برای مقایسه مقادیر متغیرها یا شناسه ها بکار برده می‌شوند.

۳-۱-۵- عملگرهای منطقی (یا عملگرهای بولی)

عملگرهای منطقی (جدول ۴-۵) بر روی عملوندهای منطقی عمل می‌کنند. عملوندهای منطقی دارای دو ارزش درستی (True) یا نادرستی (False) می‌باشند (جدول ۴-۵).

جدول ۴-۵ عملگرهای منطقی

مثال	نام	عملگر
a>y and y<x	و	And
x>y or y<x	یا	OR
Not (x)	نقیض	not

به مثال‌های زیر توجه کنید:

```
( 3 > 5 )   and   ( 7 < 8 ) = F
( 8 > 5 )   OR    ( 6 < 10 ) = T
( 3 > 5 )   OR    ( 6 < 10 ) = T
not ( 5 > 3 ) = F
```

(۳ > ۵) and (۷ < ۸) = F
(۸ > ۵) OR (٦ < ۱۰) = T
(۳ > ۵) OR (٦ < ۱۰) = T
not (۵ > ۳) = F

جدول (۵-۵) تقدم عملگرهای رابطه‌ای و منطقی را نمایش می‌دهد:

جدول ۵-۵ تقدم عملگرهای منطقی رابطه ای

not	بالاترین تقدم
> > = < < = = < >	
and	
or	پایین ترین تقدم

٤-۱-۵- عملگرهای بیتی (bitwise operator)

توسط این عملگرها می‌توان تا حدودی کارهایی که در اسمبلی قابل انجام هستند، را انجام داد. یکی از اعمالی که در زبان اسمبلی براحتی انجام‌پذیر است، انجام اعمالی بر روی بیت‌های یک بایت یا یک کلمه (Word) از حافظه است. در زبان پاسکال برای انجام این منظور از عملگرهایی استفاده می‌شود که به عملگرهای بیتی معروفند(جدول ٦-۵).

جدول ٦-۵ عملگرهای بیتی

عملگر	نوع عمل
AND	و
OR	یا
XOR	یا انحصاری
NOT	نقیض
Shl	انتقال به سمت چپ
Shr	انتقال به سمت راست

این عملگرها انجام اعمال تست، مقدار دادن و یا انتقال (shift) بیت‌ها را در یک بایت یا کلمه حافظه امکانپذیر می‌نمایند. عملگرهای بیتی فقط بر روی متغیرهایی از نوع صحیح و کاراکتری (بعداً انواع داده ها بحث خواهد شد) عمل می‌کنند.

$67 = (01000011)_2$
$253 = (11111101)_2$
01000011
and
11111101

01000001

همانطور که مشاهده می کنید در صورتی که مقدار دو بیت 1 باشد، نتیجه and 1 خواهد بود در بقیه موارد مقدار 0 حاصل می‌شود.

مثال ۲-۵ مقدار 153 or 2 را محاسبه کنید.

10011001
or
00000010

10011011

مشاهده می کنید در صورتی که ارزش دو بیت صفر باشد نتیجه or صفر و در بقیه موارد مقدار 1 حاصل می‌شود.

مثال ۳-۵ 45 Xor 7 را محاسبه نمائید.

00101101
Xor
00000111

00101010

توجه کنید در مواردی که مقدار دو بیت ارزش متفاوت داشته باشند، نتیجه Xor یک خواهد بود و در بقیه موارد ارزش 0 حاصل خواهد شد.

مثال ۴-۵ مقدار 45 shr 3 را محاسبه نمائید.

shr مخفف shift right می باشد و به مقدار خواسته شده بیت ها را به سمت راست منتقل می‌کند.

عدد اصلی	00101101
حاصل یکبار انتقال به راست	00010110
حاصل دوبار انتقال به راست	00001011

حاصل سه بار انتقال به راست 00000101

45 shr 3 = 5

در حالت کلی X shr Y نشاندهنده این است که عدد X در مبنای دو به تعداد خواسته شده Y به سمت راست منتقل می شود و حاصل یک عدد در مبنای ۲ می‌باشد.

مثال ۵-۵ not (164) را محاسبه کنید.

این عملگر عدد در مبنای دو را در نظر گرفته سپس بیت، با ارزش یک را به صفر تبدیل می‌کند و برعکس.

10100100 not

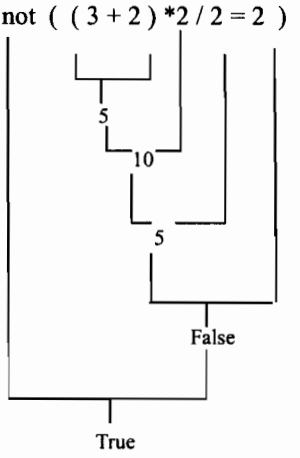
01011011 not (164) = 91

درحالت کلی تقدم عملگرها را می‌توان بصورت جدول ۷-۵ می‌توان نمایش داد.

جدول ۷-۵ تقدم عملگرها

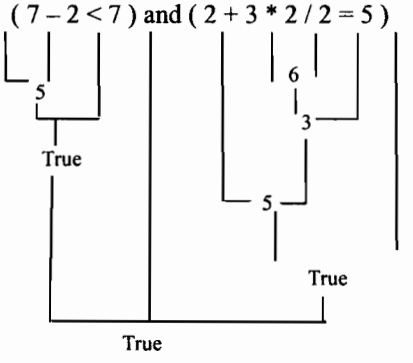
بالاترین تقدم ( )
Not
* / div / mod
+ -
Shl shr
< <= >= >
= <>
And
XOR
OR

مثال ۶-۵ مقدار (2 = 2 / 2 * (2+3)) not را محاسبه نمائید . (با توجه به اولویت عملگرها)



لذا ارزش عبارت بالا Ture می‌باشد.

مثال ۷-۵ مقدار (5 = 2 / 2 * 3 + 2) and (7 > 2 – 7) را محاسبه کنید.



لذا ارزش عبارت بالا Ture می باشد.

۲-۵- انواع داده‌ها (data types)

همانطور که در مقدمه این فصل مطرح کردیم، ورودی‌های هر برنامه دارای نوع می‌باشند و هر زبان برنامه‌نویسی این نوع داده‌ها را به گونه‌ای مشخص می‌کند. داده‌ها در پاسکال در حالت کلی به سه دسته تقسیم می‌شوند:

۵-۲-۱- داده‌های ساده (Simple Data Type)

این نوع داده‌ها، نوع‌های اصلی داده‌ها محسوب می‌شوند و بقیه داده‌ها نیز از این نوع داده‌ها حاصل می‌شوند. داده‌های ساده به دو دسته تقسیم می شوند:

الف) داده‌های استاندارد

ب) داده‌های قابل تعریف

داده‌های استاندارد به انواع مختلف تقسیم‌بندی می‌شوند که عبارتند از:

- صحیح (integer)

ورودی‌هایی که صحیح هستند جزء این دسته می‌باشند. داده‌های صحیح بر حسب اندازه حافظه در جدول ۸-۳ نمایش داده شده‌اند.

جدول ۸-۵ لیست انواع اعداد صحیح

نوع	محدوده	اندازه بر حسب بایت
Byte	از ۰ تا ۲۵۵	۱
Shortint	از ۱۲۸- تا ۱۲۷	۱
Integer	از ۳۲۷۶۸- تا ۳۲۷۶۷	۳
Word	از ۰ تا ۶۵۵۳۵	۳
Longint	از ۲۱۴۷۴۸۳۶۴۸- تا ۲۱۴۷۴۸۳۶۴۷	۴

- اعشاری (حقیقی)

زمانی که ورودی اعشاری باشد از این نوع داده، استفاده می‌کنند. مقدار حافظه مورد نیاز این داده نسبت به اعداد صحیح در کل بیشتر می باشد. مشخصات اعداد حقیقی در جدول ۹-۵ خلاصه شده است.

جدول ۹-۵ لیست انواع اعداد حقیقی

نوع	محدوده	تعداد ارقام معنی دار	اندازه برحسب بایت
Real	2.2 e – 39 ... 1.7 e 38	۱۱-۱۳	۶
Single	1.5 e -45 ... 3.4 e 37	۷-۸	۴
Double	5.0 e -324 ... 1.7 e 308	۱۵-۱۶	۸
Extended	3.4 e -4932 ... 1.1 e 4932	۱۹-۳۰	۱۰
Comp	-9.2 e 18 ... 9.2 e 18	۱۹-۳۰	۸

تذکر: نوع Comp اعداد اعشاری را شامل نمی‌شود و تنها مقادیر صحیح را در محدوده مربوطه در بر می‌گیرد ولی چون محدوده آن وسیع تر از بزرگترین محدوده اعداد صحیح می‌باشد نام آن در گروه اعداد حقیقی گنجانده شده است.

- نوع منطقی (Boolean type)

برخی از داده‌ها دو مقدار بیشتر به خود نمی‌گیرند این نوع داده‌ها را نوع داده منطقی می‌نامند و فقط ارزش True یا False را می‌تواند بپذیرد. در ضمن این نوع متغیر فقط یک بایت از فضای حافظه را اشغال می‌کند.

- نوع کاراکتری (Char type)

مقادیر این نوع داده از میان مجموعه کاراکترهای اسکی انتخاب می‌شود. به عنوان مثال 'a','b','$','*' و غیره ... به طور کلی ۲۵۶ کاراکتر وجود دارد و هر متغیر از نوع کارکتر یک بایت فضای حافظه را اشغال می‌کند.

- نوع رشته ای (String type)

رشته‌ها یکی از قوی‌ترین و پرکاربردترین نوع داده هستند و عبارتند از: مجموعه‌ای از کاراکترها.

حداکثر طول یک نوع داده رشته ای ۲۵۵ کاراکتر می‌باشد. به عنوان مثال ' pascal ' 'ali','jafar', و ... هر کدام یک رشته محسوب می‌شوند.

همانطور که ملاحظه کردید ۵ نوع داده استاندارد را در بالا مطرح کردیم.

نوع دوم داده‌های ساده همانطور در بالا ذکر کردیم، داده‌های قابل تعریف می‌باشند که شامل داده‌های شمارشی (Enumerated types) و داده‌های زیر دامنه (Subrange types) می‌باشند. (بعداً بحث خواهیم کرد)

۲-۲-۵- داده‌های ساخت‌یافته (Structural Data Types)

این نوع داده‌ها در کل به نام ساختار داده‌ای مطرح می‌شوند و از نوع داده‌های ساده منتج می‌شوند ولی خود به عنوان نوع داده با یک ساختار مشخص مطرح هستند و ابزارهای بسیار مفیدی را در اختیار برنامه‌نویس در نوشتن برنامه قرار می‌دهند. در اینجا فقط انواع آن را ارائه می‌دهیم و بحث در مورد آنها را در فصول بعدی ادامه خواهیم داد.

انواع داده‌های ساخت‌یافته عبارتند از:

- آرایه‌ها (arrays)
- رکوردها (records)
- مجموعه‌ها (sets)
- فایل‌ها (files)

۳-۲-۵- داده‌های اشاره‌گر (Pointer Data Types)

ممکن است در نوشتن برنامه، نوع داده‌های بحث شده در بالا به دلایل مختلف از جمله مشخص نبودن تعداد ورودیهای مسئله و غیره مشخص نباشد لذا نیاز به متغیرهایی هست که بتوانند آدرس متغیرهای دیگر را در خود نگه دارند، این نوع داده‌ها، داده‌های اشاره‌گر نام دارند. (بحث در این مورد بعداً مطرح خواهد شد)

۳-۵- متغیرها (Variables)

متغیر، محلی از حافظه است که دارای نوع و اسم می‌باشد. نوع متغیر همان نوع داده بوده و اسم متغیر از قواعد اسم‌گذاری شناسه تبعیت می‌کند.

در پاسکال برای معرفی متغیرها بصورت زیر عمل می‌کنند:

Var (کلمه ذخیره شده)

نوع متغیر : اسم متغیر

در معرفی متغیرها ابتدا از کلمه ذخیره شده Var استفاده می‌کنند و سپس اسامی متغیرها و نوع آنها را تعریف می نمایند. تعریف متغیرها در بخش تعاریف یک برنامه ظاهر می‌شود و محل آن بعد از اسم برنامه (در صورتی که بکار برده شود) خواهد بود.

مثال ۸-۵ متغیرهایی از نوع صحیح، اعشاری و کاراکتری تعریف کنید.

```
Var
      i , j : integer ;
      g , f : Real ;
      ch1 , ch3 : char ;
```

تذکر: همانطور که قبلاً گفتیم، بعد از اتمام هر دستور علامت ; الزامی است.

مثال ۹-۵ متغیرهایی از نوع صحیح و اعشاری تعریف کنید.

```
Var
  Radius , Volume : real ;
     i , j : integer ;
```

۵-۴- ثابت‌ها (Constants)

فرض کنید می‌خواهیم برنامه‌ای بنویسیم تا محاسبات متعددی بر روی دایره، کره و غیره انجام دهد. که همه آنها دارای ثابت پی هستند. نوشتن عدد 3.14159365 در هر بار نه تنها خسته‌کننده است، بلکه می‌تواند متضمن خطای تایپی نیز باشد. از این رو معمولاً در چنین مواقعی از ثابت ها استفاده می‌کنند.

یک ثابت نام شناسه‌ای است که در آغاز یک برنامه، یک مقدار در آن جاگزین می‌شود. درست مانند متغیرها. ثابت‌ها را می‌توان بعنوان خانه‌هایی از حافظه در نظر بگیریم که مقدار داده‌ها در آنها ذخیره می‌شود ولی مقدار ثابت مشخص می‌باشد، طوری که نمی‌توان مقدار یک ثابت را در برنامه خود بوسیله یک دستور تغییر داد.

برای تعریف یک ثابت بصورت زیرعمل می‌کنیم.

```
Const    مقدار ثابت = اسم متغیر
```

مثلاً

```
Const      Pi = 3.141593635 ;
```

بدین ترتیب می‌توان از ثابت‌ها در برنامه استفاده کرد. استفاده از ثابت‌ها خوانایی برنامه را بیشتر می کند.

۵-۵- دستور جایگزینی

برای قرار دادن یک مقدار یا مقدار یک متغیر داخل یک متغیر دیگر، از دستور جایگزینی استفاده می‌کنند.

شکل کلی یک دستور جایگزینی در پاسکال بصورت زیر است:

عبارت محاسباتی = : اسم شناسه

یا

عبارت قیاسی

یا

عبارت منطقی

مثال ۱۰-۵ دستورات جایگزینی انجام دهید.

```
Y: = 13 ;
Z: = 10 ;
X: = Y + Z ;
```

نکته: باید توجه داشته باشید مقداری که داخل یک متغیر قرار می‌گیرد، باید از نوع همان متغیر باشد. (یک نوع اعشاری را داخل یک نوع صحیح نمی‌توان قرار داد) همچنین یک نوع کاراکتری را داخل یک نوع اعشاری نمی‌توان جایگزین کرد و غیره.

مثال ۱۱-۵ کدامیک از اعمال جایگزینی زیر مجاز می‌باشند؟

```
Var
      i , j : integer ;
      f : Real ;
      ch : char ;
Begin
            i: = 1 ; j: = 13 ;
            f: = 13.5 ; ch: = ' A ' ;
            j: = i + j ;        مجاز
            ch: = i + j ;       غیر مجاز
            f: = i + j ;        مجاز
End.
```

۶-۵- افزودن توضیحات به برنامه (Comment)

افزودن مطلب توضیحی در درون خود برنامه عملی پسندیده و مطلوب است بدین ترتیب که بعد از مدتی امکان فراموشی کار با برنامه از بین می‌رود و در کل می‌توان گفت که نوشتن توضیحات در برنامه خوانایی آن را بالا می‌برد.

در پاسکال توضیحات بین دو آکولاد محصور می‌شوند.

```
{ This is comment }
{ This program written by J.Tanha }
```

و همچنین می‌توان بین یک پرانتز باز با * و یک * با پارانتز بسته توضیحات را نوشت.

```
( * This program * )
```

توجه کنید که کامپایلر توضیحات را مانند فضای خالی در نظر می‌گیرد. توضیحات را در هر جای برنامه می‌توان بکار برد.

۷-۵- چند برنامه به زبان پاسکال

در زیر چند نمونه برنامه به زبان پاسکال ارائه می‌دهیم تا مطالب گفته شده در بخش‌های مختلف را مرور کرده باشیم.

مثال ۱۲-۵ برنامه‌ای بنویسید که دو عدد ۱۰ و ۱۳ را با هم جمع کرده و در خروجی با پیغام مناسب چاپ کند.

```
Program   Example1 ;
Var
   Number1 , number2 , number3: integer ;
Begin
    Number1: = 10  ; number2: = 13  ;
    Number3: = number1 + number3  ;
    Writeln ( 'Sum of two  numbers = ' , number3  ) ;
End. { end  of  program }
```

در برنامه بالا متغیر number3 از نوع صحیح بوده و مجموع دو عدد داخل آن قرار می‌گیرد.

خروجی برنامه بالا بصورت زیر می‌باشد:

```
Sum  of  two numbers = 23
```

مثال ۱۳-۵ برنامه‌ای بنویسید که به ترتیب طول و عرض مستطیل که برابر ۵ و ۷ بوده در نظر گرفته، محیط و مساحت آن را محاسبه و در خروجی چاپ نماید.

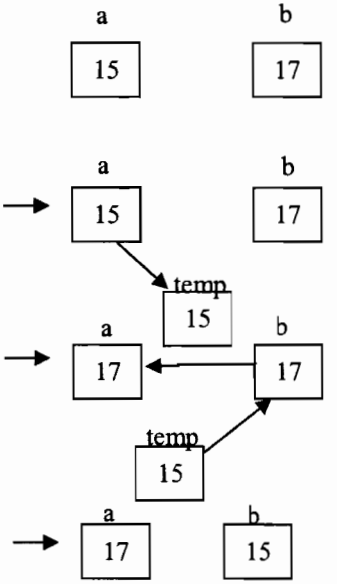
```
Program   Example2 ;
Var
    Width , length , P, S: Real ;
Begin
    Width: = 7.0 ; length: = 5 ;
    P: = 2 * ( width + length ) ;
    S: = width * length ;
    Writeln ( ' P = ' , P , ' S = ' , S ) ;
End. { end of program }
```

مثال ۱۴-۵ برنامه‌ای بنویسید که حجم یک کره به شعاع ۲۰ را محاسبه و چاپ نمایید.

```
Program    Example3 ;
Const      Pi = 3.14159365 ;
Var
     Radius , Volume: Real ;
Begin
   Radius: = 20.0 ;
   Volume: = ( 4.0 / 3.0 ) * Pi * Radius * Radius * Radius ;
   Writeln ( ' Volume = ' , Volume ) ;
End. { end of program }
```

مثال ۱۵-۵ برنامه‌ای بنویسید که دو متغیر از نوع عدد صحیح با مقادیر ۱۵ و ۱۷ را درنظر گرفته محتویات آنها را با هم جابجا نماید.

الگوریتم مسئله به صورت زیر می‌باشد :




```
Program          Example4 ;
 Var
                   a , b , temp : integer
 Begin
                   a : = 15 ; b : = 17 ; temp : = 0 ;
                   temp : = a ;
                   a : = b ;
                    b : = temp ;
                   Writeln ( ' a = ' , a , ' b = ' , b ) ;
 End . { End . of program }
```

خروجی برنامه :

a = 17 b = 15

۸-۵- نکاتی چند در مورد برنامه‌نویسی

در نوشتن برنامه رعایت موارد زیر می‌تواند در بهبود کیفیت برنامه موثر باشد.

۱. استفاده از اسامی با مفهوم برای متغیرها
۲. استفاده از دستور Const در صورتی که مقدار ثابت در برنامه وجود داشته باشد.
۳. سوال جوابی بودن برنامه (ورودی‌ها و خروجی‌ها باید دارای پیغام مناسب باشد)
۴. نوشتن برنامه با فرمت مناسب (رعایت قرار گرفتن خطوط مختلف برنامه زیر هم و فاصله گذاشتن آنها از اول سطر و غیره)

۹-۵- تمرینات

۱- با توجه به اعلان زیر:

```
Const
        Pi = 3.14159 ;
        Max = 1000 ;
Var
        X , Y: Real ;
        A , B , I: Integer ;
```

تعیین کنید کدام یک از عبارات زیر در زبان پاسکال مجاز می‌باشند و چرا ؟ سپس مقدار آن عبارت را بدست آورید.

توضیح: مقدار متغیرهای A , B , Y بترتیب 3 , 3 , 3.0 - می‌باشد.

A) I: = A mod B
B) I: = (990 – Max) div A
C) X: = A / Y
D) I: = A / B
E) X: = A mod (A / B)
F) I: = B div 0
G) I: = A mod 0
H) X: = A div B
J) X: = Pi * y
K) : = A / y

۲- حاصل عبارتهای زیر را بدست آورید:

A) 3 * 13 Mod 3 div 3 = ?
B) 7.3 * 5 / 3 = ?
C) (3 + 4 < 6) and (4 + 7 < 13) = ?
D) 33 – 8 * 3 div 3 mod (5 div 3) = ?
E) NOT (((3 – 4 MOD 3) < 5) and ((6 div 4) < > 3))

۳- حاصل عبارت‌های منطقی را به ازاء مقادیر زیر مشخص کنید.

A: = true ; B: = false ; C: = true ;

A) (A AND B) OR (A AND C) = ?
B) (A OR NOT B) and (Not A OR C) = ?
C) A OR B AND C = ?
D) NOT (A OR B) AND C = ?

۴- حاصل عبارتهای بیتی زیر را به ازاء مقادیر زیر مشخص کنید.

X: = 13 ; Y: = 3 ;

A) X and Y = ?
B) X OR Y = ?
C) NOT X = ?
D) XOR X = ?
E) X shR Y = ?
F) X shL Y = ?

۱۰-۵- تمرینات برنامه‌نویسی

۱- برنامه‌ای بنویسید که ابعاد مثلث که عبارتند از 1.3 , 3 را در نظر گرفته محیط و مساحت آن را محاسبه و با پیغام مناسب در خروجی چاپ کند.

۲- برنامه‌ای بنویسید که دو متغیر صحیح با مقادیر 15 , 3 را در نظر گرفته محتویات دو عدد را بدون استفاده از متغیر کمکی جابجا نماید.

۳- برنامه‌ای بنویسید که سه عدد بنام های Third , Second , First بترتیب با مقادیر 13 , 15 , 17 را در نظر گرفته بطور چرخشی مقادیر آنها را جابجا نموده در خروجی با پیغام مناسب چاپ کند.

۴- برنامه‌ای بنویسید که دو عدد به نامهای first وsecond با مقادیر ۱۲و ۱۳ در نظر گرفته شود و سپس مجموع و مجموع مربعات آنها را محاسبه کرده و در خروجی با پیغام مناسب چاپ نماید.

۵- برنامه‌ای بنویسید که اسم، فامیلی و شماره دانشجویی خود را در وسط صفحه نمایش چاپ کند.

فصل ۶

ورودی و خروجی

هدفهای کلی

- معرفی دستورات خروجی Write و WriteLn
- معرفی خروجی فرمت‌بندی شده
- بررسی دستورات ورودی Read و ReadLn

هدفهای رفتاری

دانشجو پس از مطالعه این فصل باید بتواند:

- عبارات مورد نظر را در خروجی نمایش دهد.
- عبارات خروجی را با فرمت مناسب چاپ کند.
- متغیرهای مورد نیاز برنامه را از ورودی دریافت نماید.
- برنامه‌های ساده به زبان پاسکال بنویسد.

مقدمه

بعد از تعریف نوع داده‌های یک برنامه نیاز است که مقادیر این متغیرها خوانده شوند و نتایج نیز چاپ شوند. این اعمال توسط دستورات ورودی و خروجی انجام می‌شود. دستورات ورودی برای خواندن مقادیر متغیرها از ورودی بکار می‌روند و دستورات خروجی نیز برای نمایش یا چاپ نتایج برنامه در خروجی (صفحه نمایش، چاپگر و غیره) بکار می‌روند.

۱-۶- خروجی با دستور Write

این دستور برای نوشتن اطلاعات در خروجی بکار می‌رود. اطلاعات خروجی می‌توانند ثابتهای عددی، مقادیر متغیرها، عبارات و غیره باشند. شکل دستور در حالت کلی بصورت زیر است:

write (متغیر۱ و متغیر۲ و .....)

یا (..... و 'عبارت ۲' و'عبارت۱ ')

یا (..... و ثابت ۲ وثابت ۱)

هدف از بکار بردن دستور Write نوشتن مقادیر منتسب به ثابت ها، متغیرها و عبارت های داخل پارانتز بوده و کنترل را برای دستورهای خواندن یا نوشتن بعدی در همان خط نگه می‌دارد.

بنابراین بعد از به کار بردن دستورچاپ، کنترل خروجی بعد از آن قرار می‌گیرد و دستورات خواندن یا نوشتن بعدی، از همان مکانی که کنترل خروجی در آن قرار دارد، ادامه پیدا می‌کند.

مثال ۱-۶ خروجی دستور زیر را تعین کنید.

```
Write ( ' Hello  World: ' ) ;
```

خروجی این دستور بصورت:

```
Hello  World:
```

توجه کنید بعد از اتمام عمل چاپ مکان‌نما در انتهای رشته قرار می‌گیرد.

مثال ۲-۶ خروجی قطعه برنامه زیر را مشخص نمائید.

```
Var
    a , b , c: integer ;
Begin
   a: = 17 ;
   b: = 5 ;
   c: = a + b ;
   Write ( ' a = ' , a , ' b = ' , b ) ;
   Write ( ' sum of two numbers = ' , c ) ;
End.
```

در صورتی که مکان‌نما در صفحه نمایش ابتدای خط باشد، در اینصورت عبارت زیر چاپ می‌شود:

```
a = 17 b = 5 sum of two numbers = 22
```

همانطور که ملاحظه می‌کنید دستور چاپ اول عبارات و متغیرها را چاپ می‌کند و با توجه به خاصیت دستور Write عبارات دستور دوم چاپ نیز پشت سر عبارات چاپ شده، نمایش داده می‌شود.

مثال ۳-۶ خروجی دستورات زیر را مشخص نمائید ؟

```
Var
    A , B: integer ;
    Ch: char ;
    R: Real ;
Begin
   A: = 10 ; B: = 15 ;
   Ch: = ' T ' ;
   R: = 12.25
   Write ( ' A = ' , A, '  B = ' , B ) ;
   Write ( ' ch = ' , ch , ' R= ' , R ) ;
   Write ( ' sum of A and B = ' , A + B ) ;
End. { end of program }
```

بعد از اجرای برنامه فوق در خروجی خواهیم داشت:

```
A = 10  B = 15  ch = TR = 1.2250000000e + 01  sum of A and B = 25
```

همانطور که ملاحظه می‌کنید، عبارات در صورتی که فاصله بین آنها داده نشود، دقیقاً پشت سر هم چاپ می‌شوند. اگر دقت کنید مشاهده می‌کنید که چون در عبارت ' B = ' اولین کاراکتر فضا خالی (space) می باشد لذا با یک فاصله از عبارت قبل از خود فاصله می‌گیرد. در صورتی که اینکار انجام نشود، همانطور که در بقیه عبارات ملاحظه می‌کنید، خروجیها دقیقاً پشت سر هم چاپ می‌شوند که خوانایی خروجی را از بین می‌برد. بنابراین خروجی باید فرمت مناسبی داشته باشد.

۲-۶- خروجی با دستور Writeln

این دستور همانند دستور Write عمل می‌کند با این تفاوت که بعد از اجرا، کنترل را به ابتدای سطر بعد منتقل می‌کند در نتیجه موجب چاپ داده‌های بعدی در ابتدای سطر بعد می‌شود.

مثال ۴-۶ قطعه برنامه مثال ۳-۶ را به صورت زیر تغییر می‌دهیم.

```
Var
    A , B: integer ;
    Ch: char ;
    R: Real ;
Begin
   A: = 10 ; B: = 15 ;
   Ch: = ' T ' ;
   R: = 12.25 ;
   Writeln ;    { new line }
   Writeln ( ' A = ', A , '  B = ', B ) ;
   Writeln ( ' Ch = ', ch , ' R = ', R ) ;
   Writeln ( ' Sum of A and B = ', A + B ) ;
End. { End of program }
```

خروجی برنامه بصورت زیر می‌باشد:

```
A = 10   B = 15
Ch = T   R = 1.225000000 e + 01
Sum of A and B = 25
```

توجه کنید دستور Writeln بدون پارامتر باعث انتقال کنترل برنامه به اول سطر جدید می‌شود به عبارت دیگر با دستور Writeln بدون پارامتر یک سطر خالی ایجاد می‌شود.

۳-۶- خروجی فرمت‌بندی شده

همانطور که در بالا مشاهده کردید با استفاده از دستورات Writeln , Write اطلاعات در خروجی چاپ می‌شود. اگر بخواهیم اطلاعات با فاصله های مشخص یا در مکان مشخصی در صفحه نمایش قرار گیرد، باید فرمت چاپ را در دستورات بیان شده مشخص کنیم.

طریقه تعیین فرمت چاپ در دستورات Writeln , Write در زیر آمده است:

- **فرمت اعداد صحیح**

فرمت اعداد صحیح بصورت زیر مشخص می‌شود:

(طول میدان: داده صحیح) Writeln یا Write

مثلاً اگر مقدار sum: = 123 باشد در اینصورت

```
Write ( sum: 5 )
```

بصورت زیر نمایش داده می‌شود.

□ □۲۳

در تعریف طول میدان برای متغیرها یا داده‌هایی از نوع صحیح به نکات زیر توجه کنید:

۱. اگر طول میدان از طول ارقام عدد صحیح بیشتر تعریف شود، عدد در منتهی‌الیه سمت راست میدان نوشته می‌شود.
۲. اگر طول میدان از طول ارقام عدد صحیح کمتر تعریف شود، طول میدان به اندازه تعداد ارقام در نظر گرفته می‌شود و طول میدان تعریف شده بی‌اثر خواهد بود.

مثال ۵-۶ نتایج حاصل از دستورات زیر را بررسی کنید.

```
X:= 3200 ;
A: = 12 ;
B: = 217 ;
Write ( X:3 , A:5 , B:5 ) ;
```

خروجی : 3200 □ □ □ 12 □ □217

همانطور که ملاحظه می کنید، در صورتی که طول میدان از طول عدد مورد نظر بیشتر باشد، عدد در منتهی الیه سمت راست میدان نوشته می‌شود.

مثال ۶-۶ نتایج حاصل از دستورهای زیر را بررسی نمائید.

```
A: = 200 ; B: = 215 ;
Write ( A: 2 , B: 2 ) ;
```

خروجی: 200215

باید توجه کنید درصورتی که طول میدان کوچکتر از طول عدد باشد بی اثر خواهد بود.

- **طول میدان اعداد اعشاری**

برای نمایش اعداد اعشاری بصورت دلخواه، می توان با تعریف طول میدان و تعداد ارقام اعشاری، عدد مزبور را نمایش داد. در حالت کلی:

Write (متغیر اعشاری: طول میدان: تعداد ارقام بعد از ممیز)

یا Writeln

برای مثال درصورتی که F = 12.415 باشد در اینصورت خروجی بصورت خواهد بود:

```
Write ( f: 8:3 )
```

□ □ 12.415

در تعریف فرمت برای اعداد اعشاری به نکات زیر باید توجه کرد:

۱. اگر طول میدان بزرگتر از تعداد ارقام عدد ذکر شود، عدد در منتهی الیه سمت راست میدان چاپ می‌شود.
۲. اگر فقط طول میدان ذکر شود، عدد به صورت نماد علمی در طول میدان مشخص شده چاپ می‌شود.
۳. از آنجائی که برای نمایش اعداد در نماد علمی حداقل ۸ محل مورد نیاز است، لذا هنگامی که تنها طول میدان ذکر شده باشد، اگر از ۸ رقم کمتر باشد، حداقل ۸ رقم در نظر گرفته می‌شود.
۴. هنگامی که طول میدان همراه با تعداد ارقام بعد از ممیز ذکر شود، اگر طول میدان کوچکتر از مقدار عدد باشد، پاسکال تنها طول میدان را به اندازه‌ای که مورد نیاز

است تصحیح کرده و آنرا برابر اندازه واقعی که عدد در آن قرار می‌گیرد، اصلاح می‌کند.

۵. اگر تعداد ارقام بعد از ممیز زیاد باشد و تعداد ارقام بعد از ممیز ذکر شده در طول میدان کمتر از تعداد ارقام اعشاری عدد باشد، تعداد ارقام اعشار مطابق درخواست برنامه نویس نشان داده خواهد شد و رقم آخر اعشار آن نسبت به عدد بعدی گرد می‌شود.

مثال ۶-۷ نتایج حاصل از دستورات زیر را بررسی کنید.

```
F : = 3.1464 ;
Write ( f : 8 : 3 ) ;        □□□3.146
Write ( f : 8 : 2 ) ;        □□□□3.15
Write ( f : 8 : 4 ) ;        □□3.1464
Write ( f : 3 : 1 ) ;         3.1
Write ( f : 2 : 1 ) ;         3.1
```

- **طول میدان کاراکترها و رشته‌ها**

برای نمایش رشته‌ها و کاراکترهای با طول میدان بصورت زیر عمل می‌کنیم.

Write (طول میدان: متغیر یا عبارت رشته‌ای یا کاراکتری)

در توربو پاسکال، کلیه موارد گفته شده در مورد اعداد صحیح برای رشته‌ها نیز صادق است. در مواردی که طول میدان کوچکتر از طول رشته مورد نظر باشد، رشته چاپ خواهد شد. مثلاً برای Pascal در فوق همان Pascal چاپ می‌شود.

مثال ۶-۸ نتایج حاصل از دستورات زیر را بررسی کنید.

```
Ch: = ' T ' ;
Write ( ch: 5 ) ;                    □□□□T
Write ( ' pascal  program ': 47 )    وسط صفحه نمایش چاپ می شود
Write ( ch: 2 ) ;                    □T
Write ( ' hello ': 7 ) ;             □□hello
Write ( ' Pascal ': 3 ) ;            Pascal
```

نکته: توجه کنید در پاسکال استاندارد اولیه، به اندازه فیلد مورد نظر رشته چاپ خواهد شد. مثلا برای مورد بالا فقط Pas چاپ خواهد شد.

همانطور که در بالا بحث شد می‌توان به خروجیها فرمت داد. استفاده از فرمت مناسب در خوانایی خروجی برنامه بسیار مفید می باشد و به بیان دیگر دادن فرمت مناسب به خروجی الزامی است.

۴-۶- ورودی با Readln , Read

از این دستور برای خواندن داده ها و اختصاص آنها به متغیرها استفاده می‌شود. در خواندن داده ها به دو موضوع باید دقت شود: ۱- منبع داده‌ها یعنی دستگاه ورودی که از آن داده‌ها خوانده می‌شود. ۲- متغیری که داده‌های خوانده شده در آن قرار می‌گیرد. توجه کنید که نوع داده‌های خوانده شده باید با نوع متغیرهایی که در آنها قرار می‌گیرند، یکی باشد.

نکته: متغیرهای بولین قابل خواندن از ورودی نیستند.

شکل کلی دستور ورودی Read بصورت زیر می‌باشد:

Read (متغیر ۱ و متغیر ۲ و .....) ;

این دستور عمل خواندن داده‌ها و ذخیره آنها در متغیرها را انجام می‌دهد و پس از اتمام عمل خواندن کنترل را برای خواندن و نوشتن‌های بعدی در همان خط نگه می‌دارد.

شکل کلی دستور ورودی Readln نیز دقیقاً مثل دستور Read می‌باشد، با این تفاوت که دستور readln باعث می‌شود که پس از انتقال داده به متغیر، کنترل به خط بعد انتقال یابد در نتیجه دستورات خواندن و نوشتن بعدی داده‌ها را از خط بعدی ادامه می‌دهد.

مثال ۹-۶ به این مثال توجه کنید.

```
Var
        a , b , c: integer ;
        f: Real ;
          .
          .
          .
        Read ( a , b , c ) ;
        Read ( f ) ;
```

موقع اجرای برنامه بدین صورت داده‌ها وارد می‌شوند.

```
12  7  15  16.25
```

با فرض اینکه اعداد 12 , 7 , 15 , 16.25 می‌باشند.

مثال ۱۰-۶ با فرض اینکه داده‌های ورودی شامل ۱۲ عدد زیر در ۲ سطر باشد:

```
1  2  3  4  5  6
7  8  9  10  11  12
```

و در برنامه دستورات زیر را داشته باشیم:

```
Read ( A , B , C ) ;
Read ( D , E , F) ;
```

بعد از اجرای دو دستور فوق بترتیب مقادیر 1 , 2 , 3 به متغیرها A , B , C و مقادیر 4 , 5 , 6 به متغیرهای D , E , F تخصیص می‌یابد.

حال اگر به جای دستورات Read از دستورات Readln بصورت زیر استفاده کنیم.

```
Readln ( A , B , C ) ;
Readln ( D , E , F ) ;
```

بترتیب مقادیر 1 , 2 , 3 به متغیرهای A , B , C و مقادیر 7 , 8 , 9 به متغیرهای D , E , F اختصاص خواهد یافت.

مثال ۱۱-۶ برنامه‌ای بنویسید که دو عدد با پیغام مناسب از ورودی دریافت کرده سپس محتویات دو عدد را با هم جابجا نموده با پیغام مناسب در خروجی چاپ کند.

```
program     Example_1  ( input , output ) ;
Var
            First , second , temp: integer ;
Begin
     Writeln ;
     Writeln ( 'Please  Enter  two  numbers ' ) ;
      Readln ( first , second ) ;
      Temp: = first  ;
      First: = second  ;
      Second: = temp  ;
      Write ( ' first = ' , first , ' second = ' : 10 , second ) ;
End. { end  of  program }
```

خروجی برنامه بالا به صورت زیر می‌باشد:

```
Please Enter two numbers
   15    17
first = 17   second = 15
```

مثال ۱۲-۶ برنامه‌ای بنویسید که قطر یک دایره را از ورودی دریافت کرده، محیط آن را محاسبه و با پیغام مناسب چاپ نمائید.

```
program     Example_2 ;
const       pi = 3.14159 ;
Var
            Circum , DIAMETER: Real ;
Begin
    Writeln ( ' please  enter  DiAMETER ' ) ;
    Readlen ( DIAMETER ) ;
    Circum: = pi * DIAMETER ;
    Write ( ' CiRcum FERENCE  is  = ' , Circum: 10: 4 ) ;
End.
```

خروجی قطعه بالا با توجه به مقدار خوانده شده محاسبه و چاپ می‌شود.

مثال ۱۳-۶ برنامه‌ای بنویسید که یک مقدار طول را بر حسب اینچ دریافت کند و معادل آن را بر حسب سانتیمتر چاپ کند.

```
program     Example_3 ;
Const       Centperinch = 2.54 ;
Var
            Inches: integer ;
            Cent: Real ;
Begin
    Write ( ' Enter  a  length  in  inches: ' ) ;
    Readln ( inches ) ;
    Cent: = Centperinch * inches ;
    Writeln ( inches , ' inches  Equals ' , Cont: 5: 2 )
End.
```

خروجی حاصل از اجرای این برنامه بصورت زیر است.

```
Enter  a  length  in  inches: 100
100  inches  Equals  254.00
```

۶-۵- تمرینات

۱- خروجی قطعه برنامه زیر را تعیین کنید ؟

```
Value1: = 27.3 ;
Value2: = -8.5 ;
Writeln ( ' Value1 is ' , Value1 ) ;
Writeln ( ' Value2 is ' , Value2 ) ;
Sum: = Value1 + Value2 ;
Writeln ( ' Sum of Two Values = ' , Sum: 6: 2 ) ;
```

۲- با توجه به داده های زیر خروجی دستورات را مشخص کنید.

داده های ورودی:

```
1   2   3   4   5   6
7   8   9   10  11  12
13  14  15  16  17  18
```

الف)

```
Read ( first ) ;
Readln ( second ) ;
Readln ( Third ) ;
Writeln ( ' They are:: ' ) ;
Write ( first: 6 , second: 6 ) ;
Writeln ( Third: 6 ) ;
```

ب)

```
Readln ( first ) ;
Read ( first , second ) ;
Readln ( second ) ;
Read ( Third ) ;
Write ( ' They are ' ) ;
Writeln ( first: 6 , second: 6 , Third: 6 ) ;
```

ج)

```
Readln ( first , second ) ;
Readln ( Third , second , Third ) ;
Read ( Third ) ;
Writeln ( first: 6 ) ;
Writeln ( second: 6 , Third: 6 ) ;
```

۳- نمودار دستوری برای دستور Read و Writeln را رسم نمائید.

۴- اگر متغیر X از نوع real و مقدار آن 12.235 و متغیر 3 از نوع صحیح و مقدار آن 100 باشد خروجی دستورات زیر را تعیین کنید؟

```
Writeln ( ' X is ': 10 , X: 6: 2 , ' I is ': 4 , I: 5 ) ;
Writeln ( ' I is ': 10 , I: 1 ) ;
Writeln ( ' X is ': 10 , X: 2: 1 ) ;
Writeln ( ' X is ': 15 , X: 7: 1 ) ;
Writeln ( ' I is ': 10 , ' X is ': 10 , X: 7: 3 ) ;
```

۶-۶- تمرینات برنامه‌نویسی

۱- برنامه‌ای بنویسید که دو عدد را از ورودی دریافت کرده و محتویات آنها را بدون استفاده از متغیر کمکی جابجا نماید.

۲- برنامه‌ای بنویسید که سه عدد صحیح Third , Second , first را از ورودی با پیغام مناسب دریافت کرده سپس محتویات این سه متغیر را بصورت چرخشی جابجا نموده با پیغام مناسب در خروجی چاپ کند.

۳- برنامه‌ای بنویسید که دمای هوا بر حسب فارنهایت را خوانده، به درجه سانتیگراد تبدیل نموده و در خروجی چاپ کند.

۴- برنامه‌ای برای یک حسابدار اداره جمع آوری مالیات بنویسید که صورت حسابهای مالیات را محاسبه نماید.

ورودی: شماره شناسایی مالیات دهنده

بهای ارزیابی شده

نرخ مالیات

خروجی: صورت حساب بافرمت مناسب شامل تمام داده‌های ورودی و میزان بدهی

۵- برنامه‌ای بنویسید که تا عدد صحیح m را از ورودی دریافت کرده و به کمک رابطه Sum = ((m+1)/2) m مقدار sum را محاسبه و با پیغام مناسب در خروجی چاپ نماید.

۶- برنامه‌ای بنویسید که اطلاعات مربوط به یک دانشجو را دریافت کرده کارنامه آن را در خروجی به همراه معدل با فرمت مناسب چاپ نماید.

ورودی: شماره دانشجوئی

درس پاسکال نمره درس پاسکال تعداد واحد

درس ریاضی عمومی نمره درس ریاضی تعداد واحد

خروجی: مشخصات دانشجو (شماره دانشجوئی) کارنامه (اسم دروس، تعداد واحدها، نمره دروس) و معدل کل دانشجو

فصل ۷

ساختارهای شرطی و کنترلی

هدفهای کلی

- معرفی دستور شرطی if و else
- معرفی دستور case
- معرفی دستورات شرطی متداخل
- بررسی دستورات تکرار for ، while ، repeat until
- بررسی چند تابع و روال استاندارد زبان

هدفهای رفتاری

دانشجو پس از مطالعه این فصل باید بتواند:

- برنامه‌هایی را بنویسد که در آنها نیاز به استفاده از شرط وجود دارد.
- تفاوتهای بین دستورات مختلف با if و if else را تشخیص دهد.
- برنامه‌هایی که نیاز به تکرار تعدادی عملیات داشته باشند را بنویسد.
- در صورت نیاز بتواند در برنامه‌ها، از روالها و توابع استاندارد زبان استفاده نماید.

مقدمه

تاکنون ساختار کلی برنامه در زبان پاسکال را بحث نمودیم. در این فصل ساختارهای شرطی و کنترلی را مورد بررسی قرار می دهیم. هرگاه در یک برنامه نیاز به استفاده از شرط یا شروط را احساس کنیم (که غالباً نیاز داریم) در این صورت از ساختارهای شرطی استفاده می‌کنیم. بطور مثال، بررسی می‌کنیم که اگر شرطی برقرار باشد، چه اعمالی را باید انجام دهیم و ممکن است در صورت برقرار نبودن شرط عمل دومی را انجام دهیم. در بسیاری از مواقع نیز نیازمند تکرار تعدادی از اعمال به تعداد معین هستیم لذا در چنین مواقعی از ساختارهای کنترلی یا تکرار استفاده خواهیم کرد.

۱-۷- دستورات شرطی

بطور کلی توسط اینگونه دستورات می‌توان بر حسب شرایط مختلف، تصمیمات متفاوتی را اتخاذ نمود و برحسب برقرار بودن یا نبودن شرایط دستورات متفاوتی را اجرا نمود.

دستورات شرطی در حالت کلی به دو نوع تقسیم می‌شوند:

۱. دستور if
۲. دستور case

۱-۱-۷- دستور if

هرگاه در طول برنامه نیاز به استفاده از شرط یا شروط داشته باشیم، از دستور if استفاده می‌کنیم. if همانطور که قبلاً اشاره کردیم، یک کلمه ذخیره شده می‌باشد و توسط آن می‌توان شرط یا شروطی را بررسی کرد، در صورتی که شرط برقرار باشد آنگاه اعمال خاصی انجام می‌شود و در غیر این صورت برنامه روال عادی خود را طی می‌کند و یا دستورات خاص دیگری اجرا می‌شوند.

دستور if بطور کلی به سه شکل بر حسب نیاز ممکن است ظاهر شود.

۱. if ساده
۲. if همراه else
۳. if های متداخل

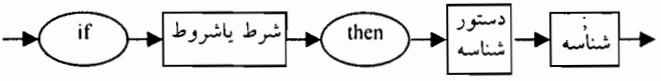
- **If – then**

در این نوع دستور شرطی اگر شرط خاصی تحقق یافته باشد، عمل یا اعمال خاصی انجام می‌شود. در غیر اینصورت برنامه روال عادی خود را طی می‌کند، در صورتی که شرط برقرار باشد ارزش منطقی Ture به خود می‌گیرد و اگر شرط برقرار نباشد، ارزش منطقی False به خود خواهد گرفت.

شکل کلی دستور if بصورت زیر می‌باشد:

if شرط یا شروط then
دستور ;

دیاگرام دستور بالا بصورت زیر می‌باشد: (شکل ۱-۷)



شکل ۱-۷ دیاگرام دستور if

اگر در دستور if با توجه به نیاز، چند شرط ظاهر شود، از عملگرهای منطقی می‌توانید استفاده کنید (and , or و غیره)

مثال ۱-۷ برنامه‌ای بنویسید که عددی را از ورودی دریافت کرده، مثبت بودن آنرا بررسی نماید.

```
Program Example1 ;
Var
        Number: integer ;
Begin
        Writeln ( 'Please enter Number: ' ) ;
        Readln ( Number ) ;
        if  Number > 0  Then
            Write ( ' Number is positive ' ) ;

End.
```

خروجی برنامه بالا بصورت زیر است:

```
Please enter number: 12
Number is positive
```

مثال ۲-۷ برنامه‌ای بنویسید که عددی را از ورودی دریافت کرده، در صورتی که زوج باشد پیغام مناسب را در خروجی چاپ نماید:

```
Program         Example2 ;
Var
                Number: integer ;
Begin
        Writeln ( 'Please enter number: ' ) ;
        Readln ( Number ) ;
        if ( Number mod 2 ) = 0  then
            Write ( ' The number is even ' ) ;
End. { end of program }
```

خروجی برنامه به صورت زیر می‌باشد:

```
Please enter number: 12
The number is even
```

مثال ۳-۷ برنامه‌ای بنویسید که عددی را از ورودی دریافت کرده و بخشپذیری آن بر ۳ را بررسی نماید.

```
Program         Example3 ;
Var
                Number: integer ;
Begin
      Writeln ( 'Please enter number: ' ) ;
      Readln ( Number ) ;
      if ( number mod 3 ) = 0 then
          Write ( ' Divisible ' ) ;
End.
```

خروجی:

```
Please enter number: 12
Divisible
```

مثال ۴-۷ برنامه‌ای بنویسید که کاراکتری از ورودی دریافت کرده، در صورتی که کاراکتر خوانده شده 'p' یا 'P' باشد در خروجی رشته 'pascal' را نمایش دهد.

```
Program          Example4 ;
Var
                 Ch: char ;
Begin
    Writeln ( 'Please enter character: ' ) ;
    Readln ( ch ) ;
    if ( ch = ' p ' ) or ( ch = ' P ' ) then
        Write ( ' pascal' ) ;
End. { end of program }
```

خروجی:

```
Please enter character: p
pascal
```

مثال ۵-۷ برنامه‌ای بنویسید که یک کاراکتر عددی را از ورودی دریافت کرده، معادل حرفی آن را در خروجی چاپ کند.

```
Program          Example5 ;
Var
                 Ch: char ; { ch is 'o'  or '1' .... or '9' }
Begin
   Writeln ( 'Please enter character: ' ) ;
   Readln ( ch ) ;
   if ch = '0' Then
         Write ( ' Zero ' ) ;
   if ch = '1' Then
         Write ( ' one ' ) ;
   if ch = '2' Then
         Write ( ' Two ' ) ;
   if ch = '3' Then
         Write ( ' Three ' ) ;
   if ch = '4' Then
         Write ( ' Four ' ) ;
   if ch = '5' Then
         Write ( ' Five ' ) ;
   if ch = '6' Then
         Write ( ' Six ' ) ;
   if ch = '7' Then
          Write ( ' Seven ' ) ;
   if ch = '8' Then
          Write ( ' Eight ' ) ;
   if ch = '9' Then
           Write ( ' Nine ' ) ;
End. { end of program }
```

خروجی برنامه بالا:

```
Please enter character: 7
Seven
```

با ورودیهای دیگر خروجی های مختلف نمایش داده می شود.

با توجه به مثال‌های ارائه شده در این بخش ملاحظه می‌کنید که در صورتی که شرط برقرار باشد، برنامه خروجی مناسب خواهد داشت ولی اگر شرط برقرار نباشد، برنامه خروجی خاص را نمایش نمی‌دهد. لذا نوع دوم شرط ها که همراه Else می‌باشد را بحث می‌کنیم.

- **دستور if همراه else**

در این دستور ابتدا شرط بررسی می‌شود، در صورتی که شرط برقرار باشد، عمل یا اعمال خاصی را انجام می‌دهد و در صورتی که شرط برقرار نباشد، عمل یا اعمال بخصوص دیگری را انجام خواهد داد. اینگونه دستورات در واقع حالت توسعه یافته دستورات if می‌باشند.

شکل کلی این دستور بصورت زیر است:

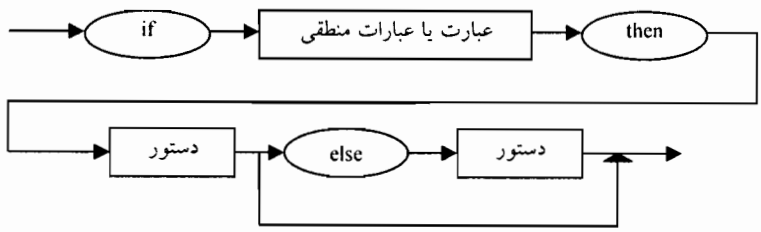
if عبارت یا عبارات منطقی then

دستور۱

else

دستور۲ ;

دیاگرام دستور بصورت:



شکل ۲-۷ دیاگرام دستور if-else

اگر در دیاگرام بالا (شکل ۲-۷) خوب دقت کنید، ملاحظه می‌کنید که در صورت نیاز از else استفاده می‌کنیم، معمولاً برای تکمیل شدن سوال و جوابی بودن برنامه از else استفاده می‌کنند و بودن آن خوانایی برنامه را بیشتر می‌کند.

نکته: در بکارگیری دستور If همراه else باید توجه کنید که بعد از ' دستور۱' شما مجاز به استفاده از علامت ' ; ' نیستید، در صورت رعایت نکردن کامپایلر زبان

پاسکال از شما اشکال می‌گیرد.

کامپایلر زمان ترجمه برنامه شامل if، ابتدا شرط را بررسی می‌کند، اگرشرط برقرارباشد، دستور یا دستورات داخل آن اجرا می‌شود، سپس کنترل برنامه به قسمتهای بعدی برنامه که پس از دستور مربوط به else قرار گرفته، منتقل می شود و درصورت برقرار نبودن شرط، کنترل برنامه دستور یا دستورات مربوط به else را اجرا می‌کند.

مثال ۶-۷ برنامه‌ای بنویسید که عددی را از ورودی دریافت کرده، فرد و زوج بودن آن را بررسی نماید.

```
Program    Example6 ;
Var
       Number: integer ;
Begin
   Writeln ( 'Please enter number: ' ) ;
    Readln ( Number ) ;
    if ( Number mod 2 ) = 0 Then
       Write ( ' even ' )
   else
       Write ( ' odd ' ) ;
End. { end of program }
```

با عدد 12 خروجی بصورت:

```
Please enter number:
12
even
```

با عدد 17 خروجی بصورت:

```
Please enter number:
17
odd
```

مثال ۷-۷ برنامه‌ای بنویسید که دو عدد از ورودی دریافت کرده، بیشترین مقدار را از بین دو عدد پیدا کرده و در خروجی چاپ نماید.

```
Program      Example 7 ;
Var
       Max , a , b  : integer ;
Begin
    Write ( 'Please enter two numbers: ' ) ;
     Readln ( a , b ) ;
     if  a >b  then
           Max: = a
    else
           Max: = b  ;
```

```
    Write ( ' Max = ' , Max ) ;
End.
```

خروجی برنامه:

```
Please enter two numbers: 12  17
Max = 17
```

برنامه بالا را می‌توان به صورت دیگری نیز نوشت:

```
Program          Example7 ;
Var
                  Max , a , b  : integer ;
Begin
    Write ( ' please enter Two Numbers: ' ) ;
    Readln ( a , b ) ;
    Max: = a ;
    if  b > Max  then
        Max: = b ;
    Write ( ' Max = ' , Max ) ;
End. { end of program }
```

خروجی برنامه:

```
Please enter two numbers: 12  17
Max = 17
```

نکته: در صورتی که بخواهیم در دستور If مجموعه‌ای از دستورات بعد از برقراری شرط یا شروط در قسمت Else قرار گیرند، باید داخل یک بلوک یعنی بین Begin و End قرار گیرند و می توان آن را بصورت‌های زیر در برنامه اجرا کرد:

۱)

```
if      عبارت منطقی      then
begin
```

```
    1 دستور ;
    2 دستور ;   } دستور مرکب
                .
                .
                .
end
else
begin
      1 دستور ;
      2 دستور ;   } دستور مرکب
            .
            .
            .
 end ;
```

۲)
```
if     عبارت منطقی      then
 begin
      1 دستور ;  ┐
                 ├ دستور مرکب
 2 دستور ;       ┘
                    .
                    .
                    .
 end ;
```

مثال ۸-۷ برنامه‌ای بنویسید که دو عدد اعشاری x , y از ورودی دریافت کرده، در صورتی که x > y باشد مقادیر آنها را تعویض نماید و در غیر اینصورت مقدار y را چاپ نماید.

```
Program          Example8 ;
Var
        temp , X , Y  : Real  ;
Begin
         Write ( 'Please  enter  two  real  numbers: ' ) ;
         Readln ( x , y ) ;
         if  x > y  then
           begin
               Temp:= X  ;
               X: = Y  ;
               Y: = temp  ;
               Write ( ' x = ' , x: 6: 2 , ' y = ' : 4 , y: 6: 2 )  ;
           end
         else
            Write ( ' y = ' , y: 6: 2  )  ;
End .
```

خروجی برنامه:

```
Please enter two real numbers: 13.25   12.5
X = 12.5   Y = 13.25
```

نکته: کلمات رزرو شده else , then خود یک دستور مستقل نبوده بلکه جزئی از دستور می‌باشند بنابراین قبل یا بعد از این کلمات نباید از علامت سمیکلان (;) استفاده شود.

مثال ۹-۷ در برنامه زیر دستورات مربوط به if , else نوع مرکب بکار برده شده است.

```
Program  Example9 ;
Const
   A1 = 10 ;
   A2 = 20 ;
Var
      M: Real ;
Begin
       Writeln ( 'Please enter a real numbers ' ) ;
       Readln ( M ) ;
       if  M > 0.0  Then
         Begin
           Writeln ( ' This number is positive ' ) ;
           Writeln ( ' And its sum with 10 is ' ) ;
           Writeln ( ( M + A1): 6: 2 ) ;
         End
        else
        Begin
           Writeln ( ' This number is Zero or negative ' ) ;
           Writeln ( ' And its sum with 20 is ' ) ;
           Writeln ( ( M + A2): 6: 2 ) ;
       End ;
   Writeln ;
   Writeln ( ' press enter key ... ': 30 ) ;
   Readln ;
End.
```

خروجی برنامه:

```
Please enter a real number
12.5
This number is positive
And Its Sum With 10 is
 22.5
```

- **If متداخل**

هرگاه در نوشتن برنامه نیاز به انتخاب یک شرط از بین چند شرط داشته باشیم، معمولاً از If متداخل استفاده می‌کنند. در چنین مواقعی استفاده از If متداخل کارائی برنامه را بالا می‌برد زیرا بجای کنترل تمام شروط فقط تا زمانیکه شرط برقرار نشده، If ها بررسی می‌شوند. بعد از برقرار شدن یکی از شروط، کنترل برنامه به بعد از If منتقل می‌شود و این در بهبود کارائی یک برنامه می‌تواند بسیار موثر باشد.

در حالت کلی If متداخل به صورت های زیر ممکن است، در برنامه ظاهر شود.

۱) if عبارت شرطی۱ then
if عبارت شرطی۲ then
دستور ۱
else
دستور۲ ;

۲) if عبارت شرطی۱ then
دستور۱
else if عبارت شرطی۲ then
دستور۲
else if عبارت شرطی۳ then
دستور۳
else
.
.
.

در if های متداخل شکل ۱ ملاحظه می‌کنید در صورتی که شرط1 برقرار نباشد، بقیه شرطها بررسی نمی‌شوند و این کارائی برنامه را بهبود می‌بخشد.

در if های متداخل شکل ۲ در صورتی که شرط1 برقرار باشد، بقیه شروط اصلاً بررسی نمی‌شوند و کنترل برنامه به بعد از If ها منتقل می‌شود.

با بررسی بیشتر if های متداخل در خواهید یافت که این دستور دارای اثر حذف یا دست کم، کاهش ارزیابی مکرر شرط‌ها می‌باشد.

نکته: در دستورات if متداخل اولین else مربوط به آخرین if می‌باشد.

مثال ۱۰-۷ برنامه نوشته شده در مثال ۵-۷ را با دستورات if متداخل بنویسید.

```
Program          Example10 ;
Var
                 Ch: char ;
Begin
    Write ( 'Please enter a character: ' ) ;
    Readln ( ch ) ;
    if ch = '0' Then
            Write ( ' Zero ' )
    else if ch = '1' Then
                    Write ( ' one ' )
    else if ch = '2' Then
            Write ( ' Two ' )
    else if ch = '3' Then
            Write ( ' Three ' )
    else if ch = '4' Then
            Write ( ' four ' )
    else if ch = '5' Then
            Write ( ' five ' )
    else if ch = '6' Then
             Write ( ' six ' )
    else if ch = '7' Then
             Write ( ' seven ' )
    else if ch = '8' Then
             Write ( ' eight ' )
    else if ch = '9' Then
             Write ( ' nine ' )
    else
             Write ( ' Error ! ' ) ;
    Writeln ;
    Writeln ( ' press Enter Key ... ': 30 ) ;
   Readln ;
End. { end of program }
```

با دقت در برنامه بالا ملاحظه می‌کنید که در صورتی یکی از شروط اجرا شود، شرط‌های بعد از آن بررسی نمی‌شود و کنترل برنامه به دستور بعد از If ها منتقل می‌شود.

مثال ۱۱-۷ برنامه‌ای بنویسید که نمره دانشجوئی را از ورودی دریافت کرده، با توجه به مقدار نمره یکی از خروجی های زیر را نمایش دهد:

Grade	خروجی
17 – 20	A
14 - 17	B
12 – 14	C
10 – 12	D
0 – 10	F

```
Var
                    Grade  : Real ;
Begin
  Write ( 'Please enter a real number : ' ) ;
  Readln ( Grade ) ;
    if  Grade > = 17.0  Then
       Writeln ( ' Grade is A ' )
    else If Grade > = 14.0 Then
       Writeln ( ' Grade is B ' )
    else If Grade > = 12.0 Then
       Writeln ( ' Grade is C ' )
    else If Grade > = 10
       Writeln ( ' Grade is D ' )
    else
       Writeln ( ' Grade is F ' );
    Writeln ( ' Press any Key ... ' : 30 ) ;
    Readln ;
End . { end of program }
```

خروجی برنامه بالا :

```
Please enter a real number : 12.75
Grade is C
```

ملاحظه می کنید که بعد از شرط سوم، بقیه شروط بررسی نمی‌شوند.

۲-۱-۷- دستور Case

دستور If یک برنامه را مجاب می‌کند تا به جایی انشعاب پیدا کند، یعنی با توجه به درستی یا نادرستی یک شرط، برنامه یکی از دو مسیر را دنبال کند. با وجود این در بسیاری از مواقع یک انشعاب دو طرفه، روشی طبیعی نیست (مثال ۱۰-۷ را ببینید) و ممکن است برنامه را پیچیده کند و فراموش نمودن یک Else یا برخی خطای ناشی از بی‌دقتی دیگر چقدر ساده بنظر می‌رسد.

زبان پاسکال دستورCase را برای چنین کاربردهایی بصورت زیر در نظر می‌گیرد:

```
Case      عبارت      Of
مقدار 1     :     دستور 1  ;
مقدار 2     :     دستور 2  ;
مقدار 3     :     دستور 3  ;
.
.
.
Otherwise
          دستور ;
End ; { End of case }
```

مثالی را بررسی می کنیم که نخست با دستورات if تودرتـو و سـپس بـا دسـتور Case پیاده سازی شده است.

مثال ۱۲-۷ برنامه‌ای بنویسید که عدد صحیح N (بـین ۱ تـا ۷) را از ورودی دریافـت کرده، معادل آن، نام روز مورد نظر در هفته را چاپ نماید.

```
Program          Example12 ;
Var
                   N : integer ;
Begin
        Writeln ( 'Please enter a number : ' ) ;
        Read ( N ) ;
        Write ( ' Day ' , N : 3 , ' is ' ) ;
        if N = 1 Then
              Writeln ( ' SATURDAY ' )
        else If N = 2 Then
              Writeln ( ' Sunday ' )
        else If N = 3 Then
              Writeln ( ' Monday ' )
        else If N = 4 Then
              Writeln ( ' Tuesday ' )
        else If N = 5 Then
              Writeln ( ' Wednesday ' )
        else If N = 6 Then
              Writeln ( ' Thursday ' )
        else If N = 7 Then
              Writeln ( ' Friday ' )
        else
              Writeln ( ' Error ! ' );
```

```
        Writeln ( 'Press any Key ... ' ) ;
        Readln ;
End . { end of program }
```

اگر در برنامه بالا دقت کنید، مقدار صحیح n با یکی از حالت‌های بالا برابر هست. در چنین مواقعی استفاده از دستور Case بسیار مفید می‌باشد.

```
Program        Example12 ;
 Var                { Using Case }
                  N : integer ;
Begin
                 Writeln ( 'Please enter a number : ' ) ;
                  Readln ( N ) ;
                  Write ( ' Day ' , N : 3 , ' is ' )
                  Case  N  of
                        1:  Writeln ( ' Saturdey ' ) ;
                        2:  Writeln ( ' Sunday ' ) ;
                        3:  Writeln ( ' Monday ' ) ;
                        4:  Writeln ( ' Tuesday ' ) ;
                        5:  Writeln ( ' Wednesday ' ) ;
                        6:  Writeln ( ' Thursday ' ) ;
                        7:  Writeln ( ' Friday ' ) ;
                        otherwise
                            Writeln ( ' Error !      )
                     End ; { End of Case }
                     Writeln ( ' Press any Key ... ' ) ;
                     Readln ;
End . { end of program }
```

خروجی برنامه بالا :

```
Please enter a number
3
Day 3 is Monday
```

نکته: توجه کنید که متغیری که در دستور Case ظاهر می‌شود، می‌تواند از نوع صحیح، اعشاری و کاراکتری باشد. در این دستور فقط حالت تساوی بین متغیر و مقادیری که در دستور ظاهر شده‌اند، بررسی می‌گردد و عملگرهای دیگر مفهومی ندارند.

مثال ۱۳-۷ برنامه‌ای بنویسید که دو عدد به همراه یک عملگر را از ورودی دریافت کرده، کار یک ماشین حساب ساده را شبیه‌سازی نماید.

```
Program          Example13 ;
 Var
                  a , b: Real ;
                   op: char
Begin
     Write ( 'Please enter two numbers: ' ) ;
      Readln ( a , b ) ;
     Write ( 'Please enter a operator: ' )
      Readeln ( op ) ;
      Case op of
         ' + ':  Writeln ( ' Sum = ' , ( a + b ): 6: 2 ) ;
         ' - ':  Writeln ( ' Subtract = ' , ( a - b ): 6: 2 ) ;
         ' * ':  Writeln ( ' Multiple = ' , ( a * b ): 6: 2 )
         ' / ':  Writeln ( ' divide = ' , ( a / b ): 6: 2 ) ;
         otherwise
          Writeln ( ' Error ! ') ;
      end. { End of Case }
End. { end of program }
```

خروجی:

```
Please enter two numbers: 12  4.0
Please enter a operator: -
Subtract =  8.00
```

دستور Case به شکل‌های مختلف می‌تواند ظاهر شود که در زیر به آنها اشاره می‌کنیم:

1) Case عبارت Of

مقدار۱ , مقدار۲ , ... : دستور۱ ;

.
.

End ;

2) Case عبارت Of

مقدار۱ : Begin

دستور ۱ ;

دستور۲ ;

.
.
.

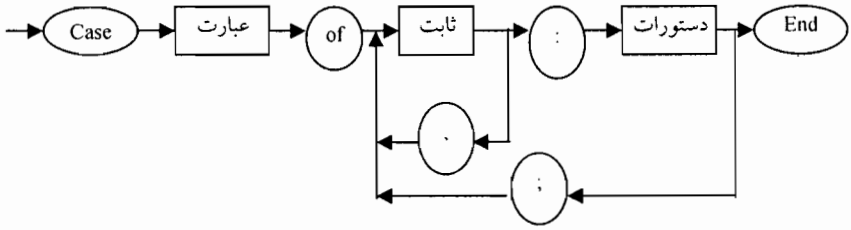
End ;

.
.
.

End ; { End of Case }

دستور Case در حالت اول مقدار عبارت را با یکی از مقادیر حالت اول بررسی می‌کند اگر یکی از مقادیر با مقدار عبارت برابر باشد، این حالت اجرا می‌شود. دستور Case در حالت ۲ زمانی بکار می‌رود که بعد از برقراری شرط چند دستور اجرا شوند. در اینصورت همانطور که ملاحظه می‌کنید مجموعه دستورات داخل یک بلوک (بین Begin و End) قرار می‌گیرند.

دیاگرام دستور Case بصورت زیر می‌باشد:



شکل ۳-۷ دیاگرام دستور case

مثال ۱۴-۷ برنامه‌ای بنویسید که بر حسب زوج یا فرد بودن مقدار یک متغیر صحیح که مقادیر آن می‌تواند در محدوده ۰ تا ۱۰ باشد، پیام مناسبی چاپ کند.

```
Program          Example14 ;
 Var
                 N : integer ;
Begin
                 Writeln ;
                 Writeln ( ' Enter An Integer In Range 0 .. 10 ' ) ;
                 Readln ( N ) ;
                 Case N of
                        0,2,4,6,8 : Writeln ( ' Even ' ) ;
                        1,3,5,7,9 : Writeln ( ' odd ' ) ;
                  end ;
End . { end of program }
```

خروجی برنامه بالا:

```
Enter An Integer In Range 0 ... 10
4
Even
```

نکته: در این ساختار، برخلاف دستور if – then – else قبل از کلمه else می‌توان از علامت سمیکالن (;) استفاده کرد.

۷-۲ ساختارهای کنترلی

بسیاری از مواقع لازم است عمل یا اعمال به تعداد دفعات معین یا نامعین انجام شوند. در چنین مواقعی زبانهای برنامه‌نویسی دستوراتی دارند که می‌توان این اعمال تکراری را انجام داد. در حالت کلی ساختارهای کنترلی شامل یک یا چند شرط و همچنین متغیر یا اصطلاحاً شمارنده‌ای برای پایان دادن به شرط می‌باشند. در زیر انواع ساختارهای کنترلی را بررسی می‌کنیم.

۷-۲-۱- حلقه for

این دستور برای انجام عمل یا اعمالی مشخص به تعداد تکرار معین بکار برده می‌شود. حلقه for شامل یک اندیس (index) مقدار اولیه (initial value) مقدار نهایی (final value) و مقدار افزاینده می‌باشد. این حلقه با قرار دادن مقدار اولیه در اندیس حلقه شروع شده و بعد از هر تکرار یک واحد به اندیس حلقه اضافه می‌کند تا در نهایت به مقدار نهایی برسد. شکل کلی حلقه بصورت زیر می‌باشد:

For اندیس := مقدار اولیه To مقدار نهایی Do

دستور ;

باید توجه داشته باشید که در حلقه for تعداد تکرار کاملاً مشخص است و حلقه دقیقاً به تعداد تکرار مشخص اجرا می‌شود.

مثال ۷-۱۵ خروجی قطعه برنامه زیر را تعیین کنید.

```
for i = 1 to 5 do
    writeln ( 'pascal' ) ;
```

بعد از اجرای این دستور ۵ بار رشته pascal در سطرهای جداگانه چاپ می‌شود.

مثال ۷-۱۶ قطعه برنامه زیر مجموع اعداد ۱ تا ۱۰۰ را محاسبه می‌کند.

```
Var
   Sum , i: integer ;
Begin
   For i:= 1 to 100 do
       Sum:= Sum + i ;
   Writeln ( 'Sum = ', Sum ) ;
End.
```


نکته: اگر بیش از یک دستور در حلقه For قرار گیرد باید بین یک بلوک قرار گیرد (End , Begin).

مثال ۷-۱۷ برنامه زیر کاربرد حلقه را نشان می‌دهد:

```
Var
   i: integer ;
Begin
   For i:= 1 to 10 do
       Writeln ( 'i = ', i: 2 ) ;
End.
```

برنامه بالا دستور چاپ را تا زمانیکه i یعنی اندیس حلقه به 10 نرسیده انجام می‌دهد وقتی i به 10 رسید، حلقه برای آخرین بار اجرا می شود لذا خروجی برنامه بالا بصورت زیر خواهد بود:

```
i = 1
i = 2
.
.
.
i = 9
i = 10
```

مثال ۷-۱۸ برنامه‌ای بنویسید که ۱۰۰ عدد از ورودی دریافت کرده، مجموع ۱۰۰ عدد را محاسبه و چاپ نماید.

```
Program     Example18 ;
Var
        i , number , Sum: integer ;
Begin
    Writeln ( 'Please enter 100 numbers: ' ) ;
    for i:= 1 to 100 do
        Begin
            Readln ( number ) ;
            Sum:= Sum + number ;
        End ;
    Writeln ( 'Sum = ', Sum ) ;
End. { end of program }
```

برنامه بالا ۱۰۰ عدد مختلف را از ورودی دریافت می‌کند و مجموع آنها را محاسبه می‌کند.

مثال ۱۹ ۷ برنامه‌ای بنویسید که ۱۰۰ عدد از ورودی دریافت کرده، اعداد زوج را از بین این اعداد تشخیص داده در خروجی با پیغام مناسب چاپ نماید.

```
Program          Example19 ;
Var
                   i , a: integer ;
Begin
                Writeln ( 'Please enter 100 numbers: ' ) ;
                for  I: = 1   to   100   do
                    Begin
                          Readln ( a ) ;
                          if ( a  mod  2 ) = 0   Then
                                Writeln ( ' The number is even ' ) ;
                    End ; { end of for }
End. { end of program }
```

برنامه بالا ۱۰۰ عدد از ورودی دریافت می‌کند، بعد از خواندن هر عدد، اگر عدد زوج باشد پیغام The number is even چاپ می‌شود و اگر عدد زوج نباشد پیغامی چاپ نمی‌شود.

سوالی که در مورد اندیس حلقه می توان مطرح کرد اینست که آیا می توان داخل یک حلقه مقدار اندیس حلقه را تغییر داد یا نه؟

بطور صریح می توان گفت که تغییر مقدار اندیس حلقه درون حلقه for کار خطرناکی است و موجب اشتباه می‌شود. دستور for زیر را در نظر بگیرید:

```
For i: = 1 to 10 do
Begin
   Writeln ( ' i =   , i ) ;
   i: = 2 ;
end ;
```

در قطعه بالا نخست i برابر 1 قرار داده می‌شود و مقدار در خروجی چاپ می‌شود پس از آن i برابر 2 قرار داده می‌شود و بعد از آن i یک واحد افزایش یافته با مقدار نهایی مقایسه می‌شود. در مرحله دوم i = 3 چاپ می شود سپس بار دیگر برابر 2 قرار داده می‌شود و این روند دوباره ادامه پیدا می‌کند. اگر دقت کنید اینکار هرگز پایان نمی‌پذیرد و اصطلاحاً یک حلقه بی‌پایان (حلقه بینهایت) حاصل می‌شود. لذا برای دوری از این نوع حلقه ها سعی می‌کنند، مقدار اندیس حلقه را تغییر ندهند.

مثال ۲۰-۷ خروجی برنامه زیر را مشخص کنید.


```
Program          Example20 ;
Var
      i : integer ;
Begin
      For  I: = 1   to   100   do
         Begin
           Writeln ( ' I   ', I ) ;
           Read ( i ) ;
         End ;
End.
```

بعنوان ورودی از اعداد 12 9 6 3 استفاده کنید.

خروجی حاصل از اجرای این برنامه بصورت زیر خواهد بود:

```
I = 1
I = 4
I = 7
I = 10
I= 13
```

به عنوان تمرین اجرا را دنبال کنید ؟

دستور for را بصورت زیر هم می‌توان بکار برد:

```
for  اندیس : = مقدار اولیه downto مقدار نهایی do
   دستور ;
```

در این شکل از دستور For ابتدا مقدار اولیه در اندیس حلقه قرار داده می شود و بعد از آن در هر تکرار حلقه یک واحد از اندیس حلقه کم می شود تا به مقدار نهایی برسد به مثال زیر توجه کنید:

```
for        i : = 5        downto     1        do
             Writeln ( ' I = ', I ) ;
```

خروجی دستور بالا بصورت زیر است:

```
I = 5
I = 4
I = 3
I = 2
I = 1
```

مثال ۲۱-۷ برنامه‌ای بنویسید که عدد صحیح n را از ورودی دریافت کرده و فاکتوریل آن را محاسبه نماید.

```
Program  Example21 ;
Var
     i , n , Fact : integer ;
Begin
      Fact : = 1 ;
     Write ( 'Please  enter  a  number ' ) ;
     Readln ( n ) ;
     For   I : = n     downto    1    do
          Fact : = Fact  *  i ;
     Writeln ( ' Fact = ', Fact ) ;
End .
```

خروجی برنامه بالا با مقدار n = 4 بصورت زیر است:

```
Please enter  A  number: 4
Fact = 24
```

مثال ۲۲-۷ برنامه‌ای بنویسید که ۵ عدد از ورودی دریافت کرده و سپس تعداد اعداد فرد و زوج را شمرده در خروجی چاپ کند.

```
Program          Example22 ;
Var
                  i , even , odd , number: integer ;
Begin
                 Writeln ( ' please  enter  Ten  Numbers ' ) ;
                  Even: = 0 ;  odd: = 0 ;
                  For   I: = 1      to    5    do
                     Begin
                              Read ( number ) ;
                               If ( number  mod  2 ) = 0  Then
                                   Even: = Even + 1
                               Else
                                        Odd: = odd + 1 ;
                        End ; { end  of  For }
                     Writeln ( 'Number  of  even = ' , even ) ;
                     Writeln ( 'Number  of  odd = ' , odd ) ;
End. { end  of  program }
```

خروجی برنامه بالا با ورودیهایی:

```
2     4     5     7     6
```

بصورت زیر می باشد:

```
Number  of  even = 3
Number  of  odd = 2
```


حلقه‌های For می توانند بصورت تودرتو هم بکار برده شوند که اصطلاحاً حلقه‌های تودرتو یا متداخل نامیده می‌شوند. شکل بکارگیری حلقه‌های تودرتو بصورت زیر می‌باشد:

```
for  اندیس1 : = مقدار اولیه  to  مقدار نهایی  do
for 2اندیس : = مقدار اولیه  to  مقدار نهایی  do
     دستور  ;
```

در این نوع حلقه های for به ازای یک مقدار حلقه اول، حلقه داخلی بطور کامل اجرا می‌شود. در مرحله بعدی نیز دوباره حلقه داخلی بطور کامل اجرا می‌شود. تا زمانیکه اندیس حلقه اول به مقدار نهایی نرسیده، حلقه داخلی بطور کامل اجرا می‌شود.

به مثال زیر توجه کنید:

```
for   i : = 1    to   2    do
   for    j : = 1    to   3   do
          Writeln ( ' Pascal' ) ;
```

خروجی دستورات بالا بصورت زیر می باشد:

```
مرحله اول (I = 1)
    Pascal
    Pascal
    Pascal

(I = 2)
    Pascal
    Pascal
    Pascal
```

مثال ۲۳-۷ خروجی قطعه برنامه زیر را تعیین کنید.

```
For   i: = 1      to   3    do
Begin
          For   j: = 1      to   3    do
                Write ( ' pascal ': 8 ) ;
          Writeln ;
 End ;
```

خروجی قطعه برنامه بالا:

(i = 1) مرحله اول Pascal Pascal Pascal

(i = 2) مرحله دوم Pascal Pascal Pascal

(i = 3) مرحله سوم Pascal Pascal Pascal

همانطور که ملاحظه می‌کنید ابتدا I برابر ۱ قرار داده می‌شود و حلقه دوم به اندازه سه بار اجرا می‌شود و رشته پاسکال را سه بار در یک سطر چاپ می‌کند، سپس دستور Writeln اجرا می‌شود و به I یک واحد اضافه می‌شود و دوباره روند بالا ادامه پیدا می‌کند.

مثال ۲۴-۷ خروجی برنامه زیر را تعیین کنید.

```
Program          Example24 ;
Var
          i , j: integer ;
Begin
          Writeln ;
          For   I: = 1     to    4    do
             Begin
               For   j: = 1     to    i    do
                        Write ( j: 3 ) ;
               Writeln ;
             End ; { end of For }
End. { end of program }
```

خروجی:

```
1
1   2
1   2   3
1   2   3   4
```

نکته: باید توجه داشته باشید که در حلقه‌های تودرتو، اندیس حلقه‌ها نباید دارای اندیسهای هم اسم باشند، در صورت اشتباه ممکن است حلقه بی‌نهایت یا اشکالات دیگری حاصل شود.

مثال ۲۵-۷ برنامه‌ای بنویسید که یک جدول ضرب ۱۰×۱۰ را در صفحه نمایش چاپ کند.


```
Program          Example25 ;
Var
          i , j: integer ;
Begin
          Writeln ;
          For   I: = 1     to    10    do
          Begin
               For   j: = 1     to    10    do
                        Write ( i * j: 4 ) ;
               Writeln ;
          End ; { end of For }
End.
```

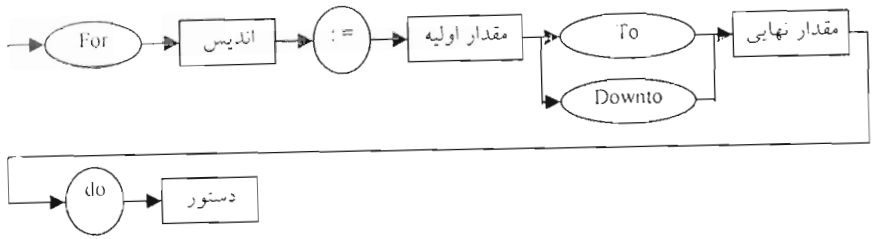
برنامه بالا جدول ضرب (۱۰×۱۰) را در صفحه نمایش چاپ می‌کند.

مثال ۲۶-۷ برنامه‌ای بنویسید که n عدد از ورودی دریافت کرده بیشترین مقدار از بین n عدد را پیدا کرده در خروجی چاپ کند.

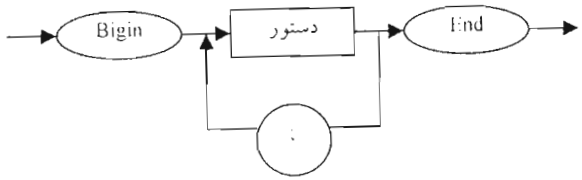
```
Program          Example26 ;
Var
          i , Number , N , Max: integer ;
Begin
          Write ( ' Please Enter Number of Information ' ) ;
          Readln ( N ) ;
          Writeln ( ' Please Enter ' , N: 4 , ' Numbers ' ) ;
          Readln ( Number ) ;
          Max: = Number ;
          For   i: = 1   to    N - 1    do
          Begin
               Readln ( Number ) ;
               if   Max < Number   Then
                    Max: = Number ;
          End ; { end of for }
          Writeln ( ' Max = ' , Max ) ;
End. { end of program }
```

در برنامه بالا نخست تعداد ورودیها را مشخص می‌کنیم بعد از آن اولین ورودی را بیشترین مقدار گرفته ورودیهای دیگر را می‌خوانیم، در صورتی که ورودی جدید از بیشترین مقداری که تا این لحظه حاصل شده، بیشتر باشد آن را بیشترین مقدار قرار می‌دهیم. این روند را تا زمانیکه همه ورودیها را نخوانده‌ایم ادامه می‌دهیم. در نهایت بیشترین مقدار بین ورودیها تعیین می‌شود.

دیاگرام دستور for در حالت کلی بصورت زیر می‌باشد.



و اگر دستور مرکب باشد قسمت دستور داخل یک بلوک قرار می‌گیرد:



شکل ۴-۷ دیاگرام دستور for

۲-۷-۲- حلقه While

همانطور که مشاهده کردید، در حلقه های For تعداد تکرار مشخص می‌باشد. اگر تعداد تکرار مشخص نباشد از حلقه While استفاده می‌کنیم. در حالت کلی هدف از بکار بردن این دستور انجام عملیاتی مشخص به تعداد دفعات نامعین است. دستور While به شکلهای زیر بکار برده می‌شود:

1) While عبارت منطقی Do
   دستور ;

یا در حالت کلی:

```
While  عبارت منطقی   Do
  Begin
      دستور1 ;
      دستور2 ;        } بدنه حلقه
        .
        .
End ;
```


در دستور While ابتدا شرط یا شروط حلقه بررسی می‌شود. در صورتی که شرط برقرار باشد کنترل برنامه به بدنه حلقه منتقل می‌شود و تا زمانیکه شرط برقرار باشد، حلقه اجرا می‌شود. باید توجه داشته باشیم که در بدنه حلقه باید دستور یا دستوراتی باشند که شرط را تغییر دهند در غیر اینصورت یک حلقه بینهایت حاصل می‌شود.

در این نوع حلقه اندیس حلقه وجود ندارد و حلقه تنها با شرط کار می‌کند. حلقه تا زمانیکه شرط برقرار باشد اجرا می‌شود.

قطعه برنامه زیر را در نظر بگیرید:

```
X: = 0 ;
Y: = 1 ;
While     x < 5     do
   x: = y + 1 ;
```

اگر به قطعه برنامه بالا دقت کنید خواهید دید که تا زمانیکه x از 5 کمتر باشد، حلقه ادامه پیدا می کند یعنی تا زمانیکه ارزش شرط حلقه True باشد حلقه تکرار می‌شود.

نکته: توجه داشته باشید که در حلقه While بعد از کلمه Do مجاز نیستید از علامت سمیکالن (;) استفاده کنید.

با این حلقه می توان حلقه For را نیز شبیه سازی نمود. (به نوعی می توان اینکار را انجام داد)

بطور مثال حلقه For زیر را در نظر بگیرید:

```
for i : = 1   to    5  do
   Write ( ' PASCAL ' , 8 ) ;
```

این قطعه برنامه با حلقه While بصورت:

```
I: - 1 ;
While ( I < = 5 )    do
Begin
        Write ( ' PASCAL ': 8 ) :
        I: = I + 1 ;
End ;
```

می‌باشد.

تا زمانیکه I به نصف عدد نرسیده حلقه اجرا می شود بعد از اتمام تکرار حلقه مقدار flag بررسی می‌شود و پیغام No prime در خروجی چاپ می‌شود.

اگر در برنامه بالا دقت کنید، وقتی ورودی برنامه عدد اول نباشد حلقه While بیش از حد لازم اجرا می‌شود. اصولاً بعد از اینکه شرط if داخل حلقه یکبار برقرار باشد نباید حلقه تکرار شود. برنامه بالا را می‌توان بصورت زیر بهینه کرد:

```
Program        Example28 ;
Var
               Number , i : integer ;
               Flag: Boolean ;
Begin
        Write ( ' Please Enter A Number: ' ) ;
        Readln ( Number ) ;
        I: = 2 ;
        Flag: = True ;
        While ( i < = ( Number div 2 ) ) and ( flag = True )   do
         begin
           If ( Number Mod i ) = 0  Then
             Flag: = false ;
           i: = i + 1 ;
         end ;
       If  flag = Ture Then
          Write ( ' prime ' )
       Else
          Write ( ' No prime ' ) ;
End.
```

برنامه بالا را می‌توان بصورت ساده زیر نوشت:

```
Program        Example28 ;
Var
               Number , i : integer ;
               Flag: Boolean ;
Begin
                Write ( ' Please Enter A Number: ' ) ;
                Readln ( Number ) ;
                I: = 2 ;
                Flag: = True ;
                While ( ( i < = ( Number div 2 ) ) and flag ) do
                   begin
                      If ( Number Mod i ) = 0  Then
                            Flag: = false ;
                      i: = i + 1 ;
                  end ;{ End of while }
                If  flag Then
                     Write ( ' prime ' )
                Else
```

مثال ۷-۲۷ برنامه‌ای بنویسید که دو عدد صحیح m , n (m > n) از ورودی دریافت کرده و بزرگترین مقسوم علیه مشترک دو عدد را محاسبه کرده، در خروجی چاپ نماید.

```
Program        Example27 ;
Var
               m , n , r: integer ;
Begin
                  Writeln ( ' Please Enter Two Numbers ' ) ;
                  Readln ( m , n ) ;
                  While ( m  Mod  n ) < > 0  Do
                  begin
                        r: = m  MOD  n ;
                        m: = n ;
                        n: = r ;
                  end ; { end of while }
                  Writeln ( ' B. M. M = ' , n ) ;
End. { end of program }
```

مثال ۷-۲۸ برنامه‌ای بنویسید که یک عدد از ورودی دریافت کرده، اول بودن آن را تشخیص دهد.

```
Program          Example28 ;
Var
               Number , i : integer ;
                Flag: Boolean ;
Begin
              Write ( ' Please Enter A Number: ' ) ;
              Readln ( Number ) ;
              I: = 2 ; flag: = Ture ;
              While i < = ( Number div 2 )  Do
                Begin
                  If ( Number Mod I ) = 0 Then
                        Flag: = False ;
                        i: = i + 1 ;
                  end ;
              If  flag = Ture Then
                  Write ( ' prime ' )
              Else
                 Write ( ' No prime ' ) ;
End. { end of program }
```

در برنامه بالا فرض کنید عدد 12 بعنوان ورودی برنامه باشد در اینصورت در مرحله اول اجرا شرط if برقرار می‌شود بنابراین مقدار متغیر flag برابر false می‌شود.

```
        Write ( ' No  prime ' ) ;
End.{ end  of  program }
```

مثال ۲۹-۷ برنامه‌ای بنویسید که تعدادی عدد از ورودی دریافت کرده، تعداد اعداد زوج از بین اعداد خوانده شده، را شمرده و در خروجی چاپ نماید (پایان داده‌ها به عدد منفی ختم می‌شود).

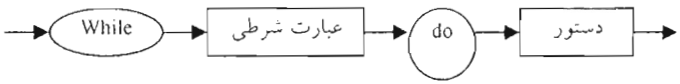
```
Program         Example29 ;
Var
                Number , count : integer ;
Begin
                Count: = 0 ;
                Writeln ( ' Please  Enter  Numbers  while  - 1: ' ) ;
                 Readln  ( Number ) ;
                 While  Number  > = 0  do
                 begin
                        If ( Number  Mod  2 )  = 0  Then
                            Count: = count + 1 ;
                        Readln ( Number ) ;
                 end ;
                 Writeln ( ' Number  of  even  numbers = ', count ) ;
End.{ end  of  program }
```

این برنامه اعداد را تا زمانیکه عدد منفی وارد نکردیم، از ورودی دریافت می‌کند و همزمان تعداد اعداد زوج را می‌شمارد.

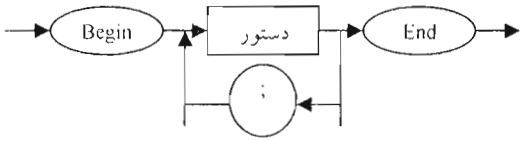
مثال ۳۰-۷ برنامه‌ای بنویسید که یک جمله را از ورودی دریافت کرده تعداد کاراکتر t را بشمارد. (جمله به نقطه '.' ختم می شود)

```
Program         Example30 ;
Var
                count : integer ;
                 Ch: Char  ;
Begin
                Writeln ( ' Please  Enter  Statement ' ) ;
                 Read ( Ch ) ;
                 While  ch < > '.'  do
                 begin
                        If ( ch = 't' )  or  ( ch = 'T' )  Then
                            Count: = count + 1 ;
                        Readln ( ch ) ;
                 end ;
                 Writeln ( ' Number  of  characters = ', count ) ;
End.{ end  of  program }
```


دیاگرام دستور While بصورت زیر می باشد:



و در صورتی که دستور مرکب باشد دیاگرام بصورت است.



شکل ۵-۷ دیاگرام دستور while

۳-۲-۷- دستور Repeat

این دستور نیز از نوع دستورات تکراری می‌باشد و به کمک آن می‌توان یک یا چند دستور را به تعداد نامعین اجرا کرد. در دستور repeat تکرار تا زمانی ادامه خواهد داشت که شرط خاصی تحقق پیدا کند. این دستور مشابه دستور while است، با تفاوتهایی که در زیر عنوان می‌کنیم:

۱. در دستور Repeat برعکس دستور While شرط حلقه در انتهای حلقه بررسی می‌شود لذا حلقه حداقل یکبار اجرا می‌شود.
۲. دستور Repeate تا زمانی اجرا می‌شود که شرط خاصی تحقق پیدا نکرده است در حالیکه دستور While تا زمانی که شرط برقرار باشد، اجرا می‌شود.
۳. دستور Repeat نیاز به بلوک ندارد و همراه Until ظاهر می‌شود.

شکل کلی این دستور بصورت زیر می‌باشد:

```
Repeat
دستور1 ;
دستور2 ;
   .
   .
until    شرط یا شروط   ;
```

مثال ۳۱-۷ برنامه‌ای بنویسید که مجموع اعداد ۱ تا ۱۰۰ را محاسبه و چاپ نماید.

۷-۳ معرفی چند پروسیجر (Procedure)

پروسیجرها یا روال‌ها قسمت‌های مستقلی از برنامه اصلی می‌باشند که به تنهایی اعمال خاصی را انجام داده و وظایف مستقل و بخصوصی بر عهده آنها گذاشته می‌شود. یک مزیت بزرگ پروسیجرها اینست که یکبار در برنامه گنجانده شده ولی در محلهای مختلف ازآن استفاده به عمل می‌آید و از اصول برنامه‌نویسی ساخت یافته‌است.

باید توجه داشته باشید هر زبان خود یک سری زیربرنامه نوشته شده دارد که در برنامه می‌توان از آنها استفاده کرد. در زبان پاسکال نیز تعدادی توابع (Function) و روال وجود دارد که به معرفی تعدادی از آنها که می‌توان در حلقه‌ها از آنها استفاده کرد، می‌پردازیم.

۷-۳-۱- پروسیجر Exit

هدف: انتقال کنترل برنامه به خارج از بلوک فعلی

```
Procedure  Exit ;
```

استفاده از این پروسیجر در هر بلوک از برنامه باعث می‌شود که کنترل برنامه بلافاصله به خارج از آن بلوک انتقال یابد. پروسیجر Exit اگر در بدنه اصلی بلوک یک پروسیجر یا تابع قرارگرفته باشد، باعث خروج از پروسیجر یا تابع شده وکنترل برنامه‌به اولین دستوری که بلافاصله پس ازدستور فراخوانی آن پروسیجر یا تابع قرارگرفته پرش می‌کند. اگر پروسیجر Exit در بلوک اصلی برنامه باشد، برنامه بلافاصله خاتمه می‌یابد.

۷-۳-۲- پروسیجر Break

هدف: خاتمه دادن به اجرای یک حلقه

```
Procedure  Break ;
```

استفاده از پروسیجر فوق باعث می‌شود که اجرای یک حلقه خاتمه یافته و کنترل برنامه به دستورالعمل بعدی انتقال یابد. برای مثال قطعه برنامه زیر را در نظر بگیرید:

```
i := 1 ;
While   i <= 10    do
    Begin
        if ( i  mod  2 ) = 0    then
                Break ;
        i := i + 1 ;
    End ;
```

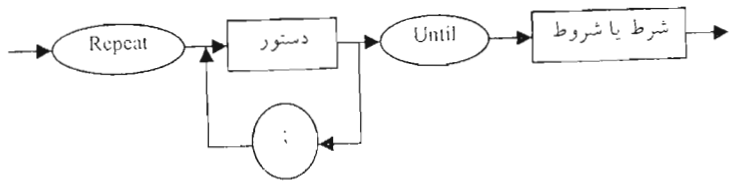
در برنامه بالا اولین بار i برابر 1 قرار داده می‌شود چون شرط if داخل حلقه

```
Program        Example31 ;
Var
               i , Sum : integer ;
Begin
                  i: = 1 ;
                  Sum: = 0 ;
                  Repeat
                        Sum: = sum + i ;
                        i: = i + 1 ;
                   Until  i > 100 ;  { End  of  Repeat }
                  Writeln ( ' Sum = ' , Sum ) ;
end.{ end  of  program }
```

مثال ۷-۳۲ برنامه‌ای بنویسید که تعدادی عدد صحیح مثبت را از ورودی دریافت کرده، مجموع و میانگین آنها را محاسبه و چاپ نماید.

```
Program        Example32 ;
Var
               i , Sum , Number : integer ;
               ave: Real ;
Begin
               Writeln ( 'Please  enter  numbers (while  is  Not  negative) ' ) ;
               Sum: = 0 ;
               Ave: = 0 ;
               Repeat
                     Sum: = sum + Number ;
                     i: = i + 1 ;
                     Readln ( Number ) ;
                Until  number < 0 ;  { End  of  Repeat }
                i: = i - 1
                ave: = Sum / i ;
                Writeln ( ' Sum = ' , Sum , ' average = ': 12 , ave: 7: 2 ) ;
end.{ end  of  program }
```

دیاگرام دستور Repeat بصورت زیر می‌باشد:



شکل ۷-۶ دیاگرام دستور repeat

برقرار نیست i برابر 2 قرار می‌گیرد در مرحله دوم چون شرط if داخل حلقه برقرار است، دستور Break اجرا شود و کار حلقه خاتمه می‌یابد یا اصطلاحاً حلقه می‌شکند.
اگر در خارج از حلقه مقدار i را چاپ کنیم مقدار 2 را نشان خواهد داد.

۷-۳-۳- پروسیجر continue

هدف: بازگشت به ابتدای حلقه

```
Procedure continue ;
```

وقتی این پروسیجر در حلقه ظاهر می‌شود کنترل برنامه به اول حلقه انتقال می‌یابد و دستورات بعد از پروسیجر اجرا نمی‌شوند.

۷-۴- ارائه چند مثال از کاربرد حلقه ها و شرط‌ها

حال تعدادی مثال از کاربرد مطالب گفته شده در این فصل ارائه می‌دهیم:

مثال ۷-۳۳ برنامه‌ای بنویسید که n عدد صحیح از ورودی دریافت کرده، بیشترین مقدار و تعداد آن را یافته در خروجی چاپ کند.

```
Program        Example33 ;
Var
              Max_Count , N , Number , i: integer ;
Begin
               Write ( 'please enter Number of data: ' ) ;
               Readln ( N ) ;
               Writeln ( 'Please enter numbers' ) ;
               Readln ( Number ) ;
               Max: = Number ;
               Max_Count: = 1 ;
               For   i: = 2   to   N   do
               Begin
                      Readln ( Number ) ;
                      If Max < Number Then
                        Begin
                              Max: = Number ;
                              Max_Count: = 1 ;
                         End
                      Else if Max = Number Then
                            Max_Count: = Max_Count + 1 ;
              End ; { end of for }
              Writeln ;
              Writeln ( 'Max =', Max , 'Max count': 12 , Max_Count ) ;
              Readln ;
End. { end of program }
```


مثال ۷-۳۴ برنامه‌ای بنویسید که n عدد را از ورودی دریافت کرده، n جمله سری زیر که به سری فیبوناچی معروف است را چاپ نماید.

0 1 1 2 3 5 8 13 .....

توجه کنید که در این سری هر جمله مجموع دو جمله قبل می باشد و در حالت کلی:
$F_n = F_{n-1} + F_{n-2}$

می‌باشد.

```
Program        Example34 ;
Var
               F1 , F2 , F3 , N: integer ;
Begin
               Write ( 'Please enter a number: ' ) ;
               Readln ( N ) ;
               F1: = 0 ; F2: = 1 ;
               Write ( F1: 5 , F2: 5 ) ;
               For   i: = 3   to   N   do
                 Begin
                        F3: = F1 + F2;
                        If ( i mod 10 ) = 0 Then
                             Writeln ;
                        Write ( F3: 5 ) ;
                        F1: = F2 ;
                        F2: = F3 ;
                End ; { end of for }
               Writeln ;
               Writeln ( 'Press any Key ....' ) ;
               Readln ;
End. { end of program }
```

برنامه بالا جملات سری را در سطرهای مختلف چاپ می‌کند (در هر سطر ۱۰ جمله).

مثال ۷-۳۵ برنامه‌ای بنویسید که جمله ای از ورودی دریافت کرده، تعداد کلمه‌های جمله را شمرده و در خروجی چاپ کند.
با فرض اینکه بین کلمه ها فقط یک فاصله وجود دارد و جمله به نقطه ختم می‌شود.

```
Enter An Integer Number: 125
                5
                2
                1
enter An Integer Number : 2171
                1
                7
                1
                2
```

مثال ۳۷-۷ برنامه‌ای بنویسید که یک عدد صحیح در مبنای ده را از ورودی دریافت کرده، به یک عدد در مبنای ۲ ببرد.

```
program         Example37 ;
Var
                Number , N , Power , R: integer ;
Begin
                Power: = 1 ;
                N: = 0 ;
                Write ( ' enter A Number: ' ) ;
                Readln ( Number ) ;
                Repeat
                      R: = Number MOD 2 ;
                      Number: = Number DIV 2 ;
                      N: = N + Power * R ;
                      Power: = Power * 10 ;
                Until Number < 2 ;
                N: = N + Number * Power ;
                Writeln ( ' Number In Base 2 = ', N ) ;
End. { end of program }
```



```
Program         Example35 ;
Var
                Word_Count : integer ;
                Ch: Char ;
Begin
                Writeln ( ' Please enter sentence ' ) ;
                Word_Count: = 0 ;
                Repeat
                      Read ( ch ) ;
                      If ( ch = ' ' ) and ( ch <> '.' ) Then
                          Word_Count: = Word_Count + 1 ;
                Until ch = '.' ;
                Writeln ;
                Writeln ( ' Word Count = ' , Word_Count ) ;
End.
```

خروجی قطعه برنامه بالا با داده های زیر:

```
Please enter sentence
This book is very useful for student.
```

بصورت :

```
Word count = 7
```

خواهد بود.

مثال ۳۶-۷ برنامه‌ای بنویسید که تعدادی عدد صحیح را از ورودی دریافت کرده، ارقام اعداد را تک تک در خروجی چاپ کند. (شرط پایان داده عدد ۱- است)

```
program         Example36 ;
Var
                Number: integer ;
Begin
                Repeat
                      Write ( ' Enter An Integer Number: ' ) ;
                      Readln ( Number ) ;
                      While Number > 0 Do
                          Begin
                              Writeln ( ' ': 12 , Number Mod 10 ) ;
                              Number: = Number DIV 10 ;
                          End ;
                Until Number = -1 ;
End. { end of program }
```

خروجی حاصل از اجرای این برنامه بصورت زیر است:

۵-۷- تمرینات

۱- خروجی قطعه برنامه‌های زیر را تعیین کنید:

الف)

```
i : = 0 ;
Sum : = 0 ;
While   i < = 120   do
Begin
        Sum : = Sum + i  ;
        i : = i + 1  ;
end ;
```

ب)

```
 i : = 0 ;
Sum : = 0 ;
While   i < = 20   do
Begin
            i : = i + 1 ;
           Sum : = Sum + i ;
end ;
```

ج)

```
for   i : = 1   to   10   do
     Writeln ( i = i + 1 ) ;
```

د)

```
N : = 2173 ;
i : = 0 ;
While N > 0 do  begin
      N : = N   div   10 ;
       i : = i + 1 ;
end ;  Write ( i ) ;
```

ح)

```
  b : = 5 ;
Repeat
   Writeln ( b , ( b  div  5 ) : 3 ) ;
    b : = b - 1  ;
Until ( b  div  3 ) = 5 ;
```

خ)

```
Count  : = 0 ;
Stop : = 4 ;
While  Count < Stop  Do
Begin
      For   K : = 1   to   Count   Do
                Write ( K : 3 ) ;
         Writeln ;
         Count : = Count + 1 ;
End ;
```


۲- حلقه‌های زیر را در نظرگرفته، با ورودی های مختلف آنها را آزمایش کنید. همچنین حلقه بینهایت ایجاد شده را مورد بررسی قرار دهید.

الف)

```
N : = 13  ;  i : = 2  ;
   While   N < > 0   do
 Begin
i : = i + 1  ;
N : = N - i  ;
   Write ( ' N = ' , N , ' i = ' , i ) ;
End  ;
```

ب) برای مقادیر مختلف n

```
Read ( n ) ;
For   i : = 1   to   n   do
Begin
     i : = i + 2  ;
     Writeln ( ' i = ' , i ) ;
End;
```

ج)

```
For   i : = 5   to   13   do
Begin
       Write ( ' i = ' , i ) ;
        i : = i + B  ;
end  ;
```

۳- در حلقه‌ها While , For بعد از کلمه do مجاز به استفاده از سمیکالن (;) نیستیم. بررسی کنید که آیا در صورتی که علامت سمیکالن در انتهای کلمه do دستورات بالا قرار داده شود، چه اتفاقی رخ می‌دهد بطور مثال حلقه های زیر را بررسی کنید:

الف)

```
For   i : = 1   to   100   do ;
    Writeln ( ' TANHA ' ) ;
```

ب)

```
i : = 1  ;
While ( i < = 100 ) do  ;
 Begin
       Write ( ' i  = ' , i ) ;
        i : = i + 2  ;
end  ;
```

۶-۷- تمرینات برنامه‌نویسی

۱- برنامه‌ای بنویسید که با استفاده از حلقه ها خروجی زیر را تولید کند.

```
                1
              1 2 1
            1 2 3 2 1
          1 2 3 4 3 2 1
        1 2 3 4 5 4 3 2 1
```

۲- برنامه‌ای بنویسید که تعدادی عدد از ورودی دریافت کرده مجموع ارقام هر عدد را در خروجی چاپ نماید. (پایان داده ها به ۱- ختم می شود)

۳- برنامه‌ای بنویسید که دو عدد صحیح N , M را از ورودی دریافت کرده سپس:

الف) اعداد فیبوناچی بین این دو عدد را چاپ کند.

ب) K را از ورودی دریافت کرده و اولین K عددفیبوناچی بین N , M را چاپ کند.

ج) اعداد فیبوناچی زوج و فرد بین N , M را از یکدیگر جدا کرده در سطرهای جداگانه چاپ نماید.

۴- برنامه‌ای بنویسید که یک اسکناس ۱۰۰۰ تومانی را به حالتهای مختلف یعنی به اسکناس ۲۰۰ تومانی، ۱۰۰ تومانی، ۵۰ تومانی، ۲۰ تومانی، ۱۰ تومانی و سکه های ۵ تومانی و ۲ تومانی و یک تومانی خرد نماید.

۵- برنامه‌ای بنویسید که یک عدد در مبنای ۲ از ورودی دریافت کرده سپس آنرا به مبنای ۱۰ برده و چاپ نماید.

۶- برنامه‌ای بنویسید که یک عدد صحیح را از ورودی دریافت نموده مقلوب آن را محاسبه و در خروجی چاپ نماید. توجه کنید مقلوب یک عدد برعکس شده یک عدد می باشد برای مثال مقلوب عدد ۴۷۳ برابر ۳۷۴ می باشد. برنامه را طوری تغییر دهید که برای اعداد اعشاری نیز بصورت زیر کار کند:

عدد 127.25

مقلوب عدد 52.721

۷- برنامه ای بنویسید که تعدادی عدد از ورودی دریافت کرده سپس:

الف) تعداد اعداد اول بین این اعداد را شمرده در خروجی چاپ نماید.

ب) تعداد اعداد کامل بین این اعداد را شمرده در خروجی چاپ کند (منظور از عدد

کامل اینست که مجموع مقسوم علیه های آن با خود عدد برابر باشد مثل $(1+2+3=6)$

پایان داده ها به عدد ۱ ختم می شود.

۸- برنامه‌ای بنویسید که n عدد صحیح از ورودی دریافت که بیشترین مقدار و کمترین مقدار و تعداد تکرار آنها را محاسبه و در خروجی چاپ نماید.

۹- برنامه‌ای بنویسید که یک عدد از ورودی خواند و سپس در صورتی که رقم صفری در عدد وجود داشته باشد آنرا حذف کرده و عدد حاصل را خروجی چاپ کند.

فصل ۸

آرایه‌ها (Arrays)

هدفهای کلی

- شناخت لزوم استفاده از ساختار داده‌ای به نام آرایه
- شناخت انواع آرایه‌ها و موارد استفاده از آنها
- شناخت مفاهیم مرتب‌سازی و جستجو
- شناخت الگوریتمهای مرتب‌سازی و جستجو

هدفهای رفتاری

دانشجو پس از مطالعه این فصل باید بتواند:

- آرایه‌ها یک بعدی را در برنامه‌های خود بکار ببرد.
- ماتریسها را پیاده‌سازی نماید.
- عمل جستجو در آرایه انجام دهد.
- یک لیست را توسط روشهای مرتب سازی مجازی، انتخابی و غیره مرتب کند.

مقدمه

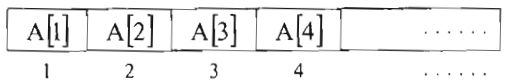
در فصل قبل در تعدادی از برنامه‌ها، تعدادی عدد را فقط با یک متغییر از ورودی دریافت کردیم و بعد از انجام عملیاتی روی هر کدام از آنها دومی را جایگزین اولی کردیم. در صورتی که در طول برنامه نیاز به همه داده ها داشته باشیم، روش فعلی که تاکنون بررسی کردیم، مشکل را حل نمی‌کند. فرضاً اگر بخواهیم تعدادی شماره دانشجویی یک کلاس را از ورودی بخوانیم سپس آنها را بصورت صعودی یا نزولی مرتب کنیم، نیاز داریم که همه این داده ها در حافظه بطور همزمان قرار گیرند. برای قرار گرفتن همزمان آنها در حافظه در زبانهای برنامه‌نویسی از آرایه ها استفاده می‌کنند.

۱-۸- آرایه و انواع آن

خانه‌های پشت سرهم از حافظه که همنوع بوده و توسط یک اسم معرفی می‌شوند، آرایه نام دارد. نحوه دسترسی به هر یک از اعضاء آرایه از طریق اندیس آرایه امکان‌پذیر است.

برای تعریف آرایه ابتدا طول آرایه که درحقیقت تعداد خانه‌های آنرا مشخص می‌کند، معین می‌گردد. سپس نوع خانه‌هایی که داده‌ها در آن قرار خواهند گرفت را تعیین می کنند و در نهایت عناصر داخل خانه‌های آرایه توسط اندیس آرایه که محل قرار گرفتن عنصر در آرایه را مشخص می کند، قرار می‌گیرند.

می‌توان آرایه را بصورت زیر در نظر گرفت.

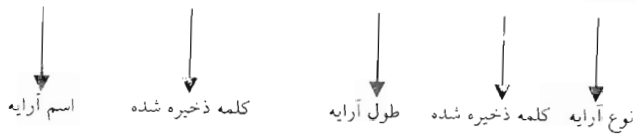


همانطور که ملاحظه می کنید داخل هر کدام از خانه یک عنصر قرار می‌گیرد و نوع عناصر باید با نوع آرایه یکی باشند.

۱-۱-۸- آرایه‌های یک بعدی

آرایه‌های یک بعدی بصورت زیر تعریف می‌شوند:


```
Name : array [ 1 .. Length ] of type ;
```



همانطور که ملاحظه می‌کنید آرایه توسط یک اسم معرفی می‌شود و همچنین طول و نوع خانه‌ها در تعریف آرایه کاملاً مشخص می‌باشند.

به طور مثال:

```
Var
    No: Array [ 1.. 50 ] of integer ;
    id: Array [ 1.. 20 ] of Byte ;
    Name , Fam: Array [ 1.. 30 ] of char ;
```

آرایه‌های تعریف شده در بالا هرکدام دارای طول ها و نوع های مختلف می‌باشد.

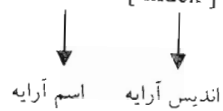
نکته: طول آرایه را معمولاً با توجه به نیاز برنامه تعیین می‌کنند.

مقداردهی آرایه‌ها مثل متغیرها به دو صورت امکان‌پذیر است:

۱- با استفاده از دستورات ورودی ۲- مقداردهی در طول برنامه در سطر بعدی

طریقه دسترسی به عناصر آرایه بصورت زیر می‌باشد:

```
Name [ index ]
```

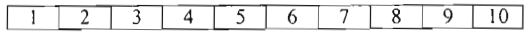


معمولاً اندیس آرایه که هنگام تعریف برنامه لحاظ می‌شود، از یک شروع می‌شود، ولی می‌تواند از یک هم شروع نشود. لذا در حالت کلی برای دسترسی به عناصر آرایه اسم آرایه همراه اندیس، محتویات خانه i ام را نمایش می‌دهد.

مثال ۱-۸ به آرایه زیر توجه کنید:

```
Var
  a : array [ 1 .. 10 ] of Byte ;
      .
      .
      .
      for i : = 1 to 10 do
          a [ i ] : = i ;
```

آرایه تعریف شده در بالا بصورت زیرمقداردهی می‌شود:



1 2 3 4 5 6 7 8 9 10

همچنین می‌توانستیم عناصر آرایه را از ورودی بخوانیم:

```
For   i : = 1   to   10   do
    Read(a [ i ] ) ;
```

برای نمایش اطلاعات یک آرایه به صورت زیر عمل می‌کنیم:

```
For   i : = 1   to   10   do
     Write ( a [ i ] : 3 ) ;
```

مثال بالا نحوه مقدار دهی، دسترسی و نمایش عناصر آرایه را نمایش می‌دهد.

مثال ۲-۸ برنامه‌ای بنویسید که ۱۰۰ عدد صحیح از ورودی دریافت کرده، بیشترین مقدار و محل وقوع آن را در خروجی چاپ نماید.

```
Program   Example2 ;
Var
                No: array [ 1.. 100 ] of integer ;
                 Max , i , index: integer ;
Begin
                Writeln ( 'Please enter ten numbers ' ) ;
                 For   i: = 1   to   100   do
                       Readln ( No [ i ] ) ;
                  Max: = No [ 1 ] ;
                  Index: = 1 ;
                  For   i: = 2   to   100   do
                         If   No [ i ] > Max  Then
                            Begin
                                    Max: = No [ i ] ;
                                     Index: = i ;
                            End ;
                  Writeln ( ' The  Maximum  is  = ' , Max ) ;
                  Writeln ( ' And  Index = ' , Index ) ;
End. { End  of  program }
```


در برنامه بالا انواع دسترسی به عناصر آرایه، مقداردهی و نمایش آنها نشان داده شده است.

مثال ۳-۸ برنامه‌ای بنویسید که یک عدد از ورودی دریافت کرده به مبنای دو ببرد.

```
Program   Example3 ;

Var
                j , Number , i:  integer ;
                 a: array [ 1.. 10 ]  of  integer ;
Begin
                Write ( 'Please  enter  a  number: ' ) ;
                 Readln ( Number ) ;
                 i: = 1 ;
                 While  Number > = 0  Do
                       Begin
                              A [ i ]: = Number  Mod 2 ;
                               Number: = Number  DIV  2 ;
                               i: = i + 1 ;
                        End ;
                  For   j: = i – 1   down to   1   do
                       Write ( a [ j ] ) ;
End. { End  of  program }
```

خروجی:

```
Please  enter  a  number: 13
1101
```

مثال ۴-۸ برنامه‌ای بنویسیدکه یک عدد از ورودی دریافت کرده عدد رابه مبنای ۱۶ ببرد.

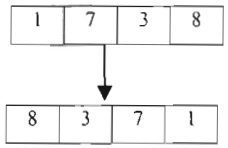
```
Program   Example4 ;
Var
        Hex: array[1..10] of char;
        j, i,r, number : integer,
Begin
        Write('Please enter a number');
        Readln(number);
        i:=1;
        while (number > 0) do begin
                r:= number mod 16;
                case r of
                        0: hex[i] := '0';
                        1: hex[i] := '1';
                        2: hex[i] := '2';
                        3: hex[i] := '3';
                        4: hex[i] := '4';
                        5: hex[i] := '5';
                        6: hex[i] := '6';
                        7: hex[i] := '7';
```

```
                8: hex[i] := '8';
                9: hex[i] := '9';
                10: hex[i] := 'A';
                11: hex[i] := 'B';
                12: hex[i] := 'C';
                13: hex[i] := 'D';
                14: hex[i] := 'E';
                15: hex[i] := 'F';
            end;
            i:= i+1;
            number:=number div 16;
        end;
        for j:=i-1 downto 1 do
            write(hex[i]);
end.
```

مثال ۵-۸ برنامه‌ای بنویسید که یک آرایه ۱۰۰ عنصری را از ورودی دریافت کرده عناصر آرایه را معکوس نماید.

بطور مثال:

فرض کنید آرایه ورودی آرایه روبرو باشد:



خروجی برنامه بصورت روبرو خواهد بود.

```
Program   Example5 ;
Var
        i , temp     : integer ;
          a   : array [1 .. 100 ] of integer ;
Begin
          Writeln ( ' please enter 100 Numbers : ' ) ;
          For   i : = 1   to   100   do
              Begin
                    Read ( a [ i ] ) ;
                    If ( i Mod 10 ) = 0 Then
                         Writeln ;
              end ;
           For   i : = 1   to   50   do
                Begin
                     Temp : = a [ 101 – i ] ;
                      a [ 101 – i ] : = a [ i ] ;
                      a [ i ] : = temp ;
                End;
               Writeln ;
```



```
          Writeln ( ' The  Inverse  of  Array ' ) ;
          For   i : = 1   to   100   Do
              Begin
                  Write ( a [ i ] :5) ;
                      If ( I  Mod  10 ) = 0  Then
                          Writeln ;
               End ;
End .
```

مثال۶-۸ نمرات دو درس ۴۰ دانشجو موجود است. برنامه‌ای بنویسید که این نمرات را از ورودی دریافت کرده سپس معدل دانشجویان را محاسبه و به همراه نمرات آنها در خروجی چاپ نماید.

```
Program   Example6 ;
Var
          i    : integer ;
          Grade1 , Grade2 , ave: Array[1..40] of  Real ;
Begin
          Writeln ( ' please  enter  Grade  of  Students: ' ) ;
          For   i: = 1   to   40   do
                 Readln ( Grade1 [ i ] , Grade2 [ i ] ) ;
          Writeln ( ' Grade1   Grade2   average ' ) ;
          For   i: = 1   to   40   do
            Begin
              Ave [ i ]: = ( Grade1 [ i ]  +  Grade2 [ i ] ) / 2 ;
              Writeln ( Grade1 [ i ]: 6: 2 , Grade2 [ i ]: 8: 2 , ave [ i ]: 8: 2);
             End ;
End.
```

۲-۱-۸- آرایه‌های دو بعدی

برای نمایش ماتریس درحافظه معمولاً از آرایه‌هایی بنام آرایه‌های دوبعدی استفاده می‌کنند. برای درک بیشتر این آرایه‌ها آنها را بصورت ماتریس در نظر می‌گیرند.

آرایه‌های دو بعدی بصورت زیر معرفی می‌شوند:

```
Name   :   array   [ 1 .. row , 1 .. column ]   of   type ;
```

اسم آرایه — کلمه ذخیره شده — تعداد سطرها — تعداد ستونها — کلمه ذخیره شده — نوع عناصر آرایه

برای مثال آرایه دو بعدی زیر را در نظر بگیرید:

```
a: array [ 1.. 5 , 1.. 5 ]  of  Real ;
```

در این صورت ما یک ماتریس ۵*۵ که هر عنصر آن می تواند یک عدد اعشاری باشد تعریف کردیم.

معمولاً برای پیاده‌سازی ماتریسها، از آرایه‌های دو بعدی استفاده می‌کنند. طریقه خواندن آرایه‌های دو بعدی بصورت زیر می‌باشد:

```
For  i := 1  to  5  do
     For  j := 1  to  5  do
          Read ( a [ i , j ] ) ;
```

و همانطور که ملاحظه می کنید برای دسترسی به عنصری که در سطر i ام و ستون j ام قرار دارد بصورت a [i , j] عمل می‌کنند.

برای نمایش یک آرایه دو بعدی در خروجی نیز مثل خواندن آرایه عمل می‌کنند و فقط دستور خواندن را با دستور Write یا Writeln عوض می‌کنند.

مثال ۷-۸ برنامه‌ای بنویسید که یک ماتریس ۵*۵ را از ورودی دریافت کرده، سپس آنرا در خروجی چاپ نماید.

```
Program  Example7 ;
Var
  a : array [ 1 .. 5 , 1 .. 5 ] of integer ;
  i , j : integer ;
Begin
                For  i := 1  to  5  do
                    Begin
                        For  j := 1  to  5  do
                                Read ( a [ i , j ] ) ;
                        Writeln  ;
                       End ;
                  Writeln ;
                  For  i := 1  to  5  do
                       Begin
                         For  j := 1  to  5  do
                                Write ( a [ i , j ] : 5 ) ;
                          Writeln ;
                       End ;
End .
```

برنامه بالا یک ماتریس ۵*۵ را از ورودی دریافت کرده، سپس آنرا به صورت یک ماتریس در صفحه نمایش چاپ می‌کند.

مثال ۸-۸ برنامه‌ای بنویسید که دو ماتریس ۱۰ * ۱۰ را از ورودی دریافت کرده، سپس جمع دو ماتریس را محاسبه ودر خروجی چاپ نماید:

```
Program  Example8 ;
Var
          a , b , c : array [ 1 .. 10 , 1 .. 10 ] of integer ;
          i , j : integer ;
Begin
          Writeln ( 'enter elements of first Matrix' ) ;
          For  i := 1  to  10  do
               For  j := 1  to  10  do
                    Read ( a [ i , j ] ) ;
          Writeln ( 'Enter Elements of second matrix' ) ;
          For  i := 1  to  10  do
               For  j := 1  to  do
                    Read ( b [ i , j ] ) ;
          For  i := 1  to  10  do
               For  j := 1  to  10  do
                    C [ i , j ] := a [ i , j ] + b [ i , j ] ;
          For  i := 1  to  10  do
              Begin
                  For  i := 1  to  10  do
                       Write ( c [ i , j ] :4 ) ;
                  Writeln ;
               End ;
End . { End of program }
```

مثال ۹-۸ برنامه‌ای بنویسید که یک ماتریس ۵×۵ را از ورودی دریافت کرده، ترانهاده ماتریس را محاسبه و چاپ نماید.

```
Program   Example9 ;

                                                    Var
                    a : array [ 1 .. 3 , 1 .. 3 ] of Real ;
                    i , j : integer ;
                    temp : Real ;
Begin
                     Writeln ( 'Enter elements of Matrix ' ) ;
                     For   i : = 1   to   3   do
                          For   j : = 1   to   3   do
                Read ( a [ i , j ] ) ;
                    For   i : = 1   to   3      do
                        For   j : = 1   to   3   do
                                If   i < j   Then
                                     Begin
                                            Temp : = a [ i , j ] ;
                                            a [ i , j ] : = a [ j , i ] ;
                                            a [ j , i ] : = temp ;
                                      end ;
                    Writeln ;
                    Writeln ( ' The Transpase of Matrix ' ) ;
                    For   i : = 1   to   3   do
                            Begin
                                  For   j : = 1   to   3   do
                                          Write ( a [ i , j ] : 8 : 2 ) ;
                                  Writeln ;
                             End ;
End . { End of program }
```

خروجی برنامه بالا:

```
Enter  elements  of  Matrix
1      2      5
0      7      8
1      12     10
The  Transpose  of  Matrix
1.00   0.00   1.00
2.00   7.00   12.00
5.00   8.00   10.00
```

مثال ۱۰-۸ اطلاعات نمرات مربوط به ۵۰ دانشجو که عبارتند از نمره سه درس، در نظر گرفته و برنامه‌ای بنویسید که اعمال زیر را انجام دهد:

الف) معدل دانشجویان را محاسبه نماید.

ب) یک آرایه به نام Value تعریف کنید و آنرا بصورت زیر مقداردهی کنید:

معدل	Value
16 – 20	A
14 – 16	B
12 – 14	C
10 – 12	D
0 – 10	F

ج) نمرات و معدل دانشجویان بانمره عالی که برابر 'A' می‌باشد را نمایش دهید.

ساختار داده برنامه بالا را می توان بصورت زیر تعریف کرد:

۱. آرایه دو بعدی بصورت

```
Grade: array [ 1.. 50 , 1.. 4 ] of Real
```

۲. آرایه یک بعدی بصورت

```
Value: array [ 1.. 50 ] of char
```

در آرایه اول سه نمره قرار داده می شود سپس معدل را محاسبه و در خانه چهارم قرار می دهیم و بر اساس معدل آرایه دوم را مقداردهی می کنیم.

```
Program   Example10 ;
Var
          Grade: array [ 1.. 50 , 1.. 4 ] of Real ;
          Value: array [ 1.. 50 ] of char ;
           i: integer ;
Begin
           Writeln ( ' Enter Three Grade for 50 students ' ) ;
           For   i: = 1   to   50   do
                  Readln ( Grade [ i , 1 ] , Grade [ i , 2 ] ,  Grade [ i , 3 ] ) ;
           For   i: = 1   to   50      do
              Grade[ i , 4 ] ; = ( Grade[ i , 1 ] + Grade[ i , 2 ] +  Grade[ i , 3 ] ) / 3 ;
           Writeln (' The students average are ' ) ;
           For   i: = 1   to   50   do
                  Writeln ( Grade [ i , 4 ]: 8: 2 ) ;
           For   i: = 1   to   50   do
                  Begin
                        If Grade [ i , 4 ] > = 16  Then
                           Value [ i ]: = ' A '
                        Else if Grade [ i , 4 ] > = 14  Then
                           Value [ i ]: = ' B '
                         Else if Grade [ i , 4 ] > = 12  Then
                           Value [ i ]: = ' C '
                         Else if Grade [ i , 4 ] > = 10  Then
                           Value [ i ]: = ' D '
                         Else
                           Value [ i ]: = ' F ' ;
                      End. { End of for }
            Writeln ; Writeln ( ' The first students ' ) ;
            For   i: = 1   to   50   do
                If    Value [ i ] = ' A '   Then
                   Begin
                         For   j: = 1   to   4   do
                            Write ( Grade [ i , j ]: 8: 2 ) ;
                          Writeln ;
                    End ;
              Writeln ;
End. { End of program }
```

نمونه‌ای از اجرای برنامه بالا برای سه دانشجو بصورت زیر می‌باشد:

```
Enter Three Grades for 3 students
        12    12    12
        13    13    13
        12    14    10
The students average are
       12.00
       13.00
       12.00
The first students
     13.00  13.00  13.00  13.00
```


۸-۱-۳ آرایه‌های چند بعدی

می‌توان آرایه‌هایی با ابعاد بیشتر از دو نیز تعریف کرد. بطور کلی برای معرفی یک آرایه چند بعدی می‌توان بصورت زیر:

```
Name :  array [ 1.. length1 ]  of   array [ 1.. length2 ] ....
                                    Of array [ 1.. lengthN ]  of  Type
```

و یا بصورت :

```
Name: array [ 1.. length1 , 1.. length2 , ....  1.. lengthN ]  of  Type
```

عمل کرد.

برای مثال آرایه سه بعدی را بصورت زیر تعریف می‌کنند:

```
Var
        a: array [ 1.. 2 , 1.. 4 , 1.. 3 ] of Real ;
```

برای دسترسی به اعضای‌این آرایه درحالت کلی‌بصورت[i , j , k] a عمل‌می‌کنند. که در آن k , j , i ابعاد آرایه بوده و در محدوده تعریف شده برای آرایه صدق می‌کنند.

در پاسکال آرایه های با ابعاد بیشتر نیز می توان تعریف کرد.

۸-۲- نکاتی چند در مورد آرایه‌ها

آرایه‌های ساده را در بخش‌های قبل بررسی کردیم در این بخش قصد داریم مفاهیم گسترده‌تری از آرایه‌ها را ارائه دهیم. محدوده آرایه‌هایی که تا حال بررسی کردیم از یک شروع شده و به اندازه طول مورد نیاز تعریف کردیم. حال انواع مختلفی از آرایه‌ها را به همراه کاربردهای آنها اشاره می کنیم.

۱- تعریف آرایه با محدوده منفی برای مثال:

```
A: array [ -10.. 10 ] of Real ;
```

عناصر تشکیل‌دهنده این آرایه عبارتند از:

A [-10] , A [- 9] , ... , A [-1] , A [0] , A [1] , ... , A [10]

این آرایه در کل شامل ۲۱ عنصر می‌باشد. کاربرد چنین آرایه‌ای می‌تواند در تبدیل دماها به هم و موارد مشابه باشد بعنوان مثال مقدار عنصر اول آرایه (A [-10]) می‌تواند یک عدد حقیقی معرف درجه فارنهایت معادل با 10- درجه سانتیگراد باشد.

۲- تعریف آرایه از نوع منطقی (Boolean) برای مثال:

```
A: array [1.. 20 ] of Boolean ;
```

در مثال بالا آرایه‌ای بنام A با ۲۰ عنصر معرفی می‌شود که هر کدام از آنها می‌توانند مساوی False یا True باشند و غالباً برای درستی یا نادرستی عبارات مختلف از این نوع آرایه ها استفاده می‌کنند.

۳- آرایه‌های با محدوده منطقی: برای مثال

```
A: array [ Boolean ] of integer ;
```

در مثال بالا آرایه‌ای بنام A با ۲ عضو A [false] , B [True] معرفی می‌شود که به هر کدام از آنها می‌توان یک عدد صحیح را تخصیص داد.

فرض کنید در یک آزمایش تصادفی، یک سکه به تعداد مشخص پرتاب می‌شود و در نظر است که با توجه به تعداد شیرها و یا خطهایی که ظاهر شده است، تصمیماتی گرفته شود. در هر بار پرتاب سکه دو حالت رخ می‌دهد که با توجه به خاصیت نوع‌های منطقی استفاده از این نوع آرایه‌ها می‌تواند مفید باشد.

۴- آرایه‌ای با محدوده کاراکتری: برای مثال

```
A: array [ 'A'.. 'Z' ] of Real ;
```

در مثال بالا آرایه‌ای بنام A با ۲۶ عنصر معرفی می‌شود که به هر کدام از آنها می‌توان یک عدد اعشاری تخصیص داد.

۵- آرایه‌ای که برای اعضای آن محدودیت قائل شویم برای مثال:

```
A: array [ 1.. 20 ] of 1.. 30
```


در مثال بالا آرایه‌ای به نام A با ۲۰ معرفی می‌شود که هر کدام از خانه‌های آرایه می‌تواند مقداری بین ۱ تا ۳۰ را در خود جای دهد.

نمونه‌های بالا همه در مورد آرایه‌های یک بعدی ذکر شدند ولی آنها را می‌توان برای آرایه‌ها چند بعدی نیز بسط داد.

مثال ۱۱-۸ به مثال زیر در مورد مقداردهی آرایه‌ها توجه کنید.

```
Const
    Digit: array [ 0.. 9 ] of char = ( '0' , '1' , '2' , ... , '9' ) ;
```

در مثال فوق آرایه digit از نوع کاراکتر معرفی شده که شامل ۱۰ عضو می‌باشد و مقادیر آنها در زمان تعریف تعیین شده‌اند.

توجه: در مسائلی که نیاز به استفاده از آرایه‌ها را احساس می‌کنید، قبل از نوشتن برنامه دقیقاً ساختار داده برنامه را تعیین کنید تا درطول برنامه دچارمشکل نشوید. دربرنامه‌های بزرگتر یکی از کارهای اصلی نوشتن برنامه، تعیین ساختار داده برنامه می‌باشد.

۳-۸- جستجو و مرتب‌سازی (Search and Sort)

یکی از مسائلی که در بحث طراحی الگوریتم بسیار مهم است، بحث مرتب‌سازی و جستجو می باشد. منظور از جستجو اینست که یک مقداری را از یک لیست جستجو کنیم و منظور از مرتب سازی اینست که یک لیست مرتب از داده ها را ایجاد کنیم. حال تعدادی الگوریتم که برای مرتب‌سازی و جستجو بکار می‌روند را بررسی می‌کنیم بخصوص زمانیکه ساختار داده ما یک آرایه باشد.

۱-۳-۸- جستجو در آرایه

همانطور که قبلاً اشاره شد هدف از جستجو عبارتست از جستجوی عضوی از یک لیست. فرض کنید در یک دفترچه تلفن کامپیوتری بخواهیم شماره تلفن مربوط به یک فرد خاص را بیابیم.

در کل دو نوع عمل جستجو را در این کتاب بررسی می‌کنیم:

۱. جستجوی خطی Linear search

۲. جستجو دودویی Binary search

در جستجوی خطی عبارت مورد جستجو را نخست با اولین عضو آرایه مقایسه می‌کنیم، اگر برابر بود عمل جستجو با موفقیت همراه بوده و عمل جستجو خاتمه می‌یابد در غیر اینصورت روند را ادامه داده و عبارت مورد جستجو را بترتیب با عضو دوم، سوم ... مقایسه می‌کنیم تا اینکه حالت تساوی حاصل شود و اگر این حالت حاصل نشد، عبارت مورد جستجو در لیست قرار ندارد.

توجه: اشکال این روش جستجو اینست که در صورتی که تعداد داده‌ها زیاد باشد زمان زیادی را از دست خواهیم داد اگر فرض کنیم تعداد عناصر آرایه n باشد در حالت متوسط به اندازه $\frac{n}{2}$ مقایسه صورت می‌گیرد.

قطعه برنامه زیر را می توان در حالت کلی برای جستجوی خطی بکار برد:

```
Flag: = False ;
i: = 0 ;
While ( i < = N ) and ( flag ) Do
     Begin
          i: = i + 1 ;
          if A [ i ] = x Then
           Begin
                Index: = i ;
                 Flag: = True ;
            End ;
       End ;
If flag Then
   Writeln ( 'The Element is found' )
Else
    Writeln ( 'The Element is not found' ) ;
```

در قطعه برنامه بالا یک آرایه بنام A بطول N در نظر گرفته شده که عناصر آن دلخواه و نامرتب هستند و x عنصری است که در لیست مورد جستجو قرار می‌گیرد. در صورتی که یافت شود شماره خانه مورد نظر حفظ می‌شود و پیغام The Element is found چاپ می‌شود.

درجستجوی دودوئی لیست اولیه بایدمرتب باشد. برای جستجو در چنین آرایه‌ای نخست اندیس وسط آرایه را پیدا می‌کنیم و عنصر واقع در این اندیس را با عبارت مورد جستجو مقایسه می‌کنیم و حالات زیر ممکن است حاصل شود:

(Low اندیس ابتدای آرایه و upper اندیس آخرین عناصرآرایه و middle اندیس عنصر وسط می‌باشد.)

الف)

```
if A [ middle ] < X Then
Low : = middle
```

در اینصورت

و مقدار جدید middle را که عبارتست از:

```
Middle : = ( low + middle ) / 2
```

محاسبه می کنیم.

ب)

```
if A [ middle ] > X Then
upper : = middle
```

در اینصورت

و دوباره مقدار جدید middle را محاسبه می کنیم.

ج)

```
if A [ middle ] = X Then
Write ( 'The Eiement is found' )
```

در صورتی که حالت‌های الف یا ب اتفاق بیفتد، عمل جستجو را تا زمانیکه Low < upper می باشد ادامه می‌دهیم و در هر مرحله که حالت ج رخ دهد عمل جستجو خاتمه می‌یابد.

برنامه مربوط به‌جستجوی دودوئی رابعنوان تمرین‌به‌خواننده واگذار می‌کنیم.

توجه: الگوریتم بالا برای آرایه مرتبی است که به طور صعودی مرتب شده باشد با تغیرات جزئی می‌توان آن را برای حالت نزولی نیزبکاربرد. همانطور که ملاحظه می‌کنید زمان موردنیاز برای جستجوی در این الگوریتم نسبت به روش قبلی خیلی بهتر می‌شود.

با توجه به دو روش بالا برای جستجو می‌توان چنین بیان کرد که:

- اگر لیست داده ها مرتب باشد استفاده از جستجوی دودوئی می‌تواند بسیار مفید باشد (از لحاظ زمانی)
- اگر لیست داده‌ها نامرتب باشد از جستجوی خطی برای جستجو می‌توان استفاده کرد و جستجوی دودوئی ممکن است در چنین حالتی به جواب نرسد.

۲-۳-۸- مرتب‌سازی

عمل مرتب‌سازی عمومی‌ترین عملی است که در سیستم‌های برنامه‌نویسی وجود دارد. مفهوم مجموعه‌ای از عناصر مرتب، بطور قابل ملاحظه‌ای با زندگی روزمره ما در هم آمیخته است. بعنوان مثال روند پیدا کردن اسم خود از یک روزنامه، حاوی اسامی قبولین کارشناسی‌ارشد را در نظر بگیرید. اگر اسامی بر اساس فامیلی مرتب باشد روند پیدا کردن اسم خود بسیار ساده می‌باشد ولی اگر اسامی مرتب نباشد، پیدا کردن اسم خود از روزنامه کار بسیار مشکلی بنظر می‌رسد.

برای مرتب‌سازی داده‌ها روشهای متفاوتی وجود دارد و تفاوت روشهای مرتب‌سازی در زمان اجرای آنها می باشد. در حالت کلی با توجه به تعداد ورودیها (داده ها) و نوع مسئله مرتب‌سازی می توان از انواع روش‌های مرتب‌سازی استفاده نمود.

حال بعضی از روش های مرتب سازی عمومی را بررسی می‌کنیم.

- **مرتب‌سازی حبابی (Bubble sort)**

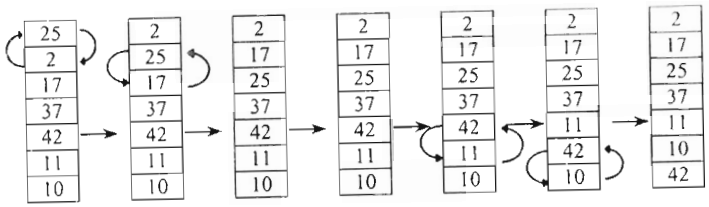
ساده‌ترین روش مرتب‌سازی روش مرتب‌سازی حبابی می‌باشد. یکی از خصوصیات بارز این نوع مرتب‌سازی این است که فهم آن ساده بوده و برنامه‌نویسی آن به سهولت انجام می گیرد.

مرتب‌سازی حبابی نخست عنصر اول و دوم را با هم مقایسه می‌کند و در صورت نیاز، آنها را جابجا می‌کند، سپس عنصر دوم و سوم را مقایسه می‌کند. این عمل را تا زمانیکه به انتهای آرایه نرسیده تکرار می‌کند، در پایان مرحله اول بزرگترین عنصر در آخرین خانه آرایه قرار می‌گیرد. در مرحله دوم از خانه اول تا خانه N - 1 عمل بالا را انجام می‌دهد. این روند را تا زمانیکه تمام عناصر آرایه مرتب نشده‌اند ادامه می‌دهد و در نهایت یک لیست مرتب شده بصورت صعودی در خروجی تولید می‌شود.

مثال زیر را در نظر بگیرید:

25 2 1737 42 11 10

مرحله اول:



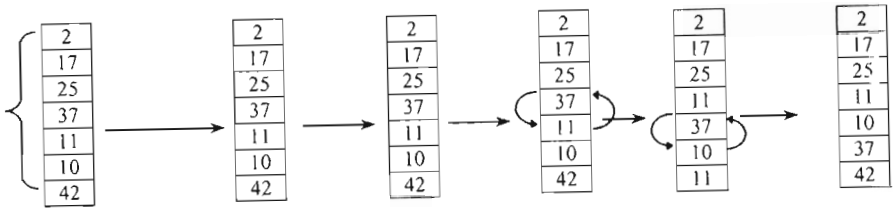
همانطور که ملاحظه می‌کنید در پایان مرحله اول عنصر با بیشترین اولویت در اخرین خانه آرایه قرار می‌گیرد.

آرایه حاصل:

2
17
25
37
11
10
42

حال ۶ عنصر اول آرایه را با روش بالا مرتب می‌کنیم.

مرحله دوم:

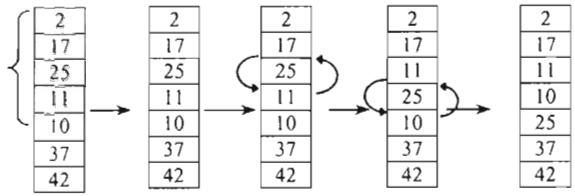


در آرایه حاصل عنصر 37 نیز در جای خود قرار می‌گیرد و دو عنصر آخر یعنی 37 , 42 مرتب می‌شوند. آرایه حاصل بعد از اتمام مرحله دوم:

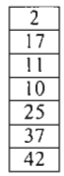
2
17
25
11
10
37
42

حال 5 عنصر اول آرایه را با روش بالا مرتب می‌کنیم.

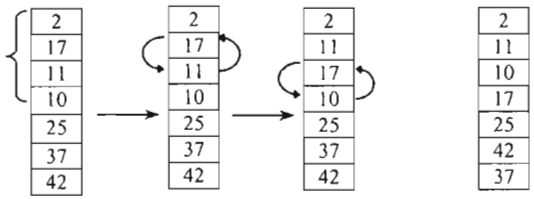
مرحله سوم:



در آرایه حاصل عنصر 25 نیز در جای خود قرار می گیرد و آرایه حاصل:

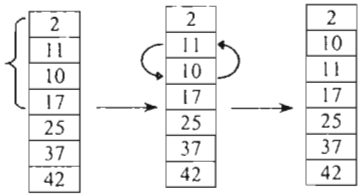


مرحله چهارم:



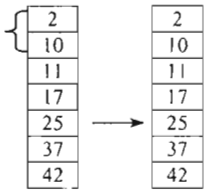
همانطور که ملاحظه می‌کنید عنصر 17 نیز درجای مخصوص به خود قرار می‌گیرد و آرایه حاصل پس از اتمام مرحله چهارم بصورت زیر می‌باشد:

مرحله پنجم:



در پایان مرحله پنج عنصر 11 در جای مخصوص به خود قرار می‌گیرد و آرایه حاصل بصورت زیر می‌باشد:

مرحله ششم:



همانطور که ملاحظه می‌کنید بعد از اتمام مراحل مرتب‌سازی آرایه حاصل یک ارایه مرتب می‌باشد.

```
For   i: = 1   to   n   do
      For   j: = 1   to   n - 1     do
        If  x [ j ] > x [ j + 1 ]  Then
            Begin
                  Temp: = x [ j ] ;
                  X [ j ]: = x [ j + 1 ] ;
                  X [ j + 1 ]: = temp  ;
            End ;
```

قطعه برنامه بالا آرایه x بطول N را به صورت صعودی مرتب می‌کند.

توجه: الگوریتم مرتب‌سازی حبابی کارایی کمتری نسبت به روشهای دیگر مرتب‌سازی دارد و همانطور که در قطعه برنامه بالا ملاحظه می‌کنید تعداد مقایسه‌ها در این روش بصورت زیر محاسبه می‌شود:

$$n - 1 + n - 2 + \ldots + 1 = \frac{(n-1)(n-2)}{2}$$

و تعداد تعویض ها n^2 می باشد و این نشان‌دهنده اینست که بطور متوسط n^2 حالت مقایسه در این روش وجود دارد و این زمان بسیار زیادی است، بخصوص اگر تعداد داده ها زیاد باشد.

- **مرتب‌سازی انتخابی (Selection sort)**

در این روش مرتب‌سازی نخست کوچکترین عنصر را در کل آرایه پیدا کرده در خانه اول آرایه قرار می‌دهیم سپس عنصر کوچکتر بعدی را یافته در خانه دوم قرار می‌دهیم این روند را تا زمانیکه کل آرایه مرتب نشده ادامه می‌دهیم الگوریتم مرتب سازی انتخابی بصورت زیر می‌باشد:

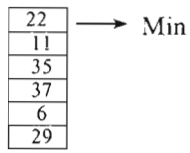
مرحله اول: موقعیت کوچکترین عنصر لیست N عنصری $A[2], \ldots, A[N]$ ، $A[1]$ را پیدا می کند و جای آن را با خانه اول عوض می‌کند.

مرحله دوم: موقعیت کوچکترین عنصر زیر لیست $N-1$ عنصری $A[N], \ldots, A[2]$ را پیدا می‌کند و جای آن را با خانه دوم عوض می‌کند.

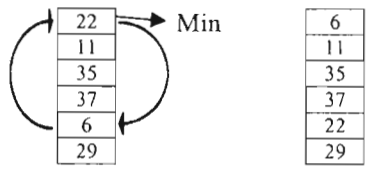
مرحله سوم: این روند را ادامه می‌دهد تا زمانیکه دو عنصر $A[N-1], A[N]$ باقی بماند با مقایسه این دو عنصر و در صورت نیاز جابجایی محتویات آنها لیست بصورت صعودی مرتب خواهد شد.

برای مثال اعداد زیر را در نظر بگیرید:

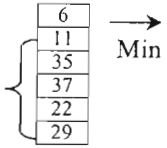
22 , 11 , 35 , 37 , 6 , 29



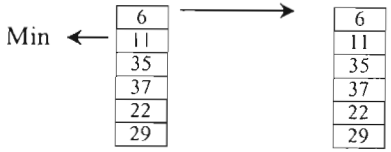
مرحله اول:



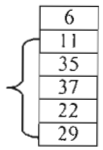
پس از پایان مرحله اول کوچکترین عنصر آرایه در خانه اول آرایه قرار می‌گیرد. و زیر لیست ۵ عنصری باید مرتب شود. آرایه حاصل:



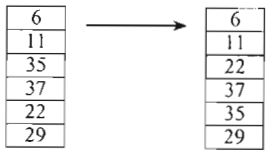
حله دوم:



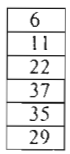
پس از مرحله دوم دو عنصر اول مرتب شده هستند و آرایه حاصل بصورت زیر می‌باشد.



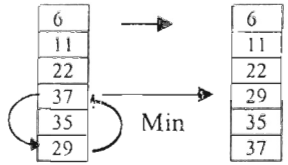
مرحله سوم:



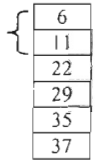
پس از اتمام این مرحله سه عنصر اول آرایه مرتب می‌شوند و آرایه حاصل:



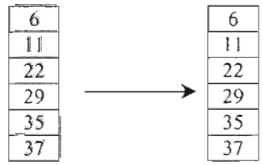
مرحله چهارم:



در پایان این چهار مرحله عنصر آرایه مرتب هستند و آرایه حاصل:



مرحله پنجم:



آرایه حاصل بطور کامل مرتب است.

قطعه برنامه‌ای که برای مرتب‌سازی انتخابی می‌توان نوشت بصورت زیر می‌باشد:

```
For   i: = 1   to   n-1   do
    Begin
            Min: = x [ i ] ;
            Index: = i ;
            For   j: = i +1 to   n   do
                  If  x [ j ] < Min Then
                  Begin
                        Min: = x [ j ] ;
                        Index: = j ;
                   End ; { find The smallest Element }
             X [ index ]: = x [ i ] ;
             X [ i ]: = Min ; { swap Minimum With other Element }
End ; { end of selection sort }
```

توجه: در این روش مرتب‌سازی تعداد تعویض‌ها در حالت کلی n و تعداد مقایسه‌ها n^2 می‌باشد و با یک مقایسه با روش قبلی ملاحظه می‌کنید که این روش نسبتاً بهتر از روش قبلی عمل می‌کند.

روشهای مرتب‌سازی بهتری از قبیل مرتب‌سازی سریع(Quick sort)، مرتب‌سازی

هرمی (heap sort) ، مرتب‌سازی ادغامی (merge sort) و غیره نیز وجود دارد که در بحث ساختمان داده مطرح خواهد شد.

۸-۴- حل چند مثال در مورد آرایه‌ها

در این بخش به تعدادی برنامه که ساختار داده آنها غالباً آرایه می‌باشد ارائه می‌دهیم از خوانندگان تقاضا می‌شود که این برنامه‌ها را برای خود اجرا کرده تا در عمل با نحوه اجرای آنها آشنا شوند.

مثال ۱۲-۸ برنامه‌ای بنویسید که یک عدد صحیح از ورودی دریافت کرده سپس اعداد اول قبل از آن را تولید و در آرایه قرار دهد.

```
Program   Example12 ;
Var
        a : array [ 1 .. 50 ] of integer ;
        i , j , N : integer ;
        k : Byte ;
        flag : Boolean ;
Begin
      Write ( ' Enter  A  Number : ' ) ;
       Readln ( N ) ; a [1] : = 2 ; a [2] : = 3 ; flag : = True ;
       k : = 2 ;
       For   i : = 4   to   N      do
              Begin
                    For   j : = 2   to   ( i div 2 )   do
                       If  ( i  mod  j ) = 0  Then
                             Flag : = false ;
                     If  flag  Then
                     Begin
                           K : = k + 1 ;
                           A [ k ] : = i ;
                      End ;
                      Flag : = Ture ;
             End ; { end  of  for }
      Writeln ( ' The prime  Numbers  before  N ' ) ;
      For   j : = 1   to   K   do
             Writeln ( a [ i ] : 5 ) ;
End . { End  of  program }
```

خروجی قطعه برنامه بالا برای عدد 12 بصورت زیر می‌باشد:

```
Enter  A  Number: 12
The prime Numbers before N
   2
   3
   5
   7
   11
```

مثال ۱۳-۸ برنامه‌ای بنویسید که یک آرایه 20 عنصری از ورودی دریافت کرده عناصری از آن که کامل می‌باشد را با صفر جایگزین نماید (عناصری که عدد کامل می‌باشند به صفر تبدیل نماید این روش نوعی حذف از آرایه محسوب می شود)

```
Program Example13 ;
Var
        a: array [ 1.. 20 ] of integer ;
        i: Byte ;
        sum , j: integer ;
Begin
        Writeln ( ' Enter 20 Numbers ' ) ;
        For   i: = 1   to   20      do
              Read ( a [ i ] ) ;
        Writeln ;
        Sum: = 0 ;
        For   i: = 1   to   20      do
              Begin
                    For   j: = 1   to   ( a[i]  div 2 )   do
                          If  ( a [ i ] mod j ) = 0  Then
                              Sum: = sum + j ;
                    If  sum = a [ i ] Then
                        A [ i ]: = 0 ;
                    Sum: = 0 ;
              End ; { end of for }
        Writeln ( ' The output of program ' ) ;
        For   i: = 1   to   20   do
              Write ( a [ i ]: 6 ) ;
End. { End of program }
```

مثال ۱۴-۸ برنامه‌ای بنویسید که نمرات حداکثر ۱۰۰ دانشجو را از ورودی دریافت کرده، میانگین نمرات کلاس و تعداد کسانی که نمره آنها زیر میانگین است را محاسبه و چاپ کند.

توجه کنید که تعداد ورودیها ممکن است دقیقاً ۱۰۰ نباشد.


```
Program Example14 ;
Var
            score: array [ 1.. 100 ] of Real ;
            Ave , sum: Real ;
            N , i , count: Word ;
Begin
        Write ( ' Enter Number of data: ' ) ;
          Readln ( N ) ; sum: = 0.0 ; count: = 0 ;
          Writeln ( ' Enter array ' ) ;
       For   i: = 1   to   N      do
             Read ( score [ i ] ) ;
        Writeln ;
        For   i: = 1   to   N   do
                Sum: = sum + score [ i ] ;
        Ave: = sum / N ; { calculate average }
        Writeln ( ' The Average scores = ' , Ave: 8: 2 ) ;
        For   i: = 1   to   N   do
              If score [ i ] < Ave  Then
                    Count: = count + 1 ;
        Write ( ' The number of students with scores ' ) ;
        Writeln ( ' < Average = ' , count: 5 ) ;
End. { End of program }
```

خروجی حاصل از اجرای این برنامه بصورت زیر است:

```
Enter Number of data: 4
Enter array
12   13   10   5
The Average scores = 10.00
The number of students with scores < Average = 1
```

مثال ۱۵-۸ برنامه‌ای بنویسید که یک ماتریس ۳*۳ را از ورودی دریافت کرده و مجموع هر سطر را انتهای همان سطر به همراه خود ماتریس در خروجی چاپ نماید.

```
Program   Example15 ;
Var
          a: array [ 1.. 3 , 1.. 3 ] of Real ;
          sum: Real ;
          i , j: byte ;
Begin
          Writeln ( ' Enter  Array ' ) ;
          for   i: = 1   to   3      do
                for   j: = 1   to   3      do
                Read ( a [ i , j ] ) ;
          Writeln  ;
          Writeln  ( ' The  result  matrix ' )  ;
          For   i: = 1   to   3   do
                Begin
                     For   j: = 1   to   3   do
                          Begin
                               Write  ( a [ i , j ]: 8: 2 )  ;
                               Sum: = sum + a [ i , j ];{ calculate sum  of  any row }
                          End  ;
                          Writeln ( sum: 8: 2  )  ;
                          Sum : = 0  ;
                End  ;
End. { End  of  program }
```

خروجی حاصل از برنامه بالا بصورت زیر است:

```
Enter array
12   10   3    4    3    3    7    3    5
The result Matrix
    12.00   10.00   3.00   25.00
     4.00    5.00   3.00   10.00
     7.00    3.00   5.00   15.00
```

مثال ۱۶-۸ برنامه‌ای بنویسید که دو ماتریس ۵*۵ را از ورودی دریافت کرده و سپس حاصلضرب دو ماتریس را محاسبه و در خروجی چاپ کند.


```
Program Example16 ;
Var
          a , b , c: array [ 1.. 5 , 1.. 5 ] of integer ;
          i , j , k: byte ;
          sum: integer ;
Begin
      Writeln ( ' Enter  first  matrix ' ) ;
      for   i: = 1   to   5      do
            for   j: = 1   to   5      do
                  Read ( a [ i , j ] ) ;
      Writeln  ;
      Writeln ( ' Enter  second  matrix ' ) ;
      for   i: = 1   to   5      do
            for   j: = 1   to   5      do
            Read ( b [ i , j ] ) ;
      Writeln  ;
      for   i: = 1   to   5      do
           for   j: = 1   to   5      do
             Begin
                  Sum: = 0 ;
                  For   k: = 1   to   5   do
                        Sum: = sum + a [ i , k ]  *  b [ k , j ]
                  a [ i , j ]: = sum ;
            End ; { Multiplication  matrix }
       for   i: = 1   to   5      do
           Begin
                for   j: = 1   to   5      do
                  Write ( c [ i , j ]: 5 ) ;
                 Writeln ;
        End ;
End.
```

برای اجرای برنامه بالا دو ماتریس ۵*۵ را از ورودی بخوانید سپس حاصلضرب دو ماتریس را در خروجی بصورت ماتریس چاپ می کند.

مثال ۱۷-۸ برنامه‌ای بنویسید که نمرات حداکثر ۱۰۰ دانشجو را از ورودی دریافت کرده و به یکی از روش‌های مرتب‌سازی، آنها را مرتب نماید.

```
Program Example17 ;
Var
        score : array [ 1 .. 100 ] of Real ;
        Min : Real ;
        n , i , index : integer ;
Begin
      Writeln ( ' Enter Number of data : ' ) ;
      Readln ( N ) ;
      Writeln ( Enter Array ' ) ;
          for   i : = 1   to   N      do
              Read ( score [ i ] ) ;
           for   i : = 1   to   N      do
              Begin
                  Min : = score [ i ] ;
                  Index : = i ;
                  for   j : = i + 1   to   N      do
                     if score [ j ] < Min do
                        Begin
                            Min : = score [ j ] ;
                            Index : = j ;
                        End ;
                  score [ index ] : = score [ i ] ;
                  score [ i ] : = Min ;
              end ;
      Writeln  ;
      Writeln ( ' The sorted array ' ) ;
      for   i : = 1   to   N      do
              Write ( score [ i ] : 8 : 5 ) ;
End . { End of program }
```

خروجی برنامه بالا بصورت زیر است:

```
Enter Number of data: 6
Enter Array
    12   17.5   13.25    14    8.5    10.5
The sorted array
    8.50   10.50   12.00   13.25   14.00   17.50
```

مثال ۱۸-۸ دو آرایه مرتب L1 , L2 حداکثر به طول ۳۰ را در نظر گرفته برنامه‌ای بنویسید که از ادغام دو آرایه، آرایه سومی بنام L3 ایجاد کند بطوریکه آرایه سوم مرتب باشد.

توجه کنید که نمی‌توانید دو آرایه رادر آرایه سوم ریخته آرایه سوم را مرتب کنید.


```
Program Example18 ;
Var
          L1 , L2: array [ 1.. 30 ] of integer ;
          L3: array [ 1.. 60 ] of integer ;
          M , N , i , j , k: Byte ;
Begin
     Write ( ' Enter Dimention of arrays: ' ) ;
     Readln ( N , M ) ;
     Write (' Enter first Array ' )  ;
     for   i: = 1   to   N      do
             Read ( L1 [ i ] ) ;
     Writeln ( 'Enter second Array ' )  ;
     for   j: = 1   to   M      do
             Read ( L2 [ i ] ) ;
        i: = 1  ; j: = 1  ; k: = 1  ;
        while ( i < = N ) and  ( j < = M )  Do
            Begin
                If  L1 [ i ] > L2 [ j ]  Then
                  Begin
                      L3 [ k ]: = L2 [ j ] ;
                      J: = j + 1  ;
                  End
                Else if  L1 [ i ]  <  L2 [j ]   Then
                    Begin
                        L3 [ k ]: = L1 [ i ] ;
                        i: = i + 1  ;
                    End
                Else
                    Begin
                        L3 [ k ]: = L1 [ i ] ;
                        k: = k + 1  ;
                        L3 [ k ]: = L2 [ j ] ;
                        i: = i + 1  ;
                        j: = j + 1  ;
                    end ;
                k: = k + 1  ;
            end ; { end of while }
        if  i < = N   Then
            for   p: = i   to   n      do
              Begin
                  L3 [ k ]: = L1 [ p ] ;
                  k: = k + 1  ;
              end
        Else if  j < = M  Then
            for   p: = j   to   M      do
              Begin
                  L3 [ k ]: = L2 [ p ] ;
                  k: = k + 1  ;
              end ;
        Writeln  ;
        Writeln ( ' The result of merge is ' ) ;
        for   i: = 1   to   k - 1      do
           Begin
               Write ( L3 [ i ] : 5 ) ;
               If ( i  mod  10 ) = 0  Then
                   Writeln  ;
           End ; { end of merge }
End. { End of program }
```

در برنامه بالا بافرض اینکه دو آرایه مرتب هستند،عمل ادغام انجام می‌شود. نحوه کار ادغام دو آرایه به این صورت است که نخست عنصراول آرایه اول باعنصراول آرایه دوم مقایسه می‌شود هرکدام کوچکتر باشد درآرایه سوم قرار می‌گیرد و مقدار اندیس دو آرایه یک واحد اضافه می‌گردد. اینکار تازمانی ادامه می‌یابد که عناصر یکی از آرایه‌ها یا هر دو در آرایه سوم در جای مخصوص به خود قرار گیرند، در صورتی که فقط یکی از آرایه‌ها تمام شده باشد عناصر باقیمانده آرایه دوم در انتهای آرایه سوم قرار می‌گیرد.

خروجی برنامه بالا بصورت زیر می باشد:

```
Enter Dimention of Arrays: 3   5
Enter first Array
   12   17   29
Enter second array
   5   7   17   34   38
The result of merge is
   5   7   12   17   17   29   34   38
```

مثال ۱۹-۸ برنامه‌ای بنویسید که یک جمله حداکثر ۸۰ کاراکتری را از ورودی دریافت کرده و سپس کاراکترهای فضای خالی (space) را با کاراکتر (*) جایگزین کند.

```
Program     Example19 ;
Var
        state: array [ 1.. 80 ] of char ;
            i , N : integer ;
Begin
       Writeln ( ' Enter  Number  of  sentence: ' ) ;
Readln ( N ) ;
            Writeln (' Enter  sentence ' )  ;
            for   i: = 1   to   N      do
                Read ( state [ i ] ) ;
            for   i: = 1   to   N      do
                if  state [ i ] = ' ' Then
                   state [ i ]: = ' * ' ;
            Writeln  ;
            Writeln ( ' The  output  sentence ' )  ;
            for   i: = 1   to   N      do
                   Write ( state [ i ] ) ;
End. { End  of  program }
```

خروجی برنامه بالا بصورت زیر است:

```
Enter  Number  of  sentence: 30
Enter  sentence
Book name  is Pascal Programming
The  output  sentence
Book*name*is*Pascal*Programming
```


مثال ۲۰-۸ برنامه‌ای بنویسید که یک آرایه ۵۰ عنصری از نوع صحیح را از ورودی دریافت کرده و سپس با دریافت، عددی از ورودی در صورت وجود محل وقوع آن را در خروجی چاپ نماید.

```
Program     Example20 ;
Const       N = 50;
Var
            List: array [ 1.. N ] of integer ;
            i , Number: integer ;
            flag: Boolean ;
Begin
        Flag = False ;
        Writeln ( ' please  enter  array: ' ) ;
        for   i: = 1   to   50      do
                    Read ( list [ i ] ) ;
        Writeln ;
        Writeln ( 'Enter  a  Number ' )  ;
        Readln ( number ) ;
        i: = 1 ;
        while ( i < = N ) and  flag  do
                Begin
                    If  list [ i ] = Number  Then
                    Begin
                            Index: = i ;
                            Flag: = True  ;
                    End ;
                    i: = i + 1 ;
                End ; { search  Algori Then }
        If  flag = Ture  Then
                Writeln ( ' The  Number  is  found ' , index: 4 )
        Else
                Writeln ( ' The  Number  is  Not  found ' )
End. { End  of  program }
```

برنامه بالا با استفاده از الگوریتم جستجوی خطی عمل جستجو را انجام می‌دهد در صورت پیدا کردن عنصر مورد نظر اندیس آن را در خروجی چاپ می‌کند.

مثال ۲۱-۸ اطلاعات حداکثر 100 دانشجو که عبارتند از:

شماره دانشجویی نمره درس 1 نمره درس 2 نمره درس 3

را در نظر بگیرید برنامه‌ای بنویسید که:

الف) اطلاعات دانشجویان را از ورودی دریافت کند.

ب) معدل دانشجویان را محاسبه نماید.

ج) اطلاعات دانشجویان را بر اساس شماره دانشجویی مرتب کند.

د) اطلاعات دانشجویان را در خروجی با پیغام های مناسب چاپ نماید.

ساختار داده‌ها برنامه بالا را بصورت زیر تعریف می‌کنیم:

```
Id : array [ 1 .. 100 ] of longint ;
Grade : array [ 1 .. 100 , 1 .. 4 ] of Real ;
```

که درآن آرایه اول برای‌شماره دانشجوئی و آرایه‌دوم برای‌نمرات ومعدل می‌باشد.

```
Program     Example21 ;
Var
          Id : array[ 1 .. 100 ] of longint ;
          Grade : array[ 1 .. 100 , 1 .. 4 ] of Real ;
Begin
      Write ( 'Please enter number of student : ') ;
      Readln ( N ) ;
      Writeln ( ' Enter id and Three Grades ' ) ;
      for   i : = 1   to   N      do
            Begin
                  Read ( id [ i ] ) ;
                    for   j : = 1   to   3      do
                       Read ( Grade [ i , j ] ) ;
                  Writeln ;
            End ; { end of input data }
      for   i : = 1   to   N      do
             Begin
                   Sum : = 0 ;
                  for   j : = 1   to   3      do
                        Sum : = sum + Grade [ i , j ] ;
                 Grade [ i , 4 ] : = sum / 3 ;
             End ;
       for   i : = 1   to   N      do
             Begin
                   min : = id [ i ] ;
                   Index : = i ;
                    for   j : = i + 1   to   N      do
                      if id [ j ] < Min  Then
                        Begin
                              min : = id [ j ] ;
                              Index : = j ;
                         End ;
                     id [ index ] : =  id [ i ]  ;
                     id [ i ] : = Min ;
                     for   j : = 1   to   4      do
                       Begin
                             Temp := Grade [ i , j ] ;
```



```
                             Grade [ i , j ] :   Grade [ index , j ] ;
                             Grade [ index , j ] : = temp ;
                       End ;
                End ; { end of sort }
                Writeln ;
          Writeln ( '   id   Grade1  Grade2  Grade3  average ' ) ;
          for   i : = 1   to   N      do
                Begin
                     Write ( id [ i ] : 8 );
                       for   j : = 1   to   4      do
                            Write ( Grade [ i , j ] : 8 : 2 ) ;
                        Writeln ;
                End ; { end of print }
      Writeln ;
      Writeln ( 'press any key to continue ... ' ) ;
      Readln ;
End . { End of program }
```

خروجی برنامه بالا بصورت زیر می‌باشد:

```
Please enter Number of student: 4
Enter id and Three Grades
7813195    17    18    10
7712171    16    14    15
7913176    12    8    10
7825176    13    16    10
Id   Grade1   Grade2   Grade3   average
7712171    16.00    14.00    15.00    15.00
7813195    17.00    18.00    10.00    15.00
7825179    13.00    16.00    10.00    13.00
7913179    12.00    18.00    10.00    10.00
press any key to continue
```

۸-۵- تمرینات

۱- به سوالات زیر پاسخ دهید:

الف) آرایه‌ای به طول 80 از نوع Boolean تعریف کنید.

ب) آرایه‌ای با محدوده کاراکتری 'A'-'Z' را از نوع اعشاری تعریف کنید.

ج) آرایه دو بعدی با ابعاد 50 * 50 از نوع Byte تعریف کنید.

د) آرایه سه بعدی با ابعاد 4 * 5 * 3 از نوع Longint تعریف کنید.

۲- کدامیک از دستورات زیر در مورد اعلان زیر صحیح است:

a: array ['A'..'Z'] of char

الف- a['A']:='Z'

ب- a['a']:='A'

ج- a['A']:=1

د- a['I']:=12

ح- a['i']:=12

۳- اعلانهای زیر را توصیف نمائید:

الف- a : array [Booleam] of integer

ب- a : array [1 .. 50 , Booleam] of Byte

ج- a : array ['A'..'Z', 1 .. 4] of Byte

د- a : array [1 .. 50] of array [2 .. 20] of double

ح- a : array [-20 .. 20 , 0 .. 10] of Byte

۴- تعداد بایتهایی که هر کدام از اعلانهای زیر اشغال می کنند را محاسبه نمائید.

الف- a: array [-20.. 10] of char

ب- a: array [-20.. -20 , 0.. -20] of 'A'..'Z'

ج- a: array [Booleam] of char

د- a: array [1.. 10 , 'A'..'Z'] of integer

۸-۶- تمرینات برنامه‌نویسی

۱ برنامه‌ای بنویسید که که یک آرایه حداکثر ۵۰ عنصری را از ورودی دریافت کرده و سپس عناصری از آرایه که اول هستند را با صفر جایگزین کرده آرایه حاصل را در خروجی چاپ کند.

۲- اطلاعات مربوط به حداکثر ۵۰ کارمند یک شرکت تولیدی عبارتند از:

حق الزحمه هر ساعت	تعداد ساعات کارکرد در ماه	کد کارمندی

درنظر بگیرید. برنامه‌ای بنویسید که:

الف- اطلاعات کارمندان را از ورودی دریافت کند.

ب- حقوق کارمندان پس از کسر کسورات قانونی که عبارتند از:

- بیمه 7 درصد
- مالیات ۱۰ درصد

را محاسبه نماید.

ج- اطلاعات کارمندان را بر اساس کد کارمندی مرتب نماید.

د- اطلاعات را با پیغام های مناسب در خروجی چاپ نماید.

۳- برنامه‌ای بنویسید که عددی از ورودی دریافت کرده سپس آن را به عامل هـای اول تجزیه نماید و حاصل را بصورت زیر در خروجی چاپ نماید:

برای مثال: 21 = (3 ^ 1) * (7 ^ 1)

۴- برنامه‌ای بنویسید که یک عدد از ورودی دریافت کرده سپس در صـورت وجـود صفرهای آن را حذف نموده، نتیجه را در خروجی چاپ نماید.

برای مثال: 100340 ⟵ 134

۵- برنامه ای بنویسد که یک ماتریس 5 * 5 را از ورودی دریافت کرده سپس مجموع هر سطر را انتهای همان سطر و مجموع هر ستون را در انتهای همان ستون چاپ نماید.

۶- برنامه‌ای بنویسید که یـک مـاتریس حـداکثر 10 * 10 را از ورودی دریافـت کـرده ماتریس هم بر حسب ستون و هم بر حسب سطر مرتب نموده به همراه مـاتریس اولیـه در یک سطر چاپ نماید.

۷- برنامه‌ای بنویسید که یک آرایه 200 عنصری ازنوع صحیح که 150 عنصرمرتب دران قرار می‌گیرد را از ورودی دریافت کرده سپس آرایه دومی با 50 عنصر را از ورودی بخواند.

آرایه دوم را طوری در آرایه اول ادغام کند که ترتیب آرایه اول به هم نخورد.

توجه: مجاز نیستید آرایه دوم را به آرایه اول اضافه کرده و عمل مرتب‌سازی انجام دهید.

۸- اطلاعات 100 دانشجو که عبارتند از:

شماره دانشجوئی	معدل	ارزشیابی

را در نظر بگیرید برنامه ای بنویسید که:

الف- شماره دانشجوئی و معدل دانشجویان را از ورودی دریافت نماید.

ب- آرایه ارزشیابی را بصورت زیر مقدار دهی نماید:

معدل	ارزشیابی
16 - 20	A
14 - 16	B
12 - 14	C
10 - 12	D
0 - 10	E

ج- معدل سوم کلاس را بهمراه تعداد تکرار و مشخصات دانشجویان که این معدل را کسب نموده اند را در خروجی چاپ نماید.

۹- گفته می شود که یک ماتریس M * N دارای نقطه زینی می باشد اگر عنصری مانند A [i , j] کوچکترین مقدار در سطر i ام و بزرگترین مقدار در ستون j ام ماتریس باشد. برنامه‌ای بنویسید که محل نقطه یا نقاط زین اسبی را در یک ماتریس M * N با علامت ستاره در صورت وجود مشخص نماید.

۱۰- برنامه ای بنویسید که عدد صحیح n را از ورودی دریافت کرده ماتریس با مشخصات زیر در خروجی چاپ نماید:

الف- n فرد است

ب- اعضای ماتریس بین 1 تا n^2 می باشد.

ج- مجموع هر سطر، هر ستون و هر قطر با هم برابر می باشد.

این ماتریس به مربع جادوئی معروف است.

فصل ۹

توابع و روالهای کتابخانه‌ای

هدفهای کلی

- شناخت ساختار تابع و روال
- شناخت توابع و روالهای استاندارد برای نوعهای صحیح
- شناخت توابع و روالهای استاندارد برای نوعهای اعشاری و کارکتری
- شناخت توابع و روالهای استاندارد ریاضی

هدفهای رفتاری

دانشجو پس از مطالعه این فصل باید بتواند:

- توابع و روالهای استاندارد را در صورت نیاز در برنامه استفاده کند.
- ساختار توابع و خروجیهای آنها را تشخیص داده و در برنامه خود از آنها استفاده کند.
- ساختار روالها و خروجیهای آنها را تشخیص داده و نتایج حاصل از آنها را مورد استفاده قرار دهد.
- توابع ریاضی را برای محاسبه عبارت ریاضی در برنامه بکار ببرد.

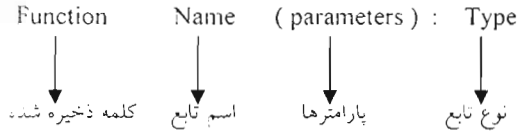
مقدمه

همانطور که قبلاً اشاره شد در زبان پاسکال توابع و روال‌های استانداردی که از قبل نوشته شده‌اند وجود دارد و استفاده از آنها کارائی برنامه را بالا می برد و می‌توان گفت بعضی مواقع گره برنامه‌نویسی را باز می‌کنند و در ضمن اشاره کردیم که خود برنامه‌نویس می‌تواند تابع یا روال‌های مورد نیاز برنامه را که در توابع استاندارد زبان نباشد بنویسد. در فصل‌های بعد خواهید دید که یک برنامه ساخت‌یافته باید به قطعات مجزا که هر جزء بعنوان تابع یا روال می‌باشد تقسیم‌بندی می‌شود. در زیر به تعدادی تابع و روال استاندارد اشاره می‌کنیم.

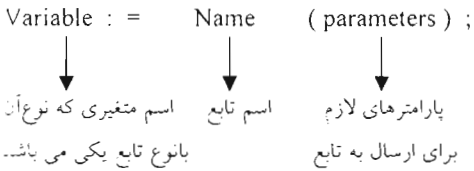
۹-۱- ساختار تابع

در کل هدف تابع اینست که متغیر یا متغیرهایی را بعنوان پارامتر از برنامه اصلی دریافت کرده و عمل خاصی را روی پارامترها انجام داده و نتیجه را به برنامه اصلی برگرداند. شکل کلی فراخوانی توابع در برنامه اصلی بصورت زیر می‌باشد:

ساختار تابع:



منظور از نوع تابع مقداری است که تابع به برنامه اصلی بر می‌گرداند. برای فراخوانی تابع در برنامه اصلی بصورت زیر عمل می‌کنیم:



همانطور که ملاحظه می‌کنید پارامترهای لازم برای تابع از طریق آرگومان تابع ارسال می‌کنیم وخروجی لازم توسط تابع به برنامه اصلی داخل یک متغیر ازنوع تابع ارسال می‌شود.

۹-۲- توابعی برای اعداد صحیح و اعشاری

در زیر توابعی که عملیات خاصی را روی اعداد صحیح و اعشاری انجام می‌دهند را ارائه می‌دهیم:

۹-۲-۱ تابع Abs

هدف: باز گرداندن قدر مطلق (absolute) پارامتری که به آن ارسال می‌شود:

شکل تابع:

```
Function  Abs ( x: Integer ): Integer ;
  Function  Abs ( x: Real ): Real ;
```

این تابع یک عبارت از نوع حقیقی یا صحیح را بعنوان آرگومان دریافت کرده سپس قدر مطلق آن را محاسبه و حاصل را برگرداند.

مثال ۹-۱

```
Var
    f : Real ;
    I : Integer ;
Begin
       F : = Abs ( - 191.15 ) ;
       I : = Abs ( - 171 ) ;
       Writeln ( ' F = ' , f : 8 : 2 , ' I = ' , I ) ;
 End .
```

خروجی حاصل :

```
F =  191.15    I =  171
```

۹-۲-۲- تابع Sin

هدف: باز گرداندن سینوس یک عدد از نوع اعشاری

شکل تابع:

```
Function   Sin ( X: Real ): Real ;
```

X یک عبارت یا عدد از نوع اعشاری بوده و حاصل مقدار سینوس X می‌باشد.

مثال ۲-۹

```
Var
   X : Real ;
Begin
    X : = sin ( 10 ) ;
    Write ( ' sin ( 10 ) = ', x :8 : 2 ) ;
End
```

۹-۲-۳- تابع Cos

هدف: باز گرداندن کسینوس یک عدد از نوع اعشاری.

شکل تابع:

```
Function   cos ( X: Real ): Real ;
```

X یک عبارت یا عدد از نوع اعشاری بوده و حاصل مقدار کسینوس X می‌باشد.

مثال ۳-۹

```
Var
   X : Real ;
Begin
   X : = cos ( 10 ) ;
   Write ( ' cos ( 10 ) = ', x : 8 : 2 ) ;
End .
```

۹-۲-۴- تابع ArcTan

هدف: بازگرداندن آرک تانژانت یک عدد از نوع اعشاری

شکل تابع:

```
Function   ArcTan ( X: Real ): Real ;
```

X یک عبارت یا عدد از نوع اعشاری بوده حاصل مقدار آرک تانژانت X می‌باشد.

مثال ۴-۹

```
Var
   X : Real ;
Begin
   X : = ArcTan ( 10 ) ;
   Write ( ' ArcTan ( 10 ) = ', x : 8 : 2 ) ;
End .
```

توجه: در توابع مثلثاتی ارائه شده در صورتی که X از نوع درجه باشد می‌توان با فرمول زیر معادل رادیان آن را محاسبه کرد:

```
Real = Deg * 3.14159 / 180
```


۹-۲-۵- تابع Exp

هدف: عدد نپر (e = 2.71828 ...) را به توان یک عدد می‌رساند.

شکل تابع:

```
Function   Exp ( X: Real ): Real ;
```

X عبارت یا متغیری از نوع اعشاری بوده و حاصل تابع نیز یک عدد اعشاری می‌باشد این تابع مقدار e^x را محاسبه می‌کند.

مثال ۵-۹

```
Var
   X : Real ;
   i : integer ;
Begin
      For   i : = 1   to   10   do
          Begin
            X : = exp ( i ) ;
            Writeln ( x : 8 : 2 ) ;
      End ;
   End .
```

۹-۲-۶- تابع frac

هدف: قسمت اعشاری یک عدد اعشاری را برمی‌گرداند.

شکل تابع:

```
Function   frac ( X: Real ): Real ;
```

X عددی از نوع اعشاری و حاصل تابع یک عدد اعشاری که قسمت اعشاری عدد X است می باشد و بعبارت دیگر این تابع قسمت اعشاری عدد ورودی را به عنوان خروجی باز می‌گرداند.

مثال ۶-۹

```
Var
     y , x : Real ;
Begin
     X : = frac ( 24.769 ) ;
     Y : = frac ( - 12.75 ) ;
     Write ( ' x = ', 8 : 3 , ' y = ', 8 : 2  ) ;
End .
```

خروجی :

```
X =    0.769   Y =    - 0.75
```

۹-۲-۷- تابع Int

هدف: قسمت صحیح یک عدد اعشاری را برمی‌گرداند.

شکل تابع: `Function Int ( X: Real ): Real ;`

X یک عبارت یا متغیر از نوع اعشاری و خروجی تابع نیز، یک عدد اعشاری است این تابع مقدار صحیح یک عدد اعشاری را در خروجی نشان می‌دهد.

مثال ۹-۷

```
Var
      y , x : Real ;
Begin
      X : = Int ( 2.87 ) ;        { 2.0 }
      Y : = Int ( - 8.76 ) ;      { - 8.0 }
End .
```

۹-۲-۸- تابع IoResult

هدف: وضعیت آخرین عملیات ورودی، خروجی انجام شده را بر می‌گرداند.

شکل تابع: `function IoResult : integer ;`

تابع فوق زمانی به کار می رود که بخواهیم وضعیت بروز خطا یا عدم بروز خطا را بعد از اعمال ورودی، خروجی(I / O) بررسی کنیم. اگر خروجی این تابع مقدار غیر صفر باشد عمل خواندن و نوشتن درفایل (بعداً بحث خواهد شد) با شکست مواجه شده است.

۹-۲-۹- تابع Ln

هدف: محاسبه لگاریتم یک عدد اعشاری در مبنای e.

شکل تابع: `Function Ln ( X: Real ): Real ;`

X یک عبارت یا متغیر از نوع اعشاری بوده و حاصل تابع نیز، یک عدد اعشاری می‌باشد.

مثال ۹-۸

```
Var
     X : Real ;
Begin
   X : = Ln ( 2.87 ) ;        { x = 3.73767 }
   Write ( ' x = ', x : 10 : 5 ) ;
End .
```


۹-۲-۱۰ تابع odd

هدف: فرد بودن عدد صحیح را بررسی می کند.

شکل تابع: `Function odd ( X: logint ): Booleam ;`

X یک عبارت از نوع longint است و تابع فرد بودن عبارت را بررسی می کند اگر مقدار خروجی تابع True باشد X فرد است و اگر مقدار خروجی تابع False باشد X فرد نیست.

مثال ۹-۹ برنامه‌ای بنویسید که اعداد فرد بین 1 تا 100 را در خروجی چاپ کند.

```
Var
    i : integer ;
Begin
     For   i : = 1   to   100   do
       If odd ( i ) Then
         Writeln ( ' i = ', j : 3 ) ;
End .
```

۹-۲-۱۱- تابع Ord

هدف: غالباً برای پیدا کردن کد اسکی یک متغیر کاراکتری بکار می‌رود.

شکل تابع: `Function Ord ( x: char ): Longint ;`

عبارت x از نوع کارکتری را بعنوان پارامتر دریافت کرده و کد اسکی آن را برمی گرداند. اگر x از نوع اسکالر باشد تابع بعنوان خروجی ترتیب قرار گرفتن x را در مجموعه ای که ابتدا به عنوان اسکالر اعلان شده، باز می‌گرداند.

مثال ۹-۱۰

```
Var
     ch : char ;
Begin
       For   ch : = ' A '   to   ' Z '   do
           Write ( ord ( ch ) : 5 ) ;
End .
```

کد اسکی ' A ' تا ' Z ' را درخروجی چاپ می‌کند کد اسکی ' A ' عدد 65 می‌باشد.

۹-۲-۱۲- تابع pi

هدف: عدد پی را بر می‌گرداند.

شکل تابع: Function pi: Real ;

این تابع برای بازگرداندن عددی پی (... 3.141592) مورد استفاده قرار می‌گیرد.

مثال ۹-۱۱ برنامه ای بنویسید که شعاع یک دایره را از ورودی دریافت کرده، سپس مساحت آنرا محاسبه و چاپ کند.

```
Var
      R : Real ;
Begin
      Readln ( R ) ;
      Writeln ( Area is equal - , ' ( pi * R * R ) : 10 : 2 ) ;
End .
```

۹-۲-۱۳- تابع Pred

هدف: مقدار قبل مقدار پارامتر را بر می گرداند.

شکل تابع: X ; Function pred (x): < same type of parameter > ;

پارامتر تابع می تواند از هر نوع باشد و با توجه به نوع پارامتر تابع نیز از همان نوع می‌باشد و خروجی تابع مقدار قبل از x می‌باشد.

مثال ۹-۱۲

```
Ch: = pred ( ' d ' ) ;   { ch = ' c ' }
i: = pred ( 15 ) ;   { i = 14 }
flay: = pred ( True ) ;   { flay = false }
i: = pred ( - 30 ) ;   { i = - 31 }
```

۹-۲-۱۴- تابع Random

هدف: برای تولید عدد تصادفی

شکل تابع:

1) Function Random: Real ;
2) Function Random (x: word): word ;

اگر تابع Random به شکل یک یعنی بدون آرگومان مورد استفاده قرار گیرد یک عدد تصادفی از نوع اعشاری بین صفر و یک تولید می کند و اگر به شکل دو بکار رود باعث تولید یک عدد تصادفی از نوع word که بزرگتر یا مساوی صفر و کوچکتر از x است خواهد شد.

در کل خروجی تابع بالا به صورت زیر است:

0 < = R < 1	-1
0 < = R < x	-2

مثال ۹-۱۳ برنامه ای بنویسید که 10 عدد تصادفی بین صفر و یک، و 10 عدد تصادفی بین 0 و 40 تولید نماید.

```
Var
    i : integer ;
Begin
     For   i : = 1   to   10   do
       Begin
            Writeln ( Random : 8 : 7 ) ;
            Writeln ( Random [ 40 ] : 8 ) ;
        End ;
End .
```

۹-۲-۱۵- تابع Round

هدف: برای گرد کردن اعداد اعشاری بکار می رود.

شکل تابع: Function Round (x: Real): Longint ;

X یک عبارت یامتغیر اعشاری بوده وخروجی تابع یک عدد ازنوع Longint می‌باشد که نتیجه گرد کردن X می‌باشد. این تابع X را به نزدیکترین مقدار صحیح گرد می‌کند.

مثال ۹-۱۴

```
i : = Round ( 57.4 )        { i = 57 }
i : = Round ( 59.5 )        { i = 60 }
i : = Round ( 12.7 )        { i = 13 }
i : = Roun ' ( 12.25 )      { i = 12 }
i : = Round ( 17.75 )       { i = 18 }
i : = Round ( 17.45 )       { i = 18 }
i : = Round ( -2.5 )        { i = -3 }
```

۹-۳-۱۶- تابع sqr

هدف: برای محاسبه مجذور یک عدد صحیح یا اعشاری بکار می‌رود.

شکل تابع:

```
Function  sqr ( x: Integer ): Integer ;
Function  sqr ( x: Real ): Real ;
```

X عبارتی یا متغییری از نوع صحیح یا اعشاری بوده و خروجی تابع نیز، یک عدد صحیح یا اعشاری می‌باشد این تابع مجذور X را بعنوان خروجی بر می‌گرداند.

مثال ۹-۱۳

```
Var
      i : integer ;
Begin
      For   i : = 1   to   10   do
        Writeln ( ' I ^ 2 = ' , sqr ( I ) ) ;
End .
```

۹-۳-۱۷- تابع sqrt

هدف: برای محاسبه جذر یک عدد بکار میرود.

شکل تابع:

```
Function  sqrt ( x: Real ): Real ;
```

X یک عبارت از نوع اعشاری بوده و خروجی تابع نیز، یک عدد اعشاری می‌باشد این تابع جذر X را بعنوان خروجی بر می‌گرداند.

توجه: X نمی تواند مقدار منفی داشته باشد.

مثال ۹-۱۶

```
Var
     Y , X : Real ;
Begin
     Y : = sqrt ( 64 ) ;          { y = 8.0 }
     Z : = sqrt ( 0.16 ) ;        { z = 0.4 }
     Write ( ' x = ' , x : 8 : 2 , ' y = ' , y : 8 : 2 ) ;
End .
```

۹-۳-۱۸- تابع suce

هدف: مقدار بعد از مقدار فعلی را برمی گرداند.

شکل تابع:

```
Function  succ ( x ): same  type  of  parameters ;
```


X یک عبارت از نوع صحیح، اولین و غیره بوده و خروجی تابع نیز از همان نوع X می باشد این تابع مقدار بعد از X را بعنوان خروجی بر می‌گرداند.

مثال ۹-۱۷

```
ch : = succ ( ' a ' )        { ch = ' b ' }
ch : = succ ( ' A ' )        { ch = ' B ' }
 i : = succ ( 15 )           { i = 16 }
flay : = succ ( false )      { flay = True }
flay : = succ ( True )       { تعریف نشده }
```

۹-۳-۱۹- تابع Trunc

هدف: قسمت صحیح یک عدد اعشاری را بر می‌گرداند.

شکل تابع:

```
Function  Trunc ( x: Real ): Longint ;
```

X یک عبارت یا متغیر از نوع اعشاری بوده و خروجی تابع یک عدد از نوع Longint می‌باشد. این تابع قسمت صحیح عدد اعشاری X را بعنوان خروجی بر می‌گرداند.

مثال ۹-۱۸

```
Var
    x , i , j : Longint ;
Begin
      i : = trunc ( 12.5 ) ;      { i = 12 }
      j : = trunc ( 12.4 ) ;      { i = 12 }
      k : = trunc ( 13.5 ) ;      { k = 13 }
      Writeln ( ' I - ' , i : 5 , ' j = ' , j : 5 , ' k = ' , k : 5 ) ;
End .
```

۹-۴- توابع از نوع کاراکتری

در این بخش توابعی را بررسی می کنیم که خروجی آنها از نوع کاراکتری باشد.

۹-۴-۱- تابع chr

هدف: معادل کاراکتری یک کد اسکی را بر می‌گرداند.

شکل تابع:

```
Function   chr ( X: Byte ): char ;
```

X یک عبارت یا متغیر از نوع بایت بوده و خروجی تابع یک کاراکتر می‌باشد. این تابع کد اسکی را دریافت کرده و معادل کاراکتری آن را بر می‌گرداند.

مثال ۹-۱۹

```
Var
     i : Byte ;
Begin
     For   i : = 15   to   91   do
     Writeln ( chr ( I ) ) ;
End .
```

برنامه بالا حروف A تا Z را در خروجی چاپ می‌کند.

۹-۴-۲- تابع Upcase

هدف: برای تبدیل یک کاراکتر به حرف بزرگتر بکار می‌رود.

شکل تابع:

```
Function   Upcase ( ch: char ): char ;
```

ch یک عبارت یا متغیر از نوع کاراکتر بوده و خروجی تابع نیز، یک کاراکتر می‌باشد این تابع حرف کوچک را به حرف بزرگ تبدیل کرده و بعنوان خروجی حروف بزرگ را بر می‌گرداند.

مثال ۹-۲۰ برنامه ای بنویسید که یک جمله حداکثر 80 کاراکتری را از ورودی دریافت کرده حروف کوچک آن را به حروف بزرگ تبدیل نماید.

```
Program   Example 20 ;
Var
                Sen: array [ 1.. 80 ] of char ;
                i , N: Byte ;
Begin
                Write ( ' enter  Numbers: ' ) ;
                Readln ( N ) ;
                For   i: = 1   to   N   do
                     Read ( Sen [ i ] ) ;
                Writeln ;
                For   i: = 1   to   N   do
                   If ( sen [ i ] > = ' a ' )  and  ( sen [ i ] < = ' z ' ) Then ;
                        sen [ i ]: =  Upcase ( sen [ i ] ) ;
                 Writeln ( ' Output  of  program ' ) ;
                 For   i: = 1   to   N   do
                     Writeln ( sen [ i ] ) ;
End. { End  of  program }
```


۹-۵- روال‌های استاندارد

در بخش‌های قبل دیدید که در توابع پارامترها به تابع ارسال می‌شود و تابع نیز مقداری را بعنوان خروجی برمی‌گرداند روالها نیز مشابه توابع عمل می‌کنند با این تفاوت که خروجی روالها از طریق پارامتر برمی‌گردانده می‌شود یا اعلانی به سیستم عامل می‌باشد. بنابراین روالها بدون نوع هستند. و نمی‌توان برای روالها نوع تعریف کرد. شکل کلی روال‌ها بصورت زیر می‌باشد:

```
Procedure   Name ( Parameters ) ;
```

پارامترهای روال — اسم روال — کلمه ذخیره شده

در معرفی یک روال، پارامتر به دو صورت زیر می تواند ظاهر شود:

الف) به صورت مقدار

ب) به صورت متغیری

اگر بصورت مقدار ظاهر شود تغیراتی که در پارامتر داخل روال داده می شود به خارج از روال انتقال نمی‌یابد ولی اگر بصورت متغیری بکار برده شود تغییرات به خارج از روال ارجاع داده می شود. بحث در مورد پارامترها را در فصل‌های آتی دنبال خواهیم کرد.

۹-۵-۱- روال Dec

هدف: یک یا چند واحد از پارامتر ارسالی کم می کند.

شکل روال:

```
Procedure  Dec ( Var  X: longint ) ;
Procedure  Dec ( Var  X: longint , N: longint ) ;
```

X یک متغیر از نوع longint و بصورت متغیری می‌باشد این روال یک واحد از پارامتر ارسالی کم می‌کند.

مثال ۲۱-۹

```
Var
      N : integer
Begin
      N : = 1201 ;
      Dec ( N ) ;       { N = 1200 }
      Writeln ( N ) ;
      Dec ( N , 200 ) ;       { N= 1000 }
      Writeln ( N ) ;
End .
```

۹-۵-۲- روال Exit

هدف: کنترل برنامه را به خارج از بلوک جاری منتقل می کند.

شکل روال:

```
Procedure Exit ;
```

این روال باعث می شود که کنترل برنامه از بلوک جاری خارج شود. اگر این روال در برنامه اصلی بکار رود باعث خروج از برنامه می‌شود. و اگر در یک روال یا تابع بکار رود باعث خروج از روال یا تابع شده و کنترل برنامه به برنامه اصلی منتقل می‌شود.

مثال ۲۲-۹

```
Var
      i , j : integer ;
Begin
      i : = 100 ;
      j : = 20 ;
      Dec ( i , j ) ;
      Write ( i ) ;
      Exit ;
End .
```

بعد از اتمام عملیات تابع Exit باعث خروج از برنامه می شود.

۹-۵-۳- روال Halt

هدف: خاتمه دادن به اجرای برنامه

شکل روال:

```
Procedure Halt ;
```

این روال باعث خاتمه اجرای برنامه شده و کنترل برنامه به سیستم عامل برمی‌گردد.

۹-۵-۴- روال Inc

هدف: اضافه کردن یک یا چند واحد به یک متغیر

شکل روال:

```
Procedure Inc ( Var X: longint ) ;
Procedure Inc ( Var X: longint , N: longint ) ;
```

X یک عبارت یا متغیر از نوع Longint می باشد این روال به مقدار متغیر x یک یا چند واحد اضافه می‌کند.

مثال ۲۳-۹

```
Var
     N , i , j : integer ;
Begin
      i : = 100 ;
      j : = 200 ;
      N : = 10 ;
      inc ( i ) ;      { i = 101 }
      inc ( j , N ) ;      { j = 210 }
      Writeln (' i = ' , i , 'j = ' , j ) ;
 End .
```

۹-۵-۵- روال Randomize

هدف: باعث تغییر نحوه تولید اعداد تصادفی می شود.

شکل روال:

```
Procedure Randomize ;
```

وقتی در برنامه از تابع Random استفاده می‌کنیم اعداد تصادفی تولید شده در اجراهای مختلف یکسان می باشد برای جلوگیری از این وضعیت قبل از استفاده از تابع Random روال Rondomize را بکار می‌بریم. تا باعث تولید اعداد تصادفی متفاوت در اجراهای مختلف گردد.

مثال ۲۴-۹

```
Var
      i : word ;
Begin
      Randomize
      For   i : = 1   to   10   do
            Writeln ( Random ( 50 ) ) ;
End .
```

توابع و روال های ارائه شده تعدادی از توابع و روال های استاندارد زبان توربوپاسکال بودند که در برنامه‌نویسی می‌تواند کمک زیادی به نوشتن برنامه داشته باشد.خوانندگان می توانند در تمرینات فصل‌های آتی با استفاده از این توابع راحت‌تر برنامه بنویسند.

۹-۶- حل چند مثال برنامه‌نویسی

در این بخش چند برنامه ارائه می دهیم تا کاربرد توابع و روال ها را مشاهده کنید.

مثال ۹-۲۵ برنامه ای بنویسید که یک ماتریس مربع n * n (n < = 10) از مقادیر صحیح را از ورودی دریافت کرده آنگاه عناصری که مربع کامل هستند را صفر کرده و در نهایت ماتریس حاصل را در خروجی چاپ کند.

```
Program   Example 25 ;
Var
            n , i , j : integer
            a : array [ 1 .. 10 , 1 .. 10 ] of integer ;
Begin
             Write ( ' enter  Number : ' ) ;
             For   i : = 1   to   N   do
                For   j : = 1   to   N   do
                   Read ( a [ i , j ] ) ;
             For   i : = 1   to   N   do
                For   j : = 1   to   N   do
                   If  a [ i , j ] sqr ( Trnnc (sqrt (Abs ( a[ i , j ] ))) ) < > 0 Then
                         a [ i , j ] : = 0 ;
             Writeln ( ' The oupput  Mafrix ' ) ;
             For   i : = 1   to   N   do
                  Begin
                        For   j : = 1   to   N   do
                            Write ( a [ i , j ] : 5 ) ;
                       Writeln  ;
                  End ;
End . { End  of  program }
```

خروجی برنامه بالا بصورت زیر است :

```
Enter  Number : 3
3   4   9   6   7   10   12   16   25
```



```
The  output  Matrix
3    0    0
6    7    10
12   0    0
```

مثال ۹-۲۶ برنامه‌ای بنویسید که یک جمله از ورودی دریافت کرده سپس حروف بزرگ جمله را به حروف کوچک تبدیل نمائید.

(جمله حداکثر 80 کارکتر)

```
Program   Example 26 ;
Var              sen : array [ 1 .. 80 ] of char ;
                 Ch : char ;
                 i : word ;
Begin
               Write ( 'Enter  sentence ' ) ;
               Repeat
                      Read ( sen [ i ] ) ;
                       Inc ( i ) ;
               Until  sen [ i ] = '.' ;
               Writeln  ;
               n : = Dec ( i ) ;
               For   i : = 1   to   n   do
                  If  ( sen [ i ] > = ' A ' ) and  ( sen [ i ] < = ' B ' )  Then
                  sen [ i ] = chr ( ord ( sen [ i ] ) + ( ord ( ' a ' ) – ord ( ' A ' )) )
               Writeln ( ' The result  sentence ' ) ;
                  For   i : = 1   to   n   do
                        Write ( sen [ i ] ) ;
End . { End  of  program }
```

خروجی حاصل از برنامه بالا بصورت زیر است :

```
Enter  sentence
This  Book  is  Pascal Programmning .
The  result  sentence
this  book  is  pascal programmning
```

۷-۹- تمرینات

۱- خروجی تمرینات زیر را تعیین کنید:

الف)

```
Var
        i , j : integer
Begin
        i : = 100 ;
        j : = 20 ;
        inc ( i ) ;
        Dec ( j , 10 ) ;
        inc ( i , j ) ;
        Writeln ( i : 5 , j : 5 ) ;
End .
```

ب)

```
Var
      y : integer ;
Begin
       y : Round ( 18.31 ) MoD 5 ;
       Writeln ( y ) ;
End .
```

ج)

```
Var
      y : integer ;
Begin
       y : = sqr ( Round ( sqrt( 12.81 ) ) ;
       Writeln ( y ) ;
End .
```

د)

```
Var
      X : Real ;
      i , j : integer ;
Begin
      J : = 12 ;
      For   i : = 1   to   10   do
          Begin
                X : = Random ( j ) ;
                Writeln ( x : 8 : 7 ) ;
              J : = j + i ;
           End ;
End .
```


۲- خروجی دستورات زیر را محاسبه نمائید:

الف)

```
Ch: = ' A ' ;
Ch: = chr ( ord ( ch ) + 3 ) ;
Write ( ch ) ;
```

ب)

```
X: = 12 ;
 Suce ( x ) ;
Ch: = ' B ' ;
Pred ( ch ) ;
Write ( x , ch ) ;
```

ج)

```
i: = Trunc ( 15.75 ) ;
j: = Trunc ( - 42.5 ) ;
k: = Trunc ( - 21.3 ) ;
Write ( i , j , k ) ;
```

د)

```
R1: = frac ( 24.35 ) ;
R2: = frac ( - 20.35 ) ;
R3: = frac ( - 12.45 ) ;
Write ( R1 , R2 , R3 ) ;
```

۸-۹- تمرینات برنامه‌نویسی

۱- برنامه‌ای بنویسید که تعداد ۱۰۰۰ شماره حساب بانکی ۷ رقمی بطور تصادفی بین ۵۱۱۹۴۳۲ و ۹۹۸۱۷۱۱ را تولید کرده و چاپ کند.

۲- برنامه ای بنویسید تا تعداد ۳۸ عدد تصادفی صحیح بین ۱۰ تا ۹۹ را تولید کرده، سپس آن را در خروجی به شکل مربع وسط صفحه نمایش چاپ کند.

۳- برنامه‌ای بنویسید که یک عدد ۲۰ رقمی از ورودی دریافت کرده آن را در یک عدد تک رقمی ضرب نموده و حاصل را در خروجی چاپ نماید.
توجه کنید که عدد ۲۰ رقمی را در یک آرایه ۲۰ عنصری از نوع charقرار دهید سپس عمل ضرب را با تبدیل آرایه به آرایه از نوع صحیح انجام دهید.

۴- برنامه‌ای بنویسید که دو عدد ۴۰ رقمی را از ورودی دریافت کرده سپس مجموع دو عدد را محاسبه و چاپ کند.

۵- قرعه‌کشی حسابهای قرض الحسنه بانکی را در نظر بگیرید، برنامه ای بنویسید که:
الف) شماره حساب و نقدینگی تعدادی مشتری را از ورودی دریافت نماید.
ب) امتیاز هر مشتری را با توجه به نقدینگی محاسبه کند (هر ۵۰۰۰۰ ریال یک امتیاز محسوب می‌شود)
ج) سپس تعداد ۵۰ سکه را قرعه کشی نماید.
توجه کنید که شماره حساب ها ۷ رقمی می باشد و بین ۹۰۰۰۰۰۰ تا ۱۰۰۰۰۰۰ قرار دارند. در محاسبه امتیاز بازای هر امتیاز باید شماره حساب مشتری در قرعه کشی یکبار تکرار شود مثلاً اگر مشتری ۵۰ امتیاز داشته باشد باید ۵۰ بار شماره حساب این مشتری در لیست قرعه کشی تکرار شده باشد تا امکان برد آن بیشتر باشد.

۶- برنامه‌ای بنویسید که یک جمله حداکثر ۸۰ کارکتری را از ورودی دریافت کند سپس با دریافت یک کارکتر جدید، جمله را با این کارکتر بصورت زیر کد نماید:
- فاصله کد اسکی کارکتر اول جمله را از کارکتر جدید محاسبه نموده به هر یک از کارکتر جمله اضافه نماید.
- اگر بعد از اضافه نمودن به کد اسکی کارکتر از رنج حروف بین a - z یا A - Z خارج شود آن را با اضافه یا کم کردن عددی به حروف بر گردانیم.
- هنگام کد کردن فقط با حروف الفبا کار خواهیم داشت و بقیه کاراکتر دست نخورده باقی می‌مانند.

۷- برنامه‌ای بنویسید که یک عدد در مبنای ۲ را از ورودی دریافت کرده و آنرا به مبنای ۱۶ ببرد.

فصل ۱۰

متغیرهای کاراکتری و رشته‌ها (String)

هدفهای کلی

- بررسی ساختار آرایه‌هایی از نوع کاراکتر
- معرفی نوع دادهٔ جدید به نام رشته
- مقایسه آرایه‌ای از کاراکتر و رشته
- شناخت توابع و روالهای استاندارد برای رشته‌ها

هدفهای رفتاری

دانشجو پس از مطالعه این فصل باید بتواند:

- آرایه‌ای از کاراکترها را در برنامه بکار ببرد.
- آرایه‌ای از کاراکترها را با رشته مقایسه کند.
- اسمی افراد و غیره را با استفاده از آرایه‌ای از رشته مرتب نماید.
- توابع و روالهای استاندارد مربوط به رشته‌ها را در برنامه خود بکار ببرد.

مقدمه

در فصل‌های قبل متغیرهای از نوع کاراکتر را بکار بردیم ولی تا این لحظه نتوانستیم که اسم و فامیلی تعدادی دانشجو را در آرایه قرار داده و مرتب کنیم. اگر دقت کنید نیاز به نوع داده جدیدی داریم که بتوانیم یک لیست از اسامی افراد، آدرس و غیره تهیه نموده و اعمالی را با آنها انجام دهیم، این نوع جدید را در این فصل معرفی کرده و بحث می کنیم. این متغیرها را متغیرهای رشته‌ای نام خواهیم داد و از آنها برای پردازش متن، حروف و غیره استفاده خواهیم کرد.

۱-۱۰- متغیرهایی از نوع کاراکتر

این متغیرها را در فصل قبل بحث کردیم. در اینجا لازم دیدیم نکات مهمی را در مورد این متغیر ها اشاره کنیم. متغیرهای کاراکتری ظرفیت پذیرش یک کاراکتر (شامل یک رقم، یک حرف از حروف و یا یک کاراکتر دیگر) را دارا می‌باشند.

مقدار دهی این متغیر ها به صورت زیر می‌باشد:

```
ch: = ' A ' ;
```

اگر بخواهیم متغیرهای کاراکتری را از ورودی بخوانیم باید دقت بیشتری به خرج دهیم برای اینکه فضاهای خالی کاراکتر محسوب می شوند و همچنین بکار بردن Enter در جای مناسب نیاز به دقت زیادی دارد.

مثال ۱-۱۰

```
Var
     i , j : integer ;
    ch , c : char ;
      .
      .
      .
Begin
     Read ( i , j , ch , c ) ;
      .
      .
      .
End .
```

برای وارد کردن ورودیها چه باید کرد ؟ آیا می‌توان بصورت زیر عمل کرد:


```
10  17 A B        یا
10  17A  B        یا
10  17AB
{10  17
 AB
```

هر چهار روش بالا برای وارد کردن اطلاعات نادرست هستند. روش‌های دوم و سوم هنگام اجرا با خطا مواجه می‌شوند (خطای نوع بوجود می آید) در روش اول داخل متغیر ch فضای خالی قرار می‌گیرد و داخل متغیر c، کاراکتر ' A ' قرار خواهد گرفت. در روش چهارم نیز در متغیر ch ، کاراکتر Enter و در متغیر c ، کاراکتر ' A ' قرار خواهد گرفت لذا هر چهار روش بالا برای خواندن متغیرهای کاراکتر نادرست هستند. یک روش ساده برای جلوگیری از این موارد اینست که اول متغیرهای کاراکتری را بخوانیم بعد متغیرهای دیگری از ورودی خوانده شود. همچنین می‌توانیم دستورات ورودی جداگانه برای خواندن متغیرهای کاراکتری بکار ببریم.

دستور خواندن قطعه برنامه بالا را می تواند بصورت تصحیح کرد:

```
Read ( ch , c , i , j )
```

ورودی اطلاعات :

```
AB  10   17
```

```
{AB
 10   17
```

همچنین می‌توان دستور خواندن را بصورت زیر تصحیح کرد:

```
Readln ( ch , c )
Read ( i , j )
```

ملاحظه می‌کنید که هنگام خواندن متغیرهای کاراکتری باید دقت بیشتری به خرج دهیم تا اطلاعات بصورت صحیح در متغیرهای مربوط به خود قرار گیرند.

متغیرهای کاراکتری را می‌توان مانند متغیرهای معمولی با هم مقایسه کرد و عملگرهای مقایسه‌ای را در مورد این متغیرها می‌توان استفاده کرد.

مثال ۲-۱۰ خروجی قطعه برنامه زیر را تعیین کنید.

```
Ch: = ' A ' ;
C: = ' a ' ;
If  ch = c  Then
    Writeln ( ' equal ' )
Else
    Writeln ( ' Not  equal ' ) ;
```

خروجی حاصل از قطعه برنامه بالا بصورت زیر است:

```
Not  equal
```

چون کد اسکی 'A' با کد اسکی کاراکتر 'a' متفاوت می‌باشد لذا خروجی بالا تولید می‌شود.

درکل می‌توان کاراکترها را از لحاظ اردینال (مرتبه) بصورت زیر دسته‌بندی کرد:

۱- رقم ها بصورت زیر مقایسه می‌شوند:

'0' < '1' < '2' < '3' < ... < '9'

رقم‌ها از لحاظ کد اسکی پشت سر هم قرار گرفته‌اند.

۲- حروف بصورت زیر مقایسه می شوند:

'A' < 'B' < ... < 'Z'< ... <'a' < 'b' < ... < 'z'

برای درک بیشتر کد اسکی‌کاراکترها و بررسی مقدار کد اسکی کاراکترها می‌توان از مثال زیر کمک گرفت.

مثال ۳-۱۰ برنامه‌ای بنویسید که معادل کاراکتری، کد اسکی از ۱ تا ۱۵۰ را تولید و در خروجی چاپ نماید.

```
Var
      i: integer ;
     ch: Char ;
 Begin
          For   i: = 48   to   122   do
             Bigen
                  Write ('       ', i , ')', chr ( i ):2) ;
                   If  i Mod  10 = 0  Then
                        Writeln;
              End ;
End.
```

خروجی قطعه برنامه بالا بصورت زیر است:

```
48) 0   49) 1   50) 2   51) 3   52) 4   53) 5   54) 6   55) 7   56) 8   57) 9
58) :   59) ;   60) <   61) =   62) >   63) ?   64) @   65) A   66) B   67) C
68) D   69) E   70) F   71) G   72) H   73) I   74) J   75) K   76) L   77) M
78) N   79) O   80) P   81) Q   82) R   83) S   84) T   85) U   86) V   87) W
88) X   89) Y   90) Z   91) [   92) \   93) ]   94) ^   95) _   96) `   97) a
98) b   99) c  100) d  101) e  102) f  103) g  104) h  105) I  106) j  107) k
108) l  109) m  110) n  111) o  112) p  113) q  114) r  115) s  116) t  117) u
118) v  119) w  120) x  121) y  122) z
```


مثال ۴-۱۰ خروجی قطعه برنامه زیر را تعیین کنید:

```
Var
      c , ch : char ;
Begin
       c : = ' ' ;
       ch : = ' " ' ;
.
.
End.
```

دستورهای بالا فضای خالی (spase) و آپستروف را بعنوان کاراکتر در متغیرهای کاراکتری جایگزین می کنند.

۱-۱-۱۰- آرایه‌ای از کاراکتر

قبلاً آرایه‌ای از کاراکتر بکار بردیم آرایه از کاراکتر بصورت زیر معرفی می‌شود:

```
Name     :  array     [ 1 ... length ]     of     char
 ↓           ↓             ↓                 ↓
اسم آرایه   کلمه ذخیره شده   طول          نوع کاراکتری
```

حال با ارائه چند مثال آرایه ای از کاراکترها را بررسی می کنیم.

مثال ۵-۱۰ برنامه ای بنویسید که یک جمله (حداکثر ۸۰ کاراکتر) از ورودی دریافت کرده، تعداد حروف و تعداد کلمه‌های آن را شمرده و در خروجی چاپ نماید.

می دانیم که هر جمله به نقطه ختم می شود، همچنین کلمه ها با یک فاصله از هم جدا می‌شوند.

```
Var
       Sen: array [ 1 ... 80 ]  of  char ;
       n , i , Word_Count : word ;
Begin
       Word_Count: = 0 ;
       i: = 0 ;
       writeln ( ' Enter  sentence ' ) ;
       Repeat
            inc ( i ) ;
           Read ( sen [ i ] ) ;
       Until  sen [ i ] = '.' ;
       Dec ( i ) ;
       n: = i ;
       for   i: = 1    to    n    do
           if  sen [ i ] = ' '  Then
               inc ( Word_Count ) ;
       Writeln ;
```

```
        Writeln ( ' Word Count = ' , Word_Count , ' Character Count = ' , n ) ;
End. { End of program }
```

خروجی حاصل از برنامه بالا بصورت زیر است:

```
Enter sentence
This Book is written By Tanha && Usef.
Word Count = 8 Character Count = 38
```

برای مقداردهی اولیه آرایه ای از کاراکتر بصورت زیر نیز می‌توان عمل کرد:

```
Name: array [1 ...10 ] of char;
  Name: = ' TANHA ' ;
```

در اینصورت رشته بالا بترتیب در خانه های آرایه قرار می گیرد و عناصر آرایه بصورت زیر خواهد بود:

T	A	N	H	A					

مثال ۶-۱۰ به دو آرایه از نوع کاراکتر توجه کنید.

```
Var
        C , ch: array [ 1 ...8 ] of char ;
Begin
        C : = 'computer' ;
        ch: = 'comput' ;
        ch [ 7 ]: = 'e' ;
        ch [ 8 ]: = 'r' ;
        if C = ch Then
            Writeln ( ' Equal ' )
        Else
            Writeln ( ' Not Equal ' ) ;
End.
```

Equal در خروجی چاپ می شود.

حال با تغییرات جزئی در مثال ۶-۱۰ خروجی را بررسی می‌کنیم:

```
ch [ 7 ]: = 'E' ;
ch [ 8 ]: = 'R' ;
```

در خروجی Not Equal چاپ می شود. چون کد اسکی کاراکترهای کوچک با بزرگ متفاوت است.

اگر مثال ۶- ۱۰ را بصورت زیر تغییر دهیم:

```
C: = ' computer' ;
ch: = 'computer' ;
```

در اینصورت Not Equal در خروجی چاپ خواهد شد.

مثال ۷-۱۰ برنامه‌ای بنویسید که یک پاراگراف ۵ خطی را که هر خط آن حداکثر ۶۰ کاراکتر است را از ورودی دریافت نماید.

```
Var
        i , j: integer ;
        paragraph: array [ 1.. 5 , 1.. 60 ] of char ;
Begin
            for   i: = 1    to   5    do
              Begin
                    j: = 0 ;
                    while Not Eoln Do
                      Begin
                            Inc ( j ) ;
                            Read (paragraph [ i , j ] )
                      End ;
                    Readln ;
              End ;
End.
```

برنامه بالا ۵ خط از یک پاراگراف را از ورودی دریافت می کند هر خط حداکثر ۶۰ کاراکتر می باشد و به Eolnختم می شود. Eoln یک تابع می‌باشد.

تابع Eoln (End of line) بدون آرگومان بوده و مقدار اولیه آن False می‌باشد با زدن دکمه Enter توسط کاربر هنگام ورود اطلاعات مقدار تابع True خواهد شد.

۱۰-۲-۱- آرایه‌های فشرده (Packed Array)

در پاسکال استاندارد هر کاراکتر داخل یک عنصر از آرایه قرار می گیرد. حال آنکه هر عنصر آرایه ظرفیت پذیرش حداقل ۴ کاراکتر (حداکثر ۱۰ کاراکتر) را دارا می‌باشد. که این ممکن است مشکل کمبود فضای حافظه را ایجاد کند لذا برای جلوگیری از اشغال حجم بزرگی از حافظه، آرایه های کاراکتری فشرده را تعریف می‌کنیم. در این حالت ظرفیت پذیرش هر عنصر از یک کاراکتر بیشتر نیست و بنابراین بدین صورت می‌توانیم از حافظه استفاده موثرتری داشته باشیم. به مثال زیر توجه کنید:

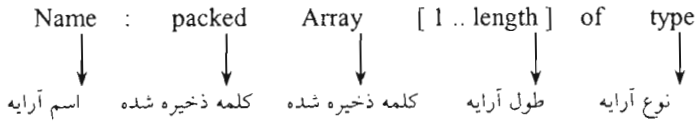
			A
			L
			I

آرایه معمولی

A
L
I

آرایه فشرده

آرایه‌های فشرده بصورت زیر تعریف می‌شوند:



حسن آرایه‌های فشرده استفاده از حجم حافظه کمتر است.

توجه: در توربو پاسکال بطور اتوماتیک آرایه‌ها بصورت فشرده درنظر گرفته می‌شود و نیاز به تعریف مجدد آنها بصورت فشرده نیست.

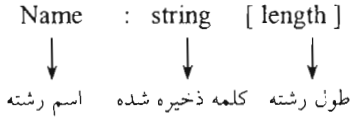
۲-۱۰- متغیرهای رشته‌ای (String)

نوع دیگری از از متغیرها در زبان پاسکال، متغیرهای رشته‌ای می‌باشند. متغیرهای رشته‌ای بصورت زیر تعریف می‌کنند:

تعریف: آریه‌ای از کاراکترها را یک رشته نامیده و متغیری از نوع آن را یک متغیر رشته‌ای می‌نامند.

این متغیر نیز مانند سایر متغیرها در قسمت تعاریف متغیرها (Var) معرفی می‌شود. در تعریف یک متغیر رشته‌ای معمولاً طول آن را مشخص می‌کنند. اگر طول تعیین نشود بطور قراردادی کامپایلر حداکثر طول را برای آن در نظر می‌گیرد. حداکثر طول رشته ۲۵۵ کاراکتر می‌باشد.

نحوه تعریف متغیر رشته ای بصورت زیر می‌باشد:



برای مثال S1 , S2 را در نظر بگیرید:

```
Var
      S1: string [ 30 ] ;
      S2: string [ 40 ] ;
      S: string
```


S1 یک رشته به طول ۳۰ کاراکتر، S2 رشته ای به طول ۴۰ و S رشته‌ای به طول ۲۵۵ کاراکتر تعریف می‌شوند.

متغیرهای رشته‌ای را بصورت زیر می‌توان مقداردهی کرد:

```
Var
      S : string [ 10 ] ;
        .
        .
        .
   Begin
          .
          .
        S : = ' pascal ' ;
        Write ( S ) ;
          .
          .
   End .
```

برای خواندن متغیرهای‌رشته‌ای ازدستورRead یا Readln بصورت زیرمی‌توان‌استفاده‌کرد:

```
Read ( s ) ;
Or Readln(s);
```

هنگام خواندن متغیرهای رشته‌ای اگر طول رشته ورودی از طول تعریف شده بیشتر باشد فقط به اندازه طول تعریف شده خوانده می شود و اگر طول رشته ورودی کمتر از طول تعریف شده باشد، رشته ورودی در منتهی‌الیه سمت چپ متغیر رشته‌ای قرار گرفته و بقیه متغیر رشته ای بدون محتوا باقی می ماند، بطوریکه اگر متغیر رشته‌ای توسط دستور خروجی چاپ شود متغیر رشته‌ای طول موثر خود را چاپ خواهد کرد (طول موثر آن قسمتی از طول تعریف شده است که در دستور ورودی پر شده است).

برای چاپ یک متغیر رشته‌ای نیز ازدستور Write یا Writeln می‌توان استفاده کرد:

```
Write ( s ) ;
Or Writeln(s);
```

مثال ۱۰-۸ برنامه‌ای بنویسید که شماره دانشجوئی، اسم و فامیل دانشجوئی را از ورودی دریافت کرده سپس در خروجی نمایش دهد.

```
Var
        Name , family: string [ 40 ] ;
         Id: longint
Begin
        writeln ( ' Enter  student  number ' ) ;
        Readln ( Id ) ;
        writeln ( ' Enter  Name ' ) ;
        Readln ( Name ) ;
        writeln ( ' Enter  family ' ) ;
```

```
        Readln ( family ) ;
        writeln ( '  Id     Name     Family ' ) ;
        writeln ( Id: 7 , Name , Family ) ;
End.
```

برنامه بالا در سطر های جداگانه شماره دانشجوئی، اسم و فامیلی را از ورودی خوانده سپس آنها را در یک سطر چاپ می‌کند.

اگر بخواهیم اسم تعدادی دانشجو کارمند و غیره را در یک آرایه قرار دهیم می‌توانیم از آرایه‌ای از نوع رشته استفاده کنیم.

آرایه‌ای از رشته بصورت زیر تعریف می‌شود:

```
Name  :  Array  [ 1 .. length ]  of  string
 ↓                    ↓              ↓
اسم آرایه          طول آرایه      نوع رشته ای
```

برای مثال اگر بخواهیم آرایه‌ای برای ورود اسم ۵۰ نفر تعریف کنیم، بصورت زیر عمل می‌کنیم:

```
Name  :  Array  [ 1.. 50 ]  of  string [ 30 ] ;
```

برای خواندن چنین آرایه‌ای بصورت زیر عمل می‌کنیم:

```
For   i : = 1   to   50   do
      Readln ( Name [ i ] ) ;
```

برای نمایش آرایه در خروجی بصورت:

```
For   i: = 1   to   50   do
      Writeln ( Name [ i ] ) ;
```

عمل می کنیم.

برای دسترسی به عنصر ام، اسم i ام آرایه ای از رشته، مثل یک آرایه دو بعدی عمل می کنیم:

```
Name [ i , j ]: = 'B'
```

برا ی مقایسه رشته ها، مثل متغیرهای دیگر عمل می کنند و عملگرهای مقایسه ای را براحتی با رشته‌ها می‌توان بکار برد.

در مقایسه دو رشته از لحاظ بزرگی و کوچکی، کد اسکی آنها در نظر گرفته می‌شود و هر کدام از رشته‌ها، دارای اولین کاراکتر با کد اسکی کوچکتر باشد از دیگری کوچکتر است.

مثال ۹-۱۰ رشته alpha و Beta را در نظر بگیرید.

Alpha	عملگر	Beta	دلیل صحیح بودن شرط
'AAA'	<	'ZZZ'	'A'<'Z'
'ALI'	<	'ali'	'A'<'a'
'AZZ'	<	'BAZ'	'A'<'B'
'BAZ'	>	'BAA'	'Z'>'A'
'BA12'	>	'BA11'	'2'>'1'

با توجه به مثال بالا ملاحظه می کنید که مقایسه بین محتوای متغیرهای رشته ای یک به یک و از سمت چپ صورت می‌گیرد.

مثال ۱۰-۱۰ برنامه‌ای بنویسید که اسم ۱۰ نفر از ورودی دریافت کرده و آنها را برحسب حروف الفبا مرتب نماید.

```
Var
   Name  :  Array  [ 1 .. 10 ]  of  string [ 10 ] ;
   Temp:string [ 10 ] ;
   i , j : Byte;
Begin
        writeln ( 'Enter Ten Names' ) ;
        for  i : = 1   to   10   do
             Readln ( Name [ i ] ) ;
        for  i : = 1   to   9   do
             for  j : = i + 1   to   10   do
                  if  Name [ i ] > Name [ j ] Then
                  Begin
                        Temp : = name [ i ] ;
                        Name [ i ] : = Name [ j ] ;
                        Name [ j ] : = temp ;
                  End ; { sort  Algrithm }
        writeln ( 'The sorted list' ) ;
        for  i : = 1   to   10   do
             writeln ( Name [ i ] ) ;
end . { End of program }
```

برنامه بالا یک لیست ۱۰ عنصری را بر اساس حروف الفبا بصورت صعودی مرتب می کند.

مثال ۱۱-۱۰ برنامه‌ای بنویسید که یک عدد در مبنای ۱۶ را به یک عدد در مبنای ۱۰ تبدیل نماید.

باید توجه کنیم که عدد در مبنای شانزده ترکیبی از ارقام و حروف می‌باشد لذا آن را بصورت رشته از ورودی دریافت می‌کنیم.


```
Var
        Number: string[10];
       Name :  string [ 10 ] ;
        n: longint  ;
        i: Byte ;
Begin
        writeln ( ' Enter  Number ' )  ;
        Readln ( Number )  ;
        i: =1  ;
        while   Number [ i ]<> eoln  Do
                 inc ( i )  ;
        n: = dec ( i )  ;
        p: = 1  ;
        n: = 0  ;
        for   i: = n    downto   1    do
          Begin
             if  Number[ i ] > = ' 0 ' and  Number[ i ] < = ' 9 '  then
                  M: = ord ( Number [ i ] ) – 48
               Else  If ( Number [ i ]  = ' A ' ) or ( Number [ i ] = ' a ' )  then
                  M: = 10
      Else  If ( Number [ i ]  = ' B ' ) or ( Number [ i ] = ' b ' )  then
                  M: = 11
            Else  If ( Number [ i ] = ' C ' ) or ( Number [ i ] = ' c ' ) then
                  M: = 12
            Else  If ( Number [ i ]  = ' D ' ) or ( Number [ i ] = ' d ' )  then
                  M: = 13
             Else  If ( Number [ i ]  = ' E ' ) or ( Number [ i ] = ' e ' )  then
                  M: = 14
            Else  if ( Number [ i ]  = ' F ' ) or ( Number [ i ] = ' f ' )  then
                  M: = 15 ;
             n: = M * P + n  ;
             P:= p * 16 ;
          end ; { End  of  for }
         writeln  ;
         writein ( ' The  Number  is  TEN  base ' )  ;
         writeln ( n )  ;
end. { End  of program}
```

یک عدد در مبنای ۱۶ مجموعه ای از ارقام و حروف (A تا F) می‌باشد. برنامه بالا ابتدا ارقام کاراکتری را به رقم تبدیل کرده، سپس آنرا به مبنای ۱۰ می برد. فاصله کد اسکی بین کاراکتر ' 0 ' تا رقم 0، ۴۸ می باشد. حروف A تا F در رشته خوانده شده بررسی شده، معادل عددی آنها که 10 تا 15 می باشد جایگزین می شوند.

مثال ۱۲-۱۰ برنامه‌ای بنویسید که اسم، فامیلی و شماره دانشجوئی حداکثر ۵۰ دانشجو را از ورودی دریافت نماید.

سپس با خواندن اسمی از ورودی، سایر اطلاعات اسم خوانده شده را در صورت وجود در خروجی نمایش دهد.

```
Var
        Name , Family : array [ 1 .. 50 ] of string [ 30 ] ;
        id : array [ 1 .. 50 ] of longint ;
        Nam : string [ 30 ] ;
Begin
        write ( 'Enter Number : ' ) ;
        Readln ( N ) ;
        for  i := 1  to  N  Do
          begin
              write ( 'Enter Name : ' ) ;
              Readln ( Name ) ;
              write ( 'Enter family : ' ) ;
              Readln ( family ) ;
              write ( 'Enter student Number : ' ) ;
              Readln ( Id ) ;
          end ;
        writeln ;
        write ( 'Enter Name : ' ) ;
        Readln ( Nam ) ;
        i := 0 ;
        flag := False ;
        while ( i <= n ) and flag Do
            begin
                inc ( i ) ;
                if  Name [ i ] = Nam then
                    flag := True ;
            end ;
         If flag=false then
             writeln ( 'Not found' ) ;
         Else
            begin
                writeln ( '  Id          Name          family  ')
                write ( Id , Name : 30 , family : 30 ) ;
            end ;
End . { End of program }
```

حال بعضی از توابع و روالهای کتابخانه ای رشته ها را بررسی می کنیم.

۱۰-۳- توابع و روالهای کتابخانه‌ای برای متغیرهای رشته‌ای

همه عملگرهایی‌که برای متغیرهای دیگر بحث کردیم در مورد متغیرهای رشته‌ای کاربرد ندارند و برخی از عملگرها مفهوم جدیدی در رشته ها پیدا می کنند. بطور مثال عملگر + در مورد رشته ها مفهوم جدیدی ارائه می‌دهد. در این بخش عملگرها، توابع و روال‌های مورد استفاده در رشته‌ها را ارائه می‌دهیم.

۱۰-۳-۸- تابع Concat

هدف: الحاق دو یا چند رشته به یکدیگر

شکل تابع:

```
Function  concat ( S1 , S2 , ... , Sn ): string  ;
```

S1 , S2 , ... , Sn متغیرهایی از نوع رشته هستند و خروجی تابع نیز یک متغیر رشته‌ای است این تابع دویا چند رشته را به هم پیوند داده و رشته حاصل را برمی‌گرداند.

مثال ۱۰-۱۳

```
Var
        Str3 , str1 , str2 :  string
        Begin
              Str1 := 'Pascal' ;
              Str2 := 'Book' ;
              Str3 := Concat ( Str1 , Str2 ) ;
              write ( Str3 ) ;
        End
```

با اجرای برنامه فوق، عبارت Pascal Book نشان داده خواهد شد.

۱۰-۳-۲- تابع Copy

هدف: استخراج یک زیر رشته (substring) از یک رشته

شکل تابع:

```
Function  copy (S: string  ;  Index: Integer  ;  count: Integer): string  ;
```

S یک عبارت یا متغیر رشته ای که می خواهیم از آن زیر رشته ای که نقطه شروع آن Index و طول آن Count می باشد جدا کنیم. لذا زیر رشته حاصل، یک رشته که طول آن به اندازه Count است، می‌باشد.

مثال ۱۰ ۱۴

```
Var
        Str , str1:  string  ;
Begin
        Str: = 'Pascal  Book' ;
        Str1: = Copy ( Str , 7 , 4 )  ; { Str1 = Book }
        write ( Str1 ) ;
End.
```

۱۰-۳-۳- روال Delete

هدف: حذف یک زیر رشته از یک رشته

شکل‌روال: Procedure delete (Var str: string ; Index: integer ; length: integer) ;

Str یک متغر رشته ای، Index یک عبارت یا متغیر صحیح و Length نیز یک عبارت یا متغیر صحیح می‌باشد. روال Delete یک زیر رشته را از یک رشته حذف می‌کند این روال از محل Index بطول Length از رشته Str حذف می‌کند و رشته حاصل بعنوان خروجی روال برگردانده می‌شود.

مثال ۱۰-۱۵

```
Var
      St : string ;
Begin
      St: = 'Pascal.Book' ;
      Delete ( St , 7 , 5 ) ;
      writeln ( St ) ;
End.
```

خروجی حاصل از برنامه فوق Pascal می‌باشد.

۱۰-۳-۴- روال Insert

هدف: درج (وارد کردن) یک رشته در یک رشته دیگر

شکل روال: Procedure Insert (Str1: string ;Var Str2: string ; Index: Byte) ;

Str1 عبارت یا متغیر رشته ای، Str2 متغیر رشته‌ای و Index عبارت عددی از نوع صحیح می‌باشد. روال Insert ، رشته Str1 رادر رشته Str2 از خانه Index درج می‌کند و رشته حاصل خروجی روال خواهد بود.

```
Str1 : = 'Pascal ' ;
Str2 : = ' Programming Book ' ;
 Insert ( Str1 , Str2 , 1 ) ;
```

رشته حاصل Pascal Programming Book خواهد بود.

مثال ۱۰-۱۶

```
Var
      Str1 , str2: string ;
Begin
      Str1: = 'Pascal 7' ;
      Str2: = 'Turbo' ;
      Insert ( Str1 , Str2 , 6 ) ;
End.
```


خروجی حاصل از برنامه بالا Turbo pascal 7 خواهد بود.

۱۰-۳-۵- تابع Length

هدف: محاسبه طول رشته

شکل تابع: Function length (Str: string): Integer ;

Str یک عبارت رشته ای بوده و خروجی تابع یک عدد صحیح می‌باشد. این تابع طول رشته ورودی را محاسبه و بعنوان خروجی بر می‌گرداند.

مثال ۱۰-۱۷

```
Var
      St: string ;
      n: integer ;
Begin
      St: = 'Turbo Pascal 7' ;
      n: = length ( St ) ;   { n = 14 }
      Write ( ' The langth of string is: ' , n ) ;
End.
```

۱۰-۳-۶- تابع Pos

هدف: برای جستجوی یک رشته داخل رشته دیگر

شکل تابع: Function pos (Str1: string ; Str2: string): Byte ;

Str1 یک عبارت یا متغیر رشته ای و Str2 نیز یک عبارت یا متغیر رشته ای می‌باشد و خروجی تابع یک عدد صحیح می باشد. این تابع محل اولین وقوع رشته Str1 در رشته Str2 را بعنوان خروجی بر می‌گرداند.

مثال ۱۰-۱۸

```
Var
      Str1 , Str2: string ;
        i: integer ;
Begin
      Str1: = ' Book ' ;
      Str2: = ' Pascal Book ' ;
      i: = Pos ( Str1 , Str2 ) ;   { i = 8 }
      Write ( ' i = ' , i ) ;
End.
```

نکته: توجه کنید که اگر عمل جستجو با موفقیت انجام نشود (یعنی رشته اول در رشته دوم وجود نداشته باشد) تابع مقدار صفر بر می‌گرداند.

مثال ۱۹-۱۰ برنامه ای بنویسید که یک رشته از ورودی دریافت کرده کاراکتر خالی (space) را با ستاره (*) جایگزین نماید.

```
Var
        St : string ;
       Flag : Boolean ;  i : Byte
Begin
          Write ( ' Enter string : ' ) ;
          Readln (St ) ;
          i : = Pos ( ' ', St ) ;
          While    i > 0     Do
            Begin
                  St [ i ] : = ' * ' ;
                  i : = Pos ( ' ', St ) ;
             end ;
    Writeln ( ' The result string ' ) ;
   Write ( St ) ;
End . { End of program }
```

خروجی حاصل از برنامه فوق بصورت زیر است:

```
Enter string: This Book is written by Yousef and Tanha
The result string
This*Book*is*written*by*Youusef*and*Tanha
```

۱۰-۳-۷- روال Str

هدف: برای تبدیل عدد به یک رشته عددی بکار می رود.

شکل روال:

```
1:  Procedure  Str ( I: integer: format , Str: string ) ;
2:  Procedure  Str ( F: Real: format , Str: string ) ;
```

درشکل 1، I یک عبارت یا متغیرعددی بوده و Str یک متغیر رشته‌ای می‌باشد این روال یک عدد صحیح با فرمت مشخص را به یک رشته عددی تبدیل می کند.

درشکل 2، F یک عبارت یا متغیر اعشاری بوده و Str یک متغیر رشته‌ای می‌باشد.

این روال یک عدداعشاری با فرمت مشخص را به فاصله یک رشته عددی تبدیل می‌کند.


```
Var
Number , Str: String  :
Begin
        Str ( 543: 5 , Str ) ;     { Str = ' 543' }
        Str ( 12.25: 7: 2 ,Str ) ;     { Str = ' 12.25' }
        Str ( 127: 3: Str ) ;     { Str = '127' }
        Str ( 3.1239: 7: 3 , Str ) ;     { Str = ' 3.124' }
End.
```

۱۰-۳-۸- روال Val

هدف: تبدیل یک رشته عددی به یک عدد

شکل روال:

```
1: Procedure  Val ( S: String  ; Var  N: integer  ; Var  Error: integer ) ;
2: Procedure  Val ( S: String  ; Var  N: Real  ; Var  Error: integer ) ;
```

S یک عبارت یا متغیر رشته ای، N یک متغیر از نوع صحیح یا اعشاری و Error نیز یک متغیر از نوع صحیح می باشد. این روال یک رشته عددی را به یک عدد صحیح یا اعشاری تبدیل می کند و آن را توسط متغیر N برمی‌گرداند. اگر عمل تبدیل بطور صحیح انجام شود، مقدار متغیر Error برابر صفر در غیر اینصورت محل وجود اشکال را مشخص می‌کند.

مثال ۲۱-۱۰

```
Var
        St: string  ;
        E , N: integer   ;
        F: Real  ;
Begin
         St = ' 475 ' ;
         Val ( St , N , E ) ;       { N = 475 , E = 0 }
         St = ' 475' ;
         Val ( St , F , E ) ;       { F =  , E = 3 }
    St = '3.1716' ;
         Val ( St , F , E ) ;       { F = 3.1716  , E = 0 }
End.
```

۱۰-۴- ارائه چند مثال در مورد رشته‌ها و کاراکترها

در این بخش مثال هایی را ارائه می‌دهیم که در آنها از توابع و روال‌های کتابخانه‌ای استفاده شده است.

مثال ۲۲-۱۰ برنامه‌ای بنویسید که یک جمله از ورودی دریافت کرده سپس در صورتی که کلمه IS وجود داشته باشد آنها را به are تبدیل نماید و در نهایت رشته حاصل را در خروجی نمایش دهد.

توجه کنید که ممکن است بیش از یک بار کلمه IS تکرار شده باشد در اینصورت همه کلمات IS را با are تبدیل کند.

```
Var
        St : string ;
Begin
      Writeln ( 'Enter sentence' ) ;
      Readln ( St ) ;
      Repeat
              i : = Pos ( 'is' , St ) ;
              if    i > 0     Then
                Begin
                     Delete ( St , i , 2 ) ;
                     Insert ( 'are' , St , i ) ;
                end ;
        Until   i = 0 ;
     Writeln ( 'The result sentence' ) ;
     Write ( St ) ;
End . { End of program }
```

خروجی برنامه بالا:

```
Enter sentence:
This is Pascal
The result sentence
This are Pascal
```

مثال ۲۳-۱۰ برنامه‌ای بنویسید که یک رشته از ورودی دریافت کرده، سپس معکوس رشته را بدست آورده بهمراه خود رشته در دو سطر جداگانه چاپ نماید.

```
Var
        St1 , St2 : string ;
        i , j : integer ;
Begin
     Writeln ( 'Enter sentence' ) ;
     Readln ( St1 ) ;
     j : = 1 ;
     For    i : = length ( St1 )    downto    1    do
          Begin
                St2 [ j ] : = St1 [ i ] ;
                Inc ( j ) ;
          End ;
Writeln ( 'The result' ) ;
     Writeln ( St1 ) ;
     Writeln ( St2 ) ;
End . { End of program }
```


مثال ۲۴-۱۰ برنامه ای بنویسید که اطلاعات حداکثر ۱۰۰ دانشجو که عبارتند از:

Name	family	Ave	Id
اسم	فامیلی	معدل	شماره دانشجوئی

از ورودی دریافت کرده سپس:

۱. اطلاعات را بر اساس فامیلی مرتب نماید.

۲. دانشجو یا دانشجویان با بیشترین معدل را یافته بهمراه سایر اطلاعات آنها در خروجی چاپ کند.

برای برنامه مطرح شده می توان ساختار داده زیر را در نظر گرفت:

۱. آرایه ای از رشته ها برای اسم و فامیلی

۲. آرایه ای از نوع اعشاری برای معدل

۳. آرایه ای از نوع صحیح برای شماره دانشجوئی

```
Var
        Name , Fam: array [ 1.. 100 ] of string ;
        Id: array [ 1.. 100 ] of longint ;
        Ave: array [ 1.. 100 ] of Real ;
        I , j , N: integer ;
        Temp1: String ;
        Temp2: Real ;
        Temp3:  Longint ;
Begin
        Write ( ' Enter  Number: ' ) ;
        Readln ( N ) ;
        For    i: = 1    to    N    do
        Begin
              Write ( ' Enter  Name: ' ) ;
              Readln ( Name ) ;
              Write ( ' Enter  Family: ' ) ;
              Readln ( fam ) ;
              Write ( ' Enter  ave: ' ) ;
              Readln ( ave ) ;
              Write ( ' Enter  Id: ' ) ;
              Readln ( Id ) ;
End ; { End  of  input }
For    i: = 1    to    N - 1    do
   For    J: = i + 1    to    N    do
        If  fam [ i ] > fam [ j ]  Then
        Begin
            Temp1: = fam [ i ] ;
            Fam [ i ]: = fam [ j ] ;
            Fam [ j ]: = temp1  ;
            Temp: = Name [ i ] ;
            Name [ i ]: = Name [ j ] ;
            Name [ j ]: = temp1  ;
            Temp2: = ave [ i ] ;
            Ave [ i ]: = ave [ j ] ;
            Ave [ j ]: = temp2  ;
            Temp3: = Id [ i ] ;
            Id [ i ]: = Id [ j ] ;
              Id [ j ]: = temp3  ;
        End ; { End  of  swap }
        Writeln ( '          The sorted Information          ' ) ;
        For    i: = 1    to    N    do
          Writeln ( Name: 30 , fam: 20 , ave: 8: 2 , Id: 8 ) ;
        Max: = ave [ 1 ] ;
        For    i: = 2    to    N    do
           If  ave [ i ] > Max  Then
               Max: = ave [ i ] ;
        Writeln ( '       The Maximum average          ' ) ;
        For    i: = 1    to    N    do
          If  Max   ave [ i ]  Then
            Writeln (Nam [ i ]: 30 , fam [ i ]: 20 , ave: 8: 2 , Id [ i ]: 8) ;
End. { End  of  program }
```


۱۰-۵- تمرینات

۱- با فرض اینکه S1 , S2 , S3 متغیر های رشته ای هستند خروجی عبارتهای زیر را تعیین کنید.

الف) `S3 = copy ( S1 , 1 , 6 ) ;`

ب) `S3 = concat ( S3 , S2 , S1 ) ;`

ج) `S3 = copy ( S2 , 1 , pos( S1 , S2 ) - 1 ) ;`

د) `Delete ( S2 ,  pos( S1 , S2 ) , length ( S1 ) ) ;`

خروجی قطعه برنامه های زیر را تعیین کنید:

الف)

```
Var
        S1: string ;
        i: integer ;
Begin
        S1: = 'ABCDEF' ;
        For    i: = 1    to    length ( S1 ) ;
            Delete ( S1 , i , 1 ) ;
        Writeln ( S1 ) ;
End.
```

ب)

```
Var
        St: string ;
        i: integer ;
Begin
        St: = ' pascalBook ' ;
        For    i: = 1    to    length ( St )    Do
           Begin
                Delete ( St , i , 1 ) ;
                Dec ( i ) ;
            End ;
        Writeln ( St ) ;
End.
```

ج)

```
Var
        St: string ;
        i: integer ;
Begin
        Readln ( St ) ;
        For    i: = 1    to    length ( St )    Do
            If ( St [ i ] > = ' 0 ' ) and ( St [ i ] <    ' 9 ' ) then
                Delete ( St , i , 1 ) ;
        Writeln ( St ) ;
End.
```

د)

```
Var
        St: string ;
        i: integer ;
Begin
        Readln ( St ) ;
        For    i: = 1    to    length ( St )    Do
           If ( St [ i ] > = ' a ' ) and ( St [ i ] < = ' Z ' ) then
              Begin
                    Delete ( St , i , 1 ) ;
                    Dec ( i ) ;
              End ;
        Writeln ( St ) ;
End.
```

۳- با فرض اینکه S1 = ' pascal Book ' , S2 = ' Book ' هستند خروجی عبارت زیر را تعیین کنید.

الف) `Insert ( S1 , S2 , pos ( S1 , S2 ) ) ;`

ب) `S3 : = copy ( S2 , pos ( S1 , S2 ) , length ( S1 ) ;`

ج) `S3 : = copy ( S2 , pos ( St , S2 ) , length ( S2 ) ;`

د) `Delete ( S1 , pos ( S2 , S1 ) , length ( S2 );`

ح)
```
S1 [ 0 ] := # 6 ;
S1 = S2
Write ( ' ok ' )
Else
Write ( ' NO ' ) ;
```


۱۰-۶- تمرینات برنامه‌نویسی

۱- برنامه‌ای بنویسید که یک پاراگراف را از ورودی دریافت کرده (حداکثر ۱۰ خط) سپس:

الف) تعداد کلمات هر سطر را شمرده و انتهای سطر نمایش دهد.
ب) تعداد حروف صدا دار را شمرده چاپ نماید.
ج) تعداد خطوط برنامه را شمرده در خروجی چاپ کند.
د) تعداد جملات هر خط را محاسبه و در سطر های جداگانه نمایش دهد.

۲- برنامه‌ای بنویسید که یک عدد در مبنای مشخص را از ورودی دریافت کرده سپس آن را به مبنای m که از ورودی خوانده می شود ببرد.

۳- برنامه‌ای بنویسید که یک پاراگراف را از ورودی دریافت کرده کلمات تکراری هر سطر را حذف نموده و پاراگراف حاصل را بهمراه پاراگراف اولیه در خروجی با پیغام مناسب نمایش دهد.
(پاراگراف حداکثر ۱۵ خط می باشد)

۴- برنامه‌ای بنویسید که یک رشته عددی را از ورودی دریافت کرده سپس آن را به یک عدد تبدیل نماید.

۵- برنامه‌ای بنویسید که تعدادی رشته از ورودی دریافت کرده سپس متقارن بودن هر رشته را بصورت زیر بررسی نماید:

رشته های ورود	پیغام
ABBA	yes
XBCBCA	No
ABCDCA	No

۶- برنامه‌ای بنویسید که دو عدد ۲۰ رقمی (بصورت رشته باید خوانده شود) را از ورودی دریافت کرده سپس مجموع و حاصل ضرب این دو عدد را محاسبه و در خروجی با پیغام مناسب چاپ نماید.

۷- برنامه‌ای بنویسید که یک رشته از ورودی دریافت کرده سپس کلمات داخل آن را بصورت عمودی کاراکتر به کاراکتر نمایش دهد:

مثال:

ورودی

This is Book

خروجی

T	i	B
h	s	o
i		o
s		k

۸- برنامه‌ای بنویسید که یک پاراگراف با حداکثر ۵ خط را از ورودی دریافت نماید. سپس مجموع ارقام یا اعداد (در صورت وجود) هر خط را محاسبه نموده در انتهای همان سطر نمایش دهد.

فصل ۱۱

برنامه‌های فرعی

هدفهای کلی

- شناخت اجزاء تشکیل‌دهنده توابع و روالها
- بررسی انواع پارامترها و متغیرها در برنامه‌های فرعی
- شناخت تفاوتهای روالها و توابع
- معرفی مزایای استفاده از زیر برنامه‌ها

هدفهای رفتاری

دانشجو پس از مطالعه این فصل باید بتواند:

- برنامه خود را به چندین زیربرنامه تقسیم‌بندی نماید.
- زیربرنامه‌ها را با توجه به قوانین موجود، در ساختار برنامه جا دهد.
- باتوجه به مزایای استفاده از آنها را در برنامه خود بکار ببرد.

مقدمه

همانطور که در فصل ۹ اشاره کردیم، زبان پاسکال تعدادی روال و تابع استاندارد دارد که در حین نوشتن برنامه می‌توان از آنها استفاده کرد. در این فصل در نظر داریم، که خودمان توابع و روال‌های مورد نیاز برنامه را بنویسیم.

نیاز به نوشتن برنامه‌های فرعی در برنامه زمانی مشاهده می شود، که بخواهیم یک برنامه بزرگ یا نسبتاً بزرگ را پیاده‌سازی کنیم. در اینصورت سعی می‌کنیم، برنامه را به قسمت‌های مجزا و جداگانه از هم تقسیم‌بندی کرده، سپس توسط برنامه‌های فرعی، قطعات جداگانه را پیاده‌سازی کرده و در نهایت آنها را به برنامه اصلی پیوند دهیم. استفاده از برنامه‌های فرعی یکی از اصول برنامه‌نویسی ساخت‌یافته می‌باشد و خوانایی برنامه توسط آنها افزایش می‌یابد.

برنامه‌های فرعی معمولاً قسمتهای مستقلی از برنامه هستند، که به تنهایی عمل خاصی را انجام می‌دهند. با این ویژگی می‌توان برنامه‌هایی نوشت، که دارای قسمتهای جداگانه و مشخص باشند و هر قسمت یک یا چند وظیفه از وظایف کلی برنامه را به انجام می‌رساند. لذا غالباً برنامه را به قسمت‌های مجزا از هم تقسیم‌بندی می‌کنند و هر قسمت توسط یک روال یا تابع پیاده‌سازی می‌شود و نتایج در برنامه اصلی فراخوانی می‌شوند. از مزایای دیگر استفاده از برنامه های فرعی رفع اشکال سریع برنامه، استفاده بهینه از حافظه، تولید قطعات با قابلیت استفاده مجدد و غیره می‌باشد.

۱-۱۱- روال‌ها

روال‌ها نوعی از برنامه‌های فرعی هستند، که به طور مستقل و جداگانه وظیفه یا وظایف خاصی از برنامه اصلی را انجام می‌دهند. روال ها در صورت نیاز اطلاعات خود را از طریق پارامترها دریافت و همچنین در صورت نیاز نتایج را از طریق پارامترها به برنامه اصلی باز می‌گردانند. پارامترها در حقیقت خطوط ارتباطی بین برنامه اصلی و برنامه‌های فرعی هستند. پارامترها باعث می شوند، که توابع و روال ها روانتر عمل کنند، زیرا آنها به برنامه های فرعی این قابلیت را می دهند که با هر فراخوانی، داده‌های مختلفی را مورد پردازش قرار دهند.

شکل کلی روال‌ها بصورت زیر می باشد.


```
Procedure   Name  ( parameters   list ) ;
   ↓          ↓                  ↓
کلمه ذخیره شده   اسم روال      لیست پارامترها

Var
      { List of locad variable }
Begin
        .
        .
          { Procedure Body }
        .
        .
end ; { End of procedure }
```

بعد از معرفی روال، متغیرهای لازم (متغیرهای محلی را بحث خواهیم کرد) در قسمت تعاریف معرفی می‌شوند. سپس بدنه اصلی روال در یک بلاک نوشته می‌شود.

در حالت کلی روال ها در برنامه اصلی بصورت زیر ظاهر می‌شوند:

```
Program   اسم برنامه اصلی   ;
تعاریف برنامه اصلی   ;
Procedure  .....        ;

Begin   { mian program }
.
.
.
   فراخوانی روال ها
.
.
.
End. { End of program }
```

همانطور که مشاهده می‌کنید بعد از تعاریف برنامه اصلی روال‌ها معرفی می‌شوند و در برنامه اصلی روال‌ها فراخوانی می‌شوند. معمولاً روالها به ترتیبی که نوشته می‌شوند، در برنامه اصلی فراخوانی می‌شوند.

در تعریف یک روال برنامه نویس باید توجه ویژه به نوع پارامترها و ترتیب آنها داشته باشد، تا در موقع فراخوانی مشکلی پیش نیاید.

همانطور که در بالا اشاره شد، از پارامترها برای عبور دادن یک یا چند مقدار به روال ها استفاده می شود و توسط آنها مقادیری که در بدنه اصلی روال مورد پردازش قرار گرفته اند، از روال خارج می گردند. پارامترها با توجه به محل وقوع آنها به دو دسته تقسیم می‌شوند:

۱. پارامترهای صوری (Formal parameters)
۲. پارامترهای واقعی (Actual parameters)

از پارامترهای صوری هنگام اعلان روال و از پارامترهای واقعی هنگام فراخوانی روال ها استفاده می‌شود. پارامترهای صوری در حالت کلی ۲ نوعند، که با توجه به نوع روال و تصمیم برنامه نویس مورد استفاده قرار می‌گیرند.

۱-۱-۱۱- پارامترهای مقداری (Value parameters)

پارامترهای مقداری، پارامترهایی هستند که مقدار متغیرهای فرستاده شده از برنامه اصلی را دریافت می کنند و وظیفه آنها فقط عبور دادن مقدار به روال می‌باشد. لذا تغییرات پارامترهای مقداری در روال به برنامه اصلی انتقال نمی‌یابد. پارامترهای مقداری بصورت زیر تعریف می‌شوند:

```
Procedure   Name ( var1 : type ; var2 : type , . . . ) ;
```

کلمه ذخیره شده — اسم روال — لیست پارامترهای از نوع مقداری

متغیرهای Var1 , Var2 , ... پارامترهای مقداری هستند. توجه کنید، که این پارامترها صوری هستند و نوع و ترتیب آنها باید با نوع و ترتیب پارامترهای واقعی در تناظر یک به یک باشند.

مثال ۱-۱۱ به برنامه زیر توجه کنید:


```
Program   main ;
Var
        x1 , x2 , y1 , y2 : integer ;
Procedure     test ( A1 , A2 , B1 , B2: integer ) ;  { formal parameters }
    ......
Begin { main }
    ....
        test ( x1 , x2 , y1 , y2 ) ; { Actual parameters }
    ......
End. { End of program }
```

همانطور که ملاحظه می کنید، روالی بنام test با ۴ پارامتر مقداری در برنامه استفاده شده است. متغیرهای x1 , x2 , y1 , y2 از برنامه اصلی به روال test ارسال شده و بترتیب مقادیر این متغیرها در متغیرهای با همان نوع در A1 , A2 , B1 , B2 قرار می گیرند.

نکته: پارامترهایی که در روال بکار برده می شود، هیچ ارتباطی به پارامترهای ارسالی از برنامه اصلی ندارند و فقط مقادیر این متغیرها از برنامه اصلی ارسال می شود. لذا اسامی پارامترهای صوری ممکن است هم اسم با پارامترهای واقعی برنامه اصلی انتخاب شوند، این به معنای این نیست که این پارامترها (پارامترهای صوری) همان پارامترهای واقعی هستند.

در مثال بالا این تناظر برقرار است:

پارامترهای صوری	متناظر است با	پارامترهای واقعی
A1		X1
A2		X2
B1		Y1
B2		Y2

مثال ۲-۱۱ به مثال زیر توجه کنید:

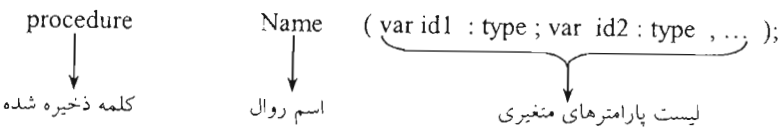
```
Program   main ;
Var
        X1 , X2 : integer ;
        Y1 , Y2 : Real ;
Procedure    test ( x1 , x2 : integer ; f1 , f2: Real ) ; { formal parameters }
    ......
Begin { Main }
    ......
     test ( x1 , x2 , f1 , f2 ) ; { Actual parameters }
    ......
End { End of program }
```

همانطور که در برنامه بالا ملاحظه می کنید، ممکن است پارامترهای واقعی و صوری یکی باشد ولی همانطور که قبلاً اشاره کردیم نباید در روال و برنامه اصلی در مورد مقادیر آنها مرتکب خطا شویم.

۲-۱-۱۱- پارامترهای متغیری (Variable parameters)

یک پارامتر متغیری، تغییرات یک پارامتر واقعی را بعنوان خروجی به برنامه اصلی برمی‌گرداند. تفاوت این گونه پارامترها با پارامترهای مقداری در ارسال تغییرات ایجاد شده در پارامترها می‌باشد. وقتی نیاز است، که نتایج روال به برنامه اصلی برگردد، از پارامترهای متغیری استفاده می‌کنند.

اعلان پارامترهای متغیری به صورت زیر می‌باشد:



متغیرهای id1 , id2 و ... پارامترهای متغیری می‌باشند که نتایج روال، توسط آنها به برنامه اصلی بازگردانده می‌شود.

مثال ۳-۱۱ روال compute_sum در زیر مجموع اعداد از ۱ تا N را محاسبه می‌کند و نتیجه این مجموع را به برنامه اصلی برمی‌گرداند:

```
procedure  compute_sum (N: integer ;var  sum : integer);
var
        I :integer; {local variable}
Begin
            For I:=1 to N do
                Sum:= I + sum;{sum is variable parameter}
End; {End of  procedure}
```

برای مشاهده نحوه عملکرد این روال، فرض کنید در برنامه اصلی متغیرهای M و TOTAL از نوع صحیح تعریف شده‌اند.

دستور فراخوانی:

```
Compute_sum(M,Total);
```


پارامترهای صوری	متناظر است با	پارامتر واقعی
N		M
Sum		Total

در روال بالا N یک پارامتر مقداری و sum یک پارامتر متغیری می‌باشد و نتیجه مجموع اعداد از یک تا N توسط متغیر sum برگردانده می‌شود.

مثال ۴-۱۱ روالی بنام ComputeSumAve بنویسید که مجموع و میانگین دو عدد را محاسبه و نتیجه را در برنامه اصلی چاپ نماید.

```
Program    Example4;
Var
        Num1, Num2: integer;
        Total,average: Real;
Procedure  ComputeSumAve(Num1 , Num2: integer ; Var  sum , ave : Real);
Begin
        Sum:=Num1+Num2;
        Ave:=sum/2;
 End; {End of procedure}
Begin {Main}
       WriteLn(' Enter Two Numbers');
       ReadLn(Num1,Num2);
       ComputeSumAve (Num1, Num2, total, average);
      Write Ln ( ' The Sum is= ', sum:8:2 , ' The average is= ' , average:8:2);
End. {End of program}
```

پارامترهای واقعی و صوری عبارتند از:

پارامترهای صوری	متناظر است با	پارامتر واقعی
Num1		Num1
Num2		Num2
sum		Total
ave		Average

Num1 , Num2 پارامترهای مقداری و sum , ave پارامترهای متغیری می‌باشند، که توسط آنها نتایج به برنامه اصلی برگردانده می‌شود.

۳-۱-۱۱- متغیرهای محلی و سراسری (Local and Global Variable)

در برنامه‌های فرعی دو نوع متغیر علاوه بر پارامترها مورد استفاده قرار می گیرند. این متغیرها متغیرهای محلی یا متغیرهای سراسری هستند.متغیرهای محلی، متغیرهایی هستند که در بلاک مربوط به خود قابل استفاده هستند. ولی متغیرهای سراسری در تمام

برنامه‌های فرعی قابل دسترس می‌باشند.

متغیرهای محلی در داخل برنامه‌های فرعی در قسمت تعاریف معرفی می‌شوند و در خارج از روال قابل دسترسی نیستند.

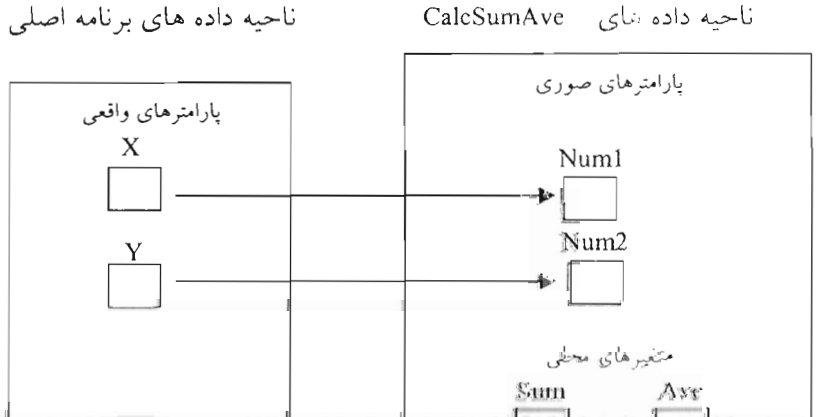
نکته: معمولاً سعی می‌کنند در صورت امکان از متغیرهای محلی به جای متغیرهای سراسری استفاده کنند.

هر بار که روالی فراخوانی می‌شود، ناحیه‌ای از حافظه برای ذخیره داده‌های روال اختصاص می‌یابد. ناحیه داده‌های روال شامل سلولهایی از حافظه است که برای ذخیره پارامترهای صوری و متغیرهای محلی یا ثابتهایی که در روال تعریف می‌شوند، بکار می رود. هرگاه که روالی پایان می‌یابد، ناحیه داده‌های مربوط به آن پاک می‌شود و وقتی روال مجدداً فراخوانی می‌شود، این ناحیه دوباره ایجاد می‌شود.

مثال ۵-۱۱ روالی برای محاسبه مجموع و میانگین دو عدد بنویسید.

```
Procedure    CalcSumAve ( Num1 , Num2: Real );
Var
            Sum , Ave: Real ;
Begin
             Sum:= Num1 + Num2 ;
             Ave:= Sum / 2 ;
             WriteLn(' The Sum is =' , sum:8:2 ) ;
             WriteLn(' The average is =',ave:8:2);
End;{End of procedure}
```

در مثال بالا Ave ، sum متغیرهای محلی می‌باشند. برای فراخوانی روال CalcSumAve(x,y); که درآن x,y دو متغیر ازنوع Real می‌باشند. به شکل زیرتوجه‌کنید:



به وضوح مشاهده می‌کنید که با فراخوانی هر روال ناحیه‌أی از حافظه برای متغیرها و پارامترهای روال تخصیص داده می‌شود. پس از اجرای روال حافظه اختصاص یافته آزاد می‌شود.

متغیرهایی که در برنامه اصلی (در قسمت تعاریف) معرفی می‌شوند به متغیرهای سراسری معروفند. خاصیت متغیرهای سراسری اینست، که در تمام برنامه‌های فرعی به محتویات اینگونه متغیرها دسترسی وجود دارد.

استفاده از متغیرهای سراسری ممکن است حافظه زیادی را هدر دهد، زیرا متغیرهای تعریف شده تا اجرای کامل برنامه از حافظه استفاده می‌کنند. به همین خاطر غالباً سعی می‌کنند، تا جائیکه امکان داشته باشد از متغیرهای سراسری استفاده نکنند.

مثال ۶-۱۱ خروجی برنامه زیر را تعیین کنید.

```
Program     Example6;
Var
        A , b , c: integer ;      {Global Variables}
Procedure     test(var b: integer ; a: integer );
var
        D: integer ;     {local variables}
Begin
        D:= 12 ;
        a:= b + d ;  b:= a + c ; c:= c+ 2 ;
End;
Begin {Main}
        A:=1; b:=2; c:=3;
        Test(a,b);
        WriteLn( ' a = ' , a ,' b = ' , b , ' c = ' , c ) ;
End.{End of program}
```

در مثال بالا C یک متغیر سراسری و d یک متغیر محلی است.

خروجی برنامه بالا بصورت زیر است:

```
a =16 b=2 c=5
```

متغیر محلی d بعد از اجرای روال test ، اگر در برنامه اصلی بکار برده شود از نظر کامپایلر ناشناخته است و اشکال کامپایلری حاصل می‌شود. لذا همانطور که قبلاً اشاره شد، متغیرهای محلی در بلاک مربوط به خود دارای ارزش و اعتبار هستند و خارج از بلاک مربوطه ناشناخته هستند.

مثال ۷-۱۱ خروجی برنامه زیر را تعیین کنید:

```
Program       Example7;
var
        a , b ,c: integer ;
procedure    Change( var   x , y: integer ) ;
var
         c:  integer ;   {local variable}
Begin
         C:= 0 ;
         C:= x ;
         X:= y ;
         Y:= c ;
End; {End  of  procedure}
Begin { Main }
      a: = 10 ;
      b:= 17 ;
      c:=  0 ;
      Change(a,b);
      WriteLn (' a= ' , a , ' b= ' , b ,  ' c =  ' , c ) ;
End. {  End  of  program }
```

در برنامه بالا متغیر c هم سراسری است و هم محلی. در چنین مواقعی درروال، متغیر مربوطه محلی محسوب می‌شود. یعنی ربطی به متغیر سراسری ندارد و در برنامه اصلی متغیر سراسری محسوب می‌شود یعنی هیچ ارتباطی به متغیر محلی ندارد.

خروجی برنامه فوق بصورت زیر خواهد بود:

```
a=17  b= 10  c= 0
```

در نوشتن برنامه با برنامه‌های فرعی، نخست برنامه‌نویس باید روال‌ها، توابع مورد نظر خود را تشخیص دهد و با استفاده از ابزارهایی که برنامه‌های فرعی در اختیار برنامه‌نویس قرار می‌دهند، آنها را پیاده‌سازی کند. استفاده صحیح از پارامترها و متغیرها هنر یک برنامه‌نویس است، که بتواند از آنها با توجه به برنامه، به نحو کاملاً صحیح استفاده نماید. استفاده ناصحیح از ابزارهای برنامه‌های فرعی می‌تواند، نتایج نادرستی را بعنوان خروجی تولید نماید، لذا در استفاده از برنامه‌های فرعی باید با دقت زیادی عمل کنیم.

روال‌ها معمولاً به سه شکل ظاهر می‌شوند.

۴-۱-۱۱- بکارگیری روال‌های بدون پارامتر

گاهی لازم است تا برنامه فرعی کاملاً مستقل (بی‌نیاز از مقادیر برنامه اصلی) در بخش‌های مختلف یک برنامه اجرا شود. در این صورت نیاز به استفاده از پارامتر بی‌مفهوم می‌باشد و از روالهای بدون پارامتر استفاده می‌کنند.

غالباً زمانیکه بخواهیم پیغام‌های خاصی را در قسمتهای مختلف برنامه نمایش دهیم، این پیغام‌ها را در یک روال قرار داده و در صورت نیاز، روال مربوطه را فراخوانی می‌کنیم.

مثال ۸-۱۱ روال Head بدون دریافت پارامتری فراخوانی می‌شود.

```
Program      Example8   ;
Procedure    Head  ;
 Begin
          WriteLn('Name    Family    Age     No');
          WriteLn('.....');
  End;
Begin{Main}
          WriteLn( ' open university ' ) ;
          Head;
End.
```

در برنامه بالا روال Head فقط برای نمایش رشته‌های ثابت بکار می‌رود. همانطور که ملاحظه می‌کنید نیاز به پارامتر ندارد.

۵-۱-۱۱- بکارگیری روال همراه پارامترهای با خاصیت ورودی

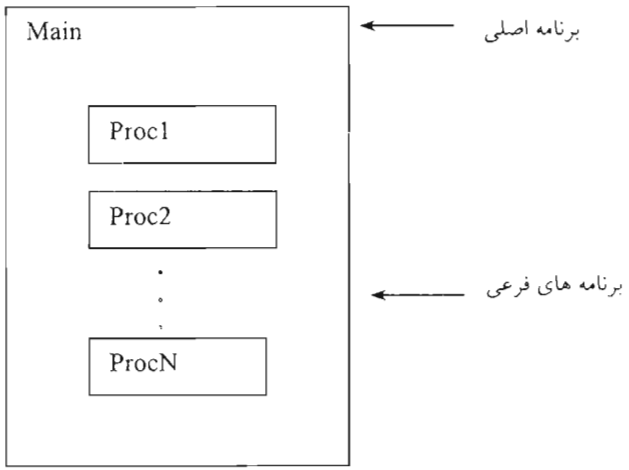
همانطور که قبلاً اشاره کردیم هدف از بکارگیری پارامترها، انتقال مقادیر از برنامه اصلی به روال‌ها می‌باشد. اگر این انتقال یک طرفه باشد یعنی فقط از برنامه اصلی به روال باشد، این نوع پارامترها فقط خاصیت ورودی خواهند داشت. (قبلا در این مورد توضیح داده شده است) در این نوع روال‌ها از پارامترهای مقداری استفاده می‌کنند.

مثال ۹-۱۱ روالی بنویسید که توسط آن مربعات اعداد۱ تا N را درخروجی چاپ نماید.

۷-۱-۱۱- ارتباط روال‌ها با یکدیگر

در پاسکال زیر برنامه‌ها نه تنها از طریق برنامه اصلی بلکه از داخل یکدیگر نیز فراخوانی می‌شوند. فراخوانی روال‌ها از داخل یکدیگر تابع قوانین کلی زیر است:

قانون اول: از هر برنامه (اصلی یا فرعی) به برنامه فرعی در صورتی می‌توان، دسترسی داشت، که در بخش تعاریف آن برنامه (اصلی یا فرعی) قرار داشته باشد. بطور مثال در شکل زیر برنامه اصلیM می‌تواند، به کلیه برنامه‌های فرعی Proc1 و Proc2 و... که در بخش تعاریف برنامه اصلی قرار دارند، مراجعه کند:



مثال ۱۱-۱۱ به برنامه زیر توجه کنید:

```
program Main;
var
        ……..
procedure proc1;
var
        ……..
Begin
        ……..
End;
Procedure proc2;
Var
        ……..
Begin
        ……
end;
Begin {Main}
        Proc1;
        Proc2;
End.
```



```
Program    Example9 ;
Var
        N , I:  integer ;
Procedure   sq (  M: integer ) ;
   Begin
            WriteLn (' sqart is = ' , M * M   ) ;
   End;
Begin {Main}
         Write(' Enter Number = ') ;
         RealLn ( N ) ;
         For     I:= 1    to   N    do
                 Sq (i) ;
End. {End of program}
```

در اینجا M یک پارامتر مقداری می‌باشد و فقط با خاصیت ورود داده‌ها از برنامه اصلی به روال بکار گرفته شده است. این پارامترها نمی‌توانند نتایج را از روال به برنامه اصلی ارجاع دهند.

۶-۱-۱۱- بکارگیری روال همراه پارامترهای با خاصیت ورودی و خروجی

در این نوع روال‌ها پارامترها دو خاصیت مهم دارند یکی انتقال داده‌ها از برنامه اصلی به روال و دیگری انتقال یا ارجاع نتایج از روال به برنامه‌اصلی می‌باشد. در این نوع روال‌ها از پارامترهای متغیری استفاده می‌کنند.

مثال ۱۰-۱۱ روالی بنام change بنویسید که توسط آن دوعدد بعنوان پارامتر دریافت کرده، مقادیر این دو متغیر را جابجا نموده و نتیجه را در برنامه اصلی چاپ نماید.

```
Program   Example10;
Var
        A , b: integer ;
Procedure  change (var  x , y: integer) ;
var
        temp:integer;
Begin
        Temp:=x;
         X:=y;
         Y:=temp;
End;
Begin  { Main }
            RealLn( a , b ) ;
            Change( a , b );
            WriteLn (' a= ' , a , ' b= ' , b ) ;
End. {End of program}
```

در برنامه بالا پارامترهای x,y خاصیت ورودی و خروجی دارند.

روال‌های proc2 و proc1 داخل برنامه اصلی Main تعریف شده‌اند. لذا براحتی می‌توان در برنامه اصلی به آنها دسترسی پیدا کرد. روال‌ها خود نیز داخل هم می‌توانند قرار بگیرند. دقیقاً مثل روال‌های معمولی با این تفاوت که در داخل روال باید تعریف شوند. به اینگونه روالها، اصطلاحاً روال‌های متداخل یا تودرتو می‌گویند.

طریقه تعریف روال‌های تودرتو در حالت کلی بصورت زیر است:

```
procedure proc1;
var
   .............
procedure proc2
var
   .............
  Begin {proc2}
   .............
  End;
Begin { proc1 }
   .............
(فراخوانی روال)    proc2;
   .............
end; { End of proc1 }
```

بهمین ترتیب می‌توان روالهای تودرتوی بیشتری نیز تعریف کرد. اما باید قوانین فراخوانی را درست بکار ببریم. حال قانون یک را در مورد مثال زیر بررسی می‌کنیم:

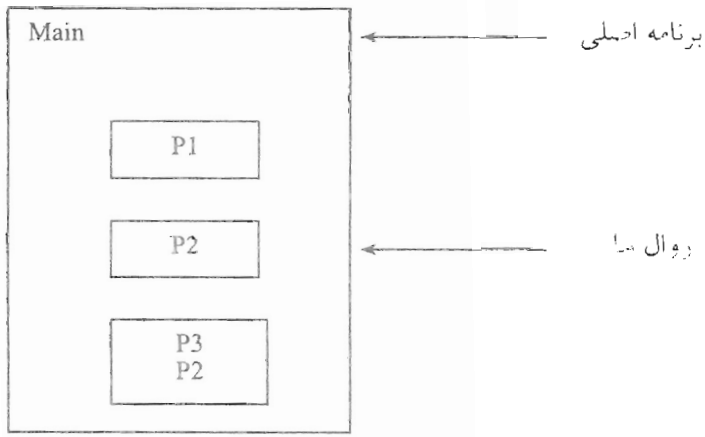
مثال ۱۲-۱۱

```
program test;
var
   .............
procedure proc1;
var
   .............
procedure proc2;
var
   .............
Begin {proc2}
      WriteLn('call from proc one');
End;
Begin {proc1}
      WriteLn('call from Main program');
Proc2;
End;
Begin {Main}
   Proc2;          این فراخوانی امکانپذیر نیست
Proc1;
End. {End of program}
```


در برنامه بالا فراخوانی روال proc2 در برنامه اصلی امکانپذیر نیست، چون در بخش تعاریف برنامه اصلی ظاهر نشده است. روال proc2 در بخش تعاریف روال proc1 ظاهر شده است، لذا در روال proc1 می‌توان آن را فراخوانی کرد. بنابراین قانون یک در این برنامه رعایت نشده است.

قانون دوم: اگر روال‌ها در بخش تعاریف یک برنامه به موازات یکدیگر تعریف شوند و نه در داخل هم، در روال‌های بعدی امکان رجوع به روال‌های قبلی وجود خواهد داشت. بعبارت دیگر به روالی می‌توان دسترسی پیدا کرد که قبلاً تعریف شده باشد.

بطور مثال اگر p1,p2,p3 سه روال باشند دراینصورت می‌توان بصورت زیرعمل کرد:



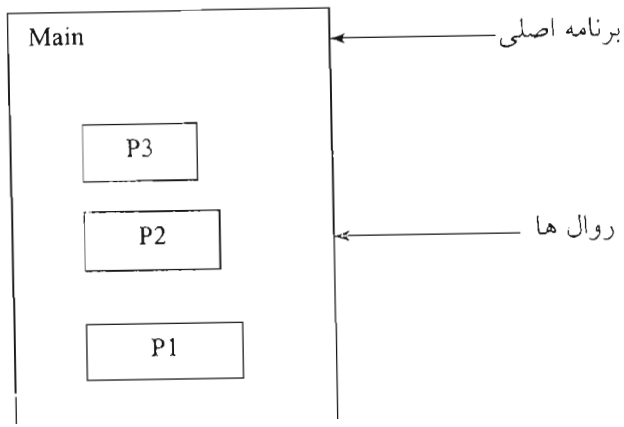
در شکل بالا چون p2 قبلاً معرفی شده است می‌توان آنرا داخل p3 فراخوانی کرد.

مثال ۱۳-۱۱

```
Program      Main ;
var
      ..........
Procedure      proc1 ;
Var
      .........
Begin
      ........
 end; {End of proc1}
Procedure      Proc2 ;
var
      ..........
Begin
      ..........
      proc1; {call proc1}
      .........
end; {End of proc2}
Begin {Main}
      Proc1;
      Proc2;
End. {End of program}
```

همانطور که ملاحظه می‌کنید، چون روال proc1 قبل از روال proc 2 در بخش تعاریف معرفی شده است، لذا می‌توان آن را داخل روال proc2 فراخوانی نمود.

قانون سوم: اگر روال‌ها در بخش تعاریف برنامه‌ای بصورت متداخل باشند، در اینصورت از روال درونی می‌توان به روال‌های بیرونی که قبلاً تعریف شده، مراجعه کرد و همچنین از روال درونی به روال‌های بیرونی که به موازات یکدیگر قرار گرفته‌اند، نیز می‌توان دسترسی پیدا کرد. بطور مثال در شکل زیر روال p1 می‌تواند به روال‌های p2,p3 دسترسی پیدا کند. وهمچنین روال درونی p4 می‌تواند به‌روال‌های p3 , p2 مراجعه کند.



مثال ۱۴-۱۱

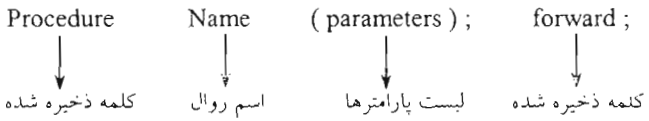
```
Program      Main ;
var
      k: integer ;
Procedure  Add  ( j: integer ) ;
var
      i: integer ;
Procedure   Print ( i: integer ) ;
Begin
            WriteLn;
            WriteLn (i , j);
End;
Begin {Add}
               I:= j + 10 ;
               Print ( I ) ;
 End;
Begin  {  Main  program  }
       WriteLn;
       Write(' please Enter: ') ;
       ReadLn( k ) ;
       Add ( k ) ;
End.{End of program}
```

در روال بالا k یک متغیر سراسری می‌باشد و روال تو در تو بنام Add , print درقسمت تعاریف معرفی شده‌اند. روال Print داخل روال Add تعریف شده است. توجه کنید که در برنامه اصلی نمی توان مستقیماً به روال Print دسترسی پیدا کرد.

۸-۱-۱۱- اعلان روال‌ها به روش forward

همانطور که قبلاً اشاره کردیم از روالی می‌توان در روال دیگر استفاده کرد، که قبلاً تعریف شده باشد. در توربوپاسکال نقیصه فوق به کمک اعلان forward قابل حل است. بدین صورت که اگر روالی به هنگام تعریف با اعلان forward همراه باشد، بدون رعایت از پیش تعریف شدن می‌تواند، در روالهای دیگر ظاهر شود.

شکل اعلان به روش forward بصورت زیر می‌باشد:



در تعریف روال به روش forward قبل از تعریف روال‌های دیگر، تعریف روال مربوطه را همراه با کلمه forward ارائه می‌دهیم. در اینصورت در تمام روال‌های بعدی به روال با اعلان forward دسترسی خواهیم داشت.

مثال ۱۵-۱۱ مثال زیر نخست چند روال را بدون استفاده از اعلان forward ارائه می‌دهد، سپس با اعلان forward اشکالات را برطرف می‌کند.

```
Program   Main ;
Var
      ..........
Procedure    proc1;
Var
        ........
Begin
        ..........
end;
Procedure   proc2;
var
        ..........
Begin
        Proc3  ; { call  proc3}
End;
Procedure   proc3 ;
Var
        ..........
Begin
        ........
end;
Begin {Main program}
        Proc1;
        Proc2;
        Proc3;
End. {End of program}
```

همانطور که در بالا ملاحظه می‌کنید روال proc2 روال proc3 را فراخوانی کرده است. که طبق قوانین مطرح شده در روال‌ها این کار مجاز نیست و نمی‌توان چنین کاری را در حالت معمولی انجام داد.

اعلان روال با forward می‌تواند این مشکل را حل کند.


```
Program   forward_Main ;
Var
        .........
Procedure     proc3  ;   forward ;
Procedure     proc1;
var
        ........
Begin
        .........
end;
Procedure     proc2;
var
        .........
Begin
          .........
            proc3;
             ......
end;
Procedure      proc3;
var
        ........
Begin
       .......
End;
Begin {Main program}
          .........
         proc1;
         proc2;
         proc3;
         .........
End. {End of program}
```

همانطور که ملاحظه می‌کنید با اعلان روال proc3 بصورت forward، بقیه روالها می‌توانند به آن دسترسی پیدا کنند.

مثال ۱۶-۱۱ به روال‌های زیر که با اعلان forward ظاهر شده‌اند دقت کنید.

```
Program       Main  ;
Procedure     Second ;  forward ;
Procedure     First;
Begin
          WriteLn(' This is the first procedure ');
          WriteLn(' Press any key to call The Second   procedure ') ;
          Writen ;
           ReadLn ;
           Second ;    {call second procedure}
           ReadLn ;
 End;{End of first procedure}
{*********************************************************}
Procedure      Second ;
Begin
         WriteLn(' Ok , now second procedure  Begin called ') ;
  End;{End of second procedure}
{*********************************************************}
Begin {Main program}
         First;
End. {End of Main program}
```

همانطور که ملاحظه می‌کنید با اعلان روال second به روش forward توانستیم آن را در روال first فراخوانی کنیم.

۱۱-۲- توابع (Functions)

نوع دیگری از برنامه‌های فرعی، توابع می‌باشند. توابع مانند روال‌ها، پیمانه‌های مستقلی هستند. با این تفاوت که روال‌ها می‌توانند، تعدادی خروجی داشته باشند، در حالی که توابع فقط یک خروجی دارند. در روال‌ها معمولاً خروجی‌ها توسط پارامتر به برنامه اصلی ارجاع داده می‌شود. ولی در توابع اینکار به نحو دیگر انجام می‌گیرد.

شکل کلی اعلان یک تابع بصورت زیر می‌باشد:

```
Function    Name ( List Of Parameters )  :  Function Type ;
```

کلمه ذخیره شده — اسم تابع — لیست پارامترها — نوع تابع

تابع فقط می‌تواند، یک خروجی داشته باشد. نوع خروجی تابع، همان نوع تابع محسوب می‌شود. لذا با توجه به نوع خروجی تابع، نوع تابع تشخیص داده می‌شود. ذکر این نکته خالی از لطف نیست که، نوع پارامترهای صوری توابع معمولاً مقداری هستند چرا که توابع نوعی برنامه‌های فرعی هستند که فقط یک خروجی برمی‌گردانند. لذا استفاده از پارامترهای از نوع متغیری پسندیده نمی‌باشد. (اشکال کامپایلری در توربوپاسکال ندارد ولی در پاسکال استاندارد این کار اشکال کامپایلری دارد.)

مقدار خروجی توابع توسط اسم تابع برگردانده می‌شود. توابع مثل روالها بعد از قسمت تعاریف برنامه اصلی ظاهر می‌شود.

شکل کلی توابع بصورت زیر می‌باشد:

```
Function   Name ( Parameters ) : Type ;
Var
     ..........
Begin
     ..................
     ..................
     ..................
     Name := Resualt Of Function ;
End ;
```

توجه به نکات زیر در بکارگیری توابع ضروری بنظر می‌رسد:

- تمام پارامترهای تابع باید از نوع مقداری باشند.
- نوع داده نتیجه تابع در انتهای عنوان تابع و بعد از لیست پارامترهای صوری قرار می‌گیرد.
- در بدنه تابع، خروجی تابع با نسبت دادن مقدار به نام تابع مشخص می‌شود.

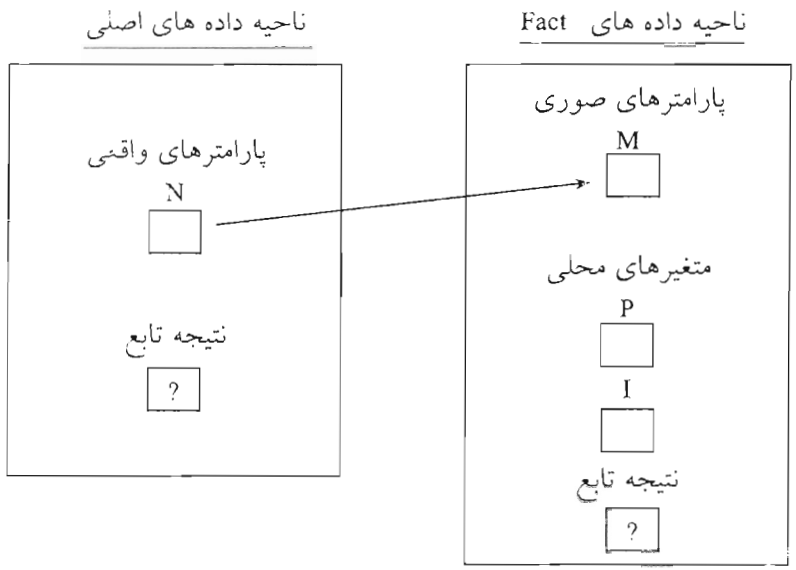
در ضمن آخرین داده‌ای که به نام تابع نسبت داده شود به عنوان خروجی تابع برگردانده می‌شود.

مثال ۱۱-۱۷ برنامه‌ای بنویسید که یک عدد از ورودی دریافت کرده سپس توسط تابعی بنام fact، فاکتوریل عدد را محاسبه در برنامه اصلی چاپ نماید.

```
Program      Example17 ;
Var
     N: Integer ;
Function      Fact ( M : Integer ) :  longint ;
Var
     P , I : Integer ;
Begin
      P:= 1 ;
      For    i:= 2   To    M   do
             P:= P * I ;
       Fact:= P ;
End ;
Begin { Main }
     Write ( ' Enter Number = ' ) ;
     Readln ( N ) ;
     Writeln ( ' Factorial  is = ',  Fact ( N ) ) ;  { Call    Function }
End.
```

در برنامه بالا متغیر سراسری N به تابع fact ارسال می‌شود، سپس فاکتوریل عدد N محاسبه توسط تابع به برنامه اصلی برگردانده می‌شود. نتیجه تابع یک متغیر صحیح longint می‌باشد.

ناحیه داده‌ها بعد از فراخوانی بشکل زیر می‌باشد:



توابع در حالت کلی به دو صورت زیر بکار برده می‌شوند:

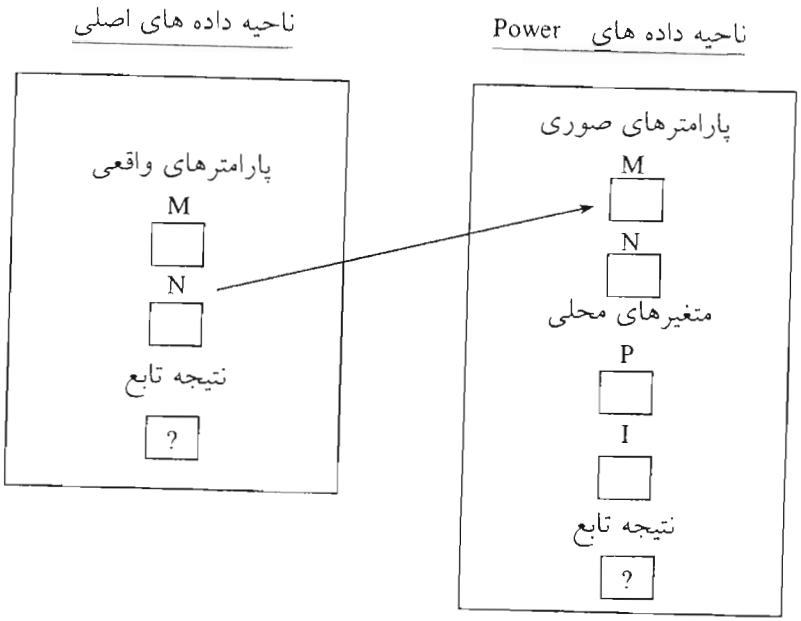
- توابع بدون پارامتر
- توابع با پارامترهای از نوع مقداری

نمونه‌های توابع در مثال‌هایی که بعداً مطرح خواهد شد ملاحظه خواهید کرد.

مثال ۱۸-۱۱ برنامه‌ای بنویسید که دو عدد صحیح M , N را از ورودی دریافت کرده، سپس توسط تابعی بنام power، M را به توان N برساند.

```
Program   Example18 ;
Var
      M , N : integer ;
Function     Power ( M , N : integer ) : Longint ;
Var
      P: Longint ; I: integer ; { local variable }
Begin
      P:= M ;
      For  I:= 2  To  N  do
            P:= P * M ;
      Power:= P ;
End ;
{*************************************************}
Begin
     Writeln ( 'Enter Two Numbers' ) ;
      Readln ( M , N ) ;
     Writeln ;
     Writeln ( M , ' To The Power ' , N , ' Equal = ' , Power( M , N ) ) ;
End. { End Of Program }
```


ناحیه داده‌ها بعد از فراخوانی تابع power بصورت زیر می‌باشد:



۳-۱۱- توابع بازگشتی (Recursion Functions)

در پاسکال یک تابع یا روال می‌تواند، خودش را فراخوانی نماید. پیمانه‌ای که خودش را فراخوانی می‌کند یک پیمانه بازگشتی نام دارد. این نوع توابع در بدنه خود، خودشان را فراخوانی می‌کنند. این فراخوانی تا برقراری یک شرط خاص که غالباً به یک عدد ثابت ختم می‌شود، ادامه پیدا می‌کند. سپس مقدار تابع از پایین به بالا محاسبه می‌شود و در نهایت نتیجه تابع حاصل می‌شود.

نکته: نوشتن همین توابع به صورت بازگشتی ممکن است سرعت برنامه را بکاهد. بنابراین فقط در برخی مواقع که ناگزیر هستیم، به نوشتن توابع بصورت بازگشتی اقدام می‌کنیم.

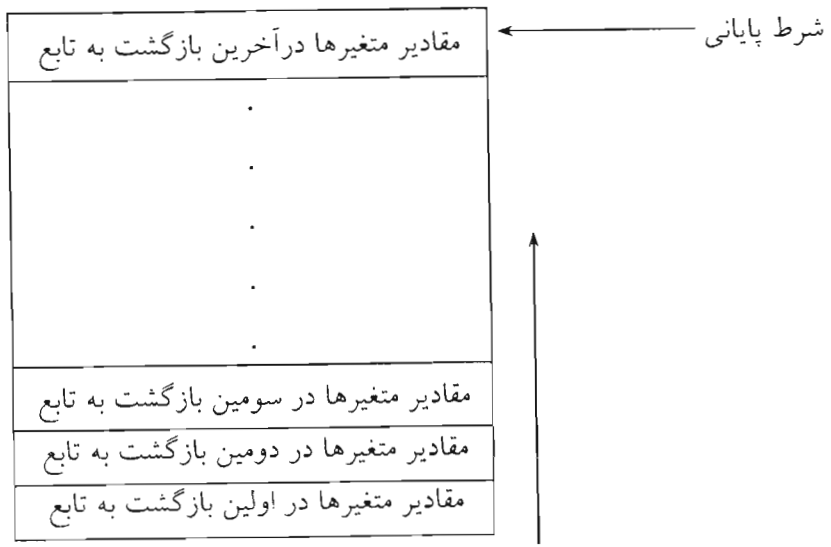
در کلیه برنامه‌های بازگشتی وجود یک شرط پایانی ضروری است. اگر برنامه‌نویس نتواند، شرط پایانی را در مراجعه‌های مکرر تابع کنترل نماید. پیغامی به صورت زیر روی صفحه نمایش ظاهر می‌شود:

Stack over flow

هنگامی که تابع بازگشتی به خود مراجعه می‌کند، مقادیر فعلی متغیرهای خود را در محلی از حافظه بنام پشته (stack) قرار می‌دهد، اگر بازگشت به تابع بازگشتی مجدداً صورت گیرد، مقادیر فعلی متغیرها مجدداً بدنبال مقادیر قبلی و اصطلاحاً در پشت مقادیر اولیه قرار می‌گیرند. هنگامی که شرط پایانی در تابع بازگشتی رخ می دهد، در اولین بازگشت مقادیری را که هنگام مراجعه به خود، در پشته نگهداری کرده، مجدداً در دسترس قرار می‌دهد و بهمین ترتیب در بازگشت‌های بعدی این عمل تکرار می‌شود تا مقدار تابع محاسبه شود.

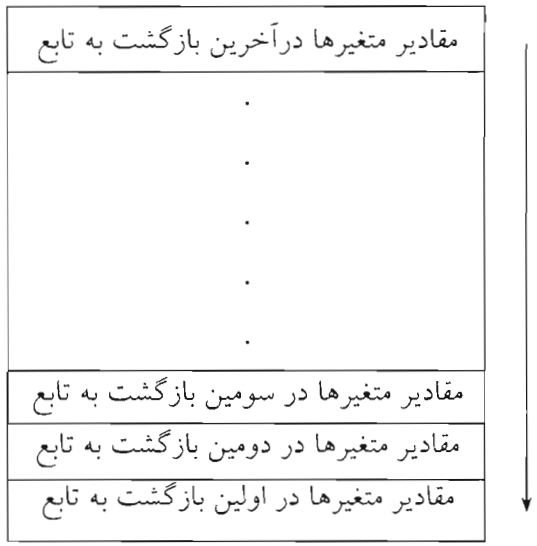
شکل زیر نحوه قرار گفتن مراجعه‌های یک تابع بازگشتی به پشته را نمایش می‌دهد.

ترتیب قرار گرفتن مقادیر در پشته:



نکته: در یک پشته نحوه دسترسی به عناصر آن از بالا به پایین و نحوه قرار گرفتن عناصر از پایین به بالا می‌باشد. اصطلاحاً پشته را به لوله آزمایش تشبیه می‌کنند یعنی اولین ورودی، آخرین خروجی (آخرین ورودی اولین خروجی) می‌باشد.

ترتیب دسترسی به مقادیر هنگام بازگشت از تابع بازگشتی از بالا به پایین می‌باشد و می‌توان آنرا بصورت زیر نمایش داد:



بابراین اگر برنامه‌نویس نتواند در برنامه، شرط پایانی توابع بازگشتی را کنترل نماید. حافظه‌ای که برای ذخیره‌سازی در نظر گرفته شده، پر خواهد شد و پیغام پر بودن یا سرریزی پشته ظاهر خواهد شد. (مقدار حافظه پشته محدود می‌باشد)

سوالی که مطرح است، اینست که چگونه می‌توان یک تابع بازگشتی نوشت. با کمی تامل در مسائل مطرح شده می‌توان، براحتی به سوال مطرح شده پاسخ داد. بعضی از مسائل ذاتاً بازگشتی هستند. برای مثال محاسبه فاکتوریل عدد را بررسی می‌کنیم:

همانطور که می‌دانیم فاکتور عدد مانند N را می‌توان بصورت زیر محاسبه کرد:

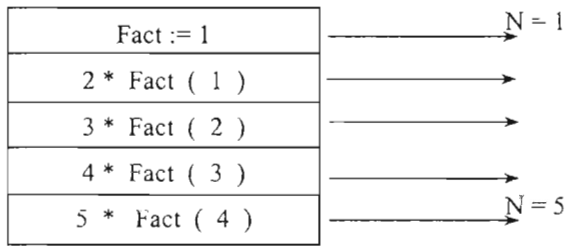
$$\begin{cases} N! = N*(N-1)! & \text{اگر} \quad N > 1 \\ N! = 1 & \text{اگر} \quad N = 0 \text{ OR } 1 \end{cases}$$

اگر در رابطه بالا دقت کنید، متوجه می‌شوید، تا زمانیکه N به یک یا صفر نرسد، مقدار فاکتوریل قابل محاسبه نیست. در این گونه مسائل استفاده از توابع بازگشتی، می‌تواند راحت‌تر و ساده‌تر از نوشتن توابع غیربازگشتی باشد. ولی همانطور که قبلاً گفتیم ممکن است، در بعضی مواقع سرعت برنامه را کاهش دهد.

مثال ۱۹-۱۱ برنامه‌ای بنویسید که با استفاده از توابع بازگشتی فاکتوریل عدد صحیح N را محاسبه نماید.

```
Program    Example19 ;
Var
       N : integer ;
Function     Fact (  N : integer ) :  Longint ;
Begin
       If  N = 1  Then
            Fact:= 1
       Else
            Fact:= N  *  Fact ( N - 1 ) ;
End ;
{************************************************************}
Begin
     Writeln ( ' Enter   Number ' ) ;
      Writeln ( ' Factorial  is = ', Fact ( N ) ) ;  { Call    Function }
End.
```

در برنامه بالا ترتیب قرار گرفتن مقادیر در پشته بصورت زیر می‌باشد:



بعد از شرط پایانی مقدار تابع بصورت زیر مرحله به مرحله محاسبه می‌شود:

مرحله اول :

Fact := 1

2 * Fact (1)
3 * Fact (2)
4 * Fact (3)
5 * Fact (4)

مرحله دوم :

Fact := 2 * 1

3 * Fact (2)
4 * Fact (3)
5 * Fact (4)

مرحله سوم :

Fact := 3 * 2

4 * Fact (3)
5 * Fact (4)

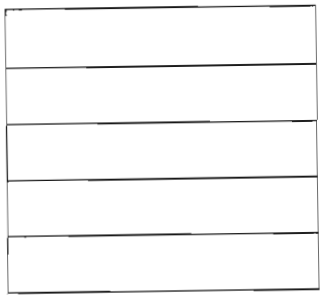
مرحله چهارم :

Fact := 4 * 6

5 * Fact (4)

: مرحله پنجم

Fact := 5 * 24



در نهایت مقدار ۱۲۰ توسط تابع به برنامه اصلی ارجاع داده می‌شود. در حالت کلی از روش بالا برای محاسبه مقدار تابع بازگشتی بصورت دستی عمل می‌کنند.

در مورد روال‌ها نیز می‌توان از روال‌های بازگشتی بهره برد و تقریباً مثل توابع عمل می‌کنند. فقط همان تفاوت بین تابع و روال به قوت خود باقی است.

مثال ۲۰-۱۱ شخصی حقوق اولیه‌اش ۱۸۰۰۰۰۰ ریال است و هر سال ۵۰۰۰۰۰ ریال، به اضافه ۲۰ درصد افزایش حقوق دارد. حقوق این شخص در سال m ام از رابطه زیر بدست می‌آید:

$$a_m = \begin{cases} 1800000 & \text{if} \quad m = 1 \\ 1.2\, a_{m-1} + 500000 & \text{if} \quad m >= 2 \end{cases}$$

اگر بخواهیم، حقوق فرد را در سال N ام محاسبه کنیم، از روال بازگشتی زیر می‌توان استفاده کرد:

```
Procedure     CalculateSal ( N: Integer ; Var   Salary: Real ) ;
Begin
        If     N = 1    Then
               Salary: = 1800000
        Else
               Begin
                     CalculateSal ( N – 1 , Salary ) ;
                     Salary:= 1.2 * Salary  +  500000 ;
               End ;
   End. { End  Of  Procedure }
```

مثال ۲۱-۱۱ تابعی برای محاسبه بزرگترین مقسوم علیه مشترک دو عدد صحیح N , M بنویسید.

```
Function     GCD (  M , N: integer ) :  Integer ;
Begin
      If  ( N <= M )  and  ( M  Mod  N  =  0 )  Then
            GCD:= N
      Else
            GCD:=  GCD ( M , N ) ;
End ;
```

۴-۱۱- مقایسه توابع و روال‌ها

توابع و روال‌ها هر دو برنامه‌های فرعی هستند، که بطور مستقل وظایفی را بر عهده دارند. ولی در این میان از بعضی جنبه‌ها متفاوت می‌باشند که عبارتند از:

- نحوه فراخوانی آنها با هم متفاوت است. روال‌ها از طریق عبارات روال فراخوانده می‌شوند، درصورتی که فراخوانی تابع توسط عبارات مقایسه‌ای و یا تخصیص نتیجه به یک متغیر صورت می‌گیرد.
- هنگام اعلان یک تابع، نوع تابع یا نوع نتیجه حاصل از تابع باید ذکر شود، در صورتی که روال‌ها نیازی به این کار ندارند.
- توابع فقط یک خروجی برمی‌گردانند، ولی روال‌ها می‌توانند، چندین خروجی برگردانند. درضمن نتیجه توابع توسط اسم تابع فرستاده می‌شود ولی روال‌ها از طریق پارامترها، نتایج را برمی‌گردانند.

۵-۱۱- طریقه ارسال آرایه‌ها به توابع و روال‌ها

آرایه‌ها خود مجموعه‌ای از داده‌ها می‌باشند، لذا برای ارسال آنها نمی‌توانیم از روش معمولی ارسال پارامترها استفاده کنیم. و بطور مستقیم نمی‌توان آنها را به برنامه‌های فرعی انتقال داد. برای ارسال آرایه‌ها به عنوان پارامتر به برنامه‌های فرعی از دستور Type استفاده می‌کنند. دستور Type قبل از تعاریف برنامه اصلی بکار می‌رود و توسط این دستور در واقع یک نوع ساده‌سازی در تعاریف فراهم می‌شود. و با این ساده‌سازی می‌توان داده‌های ساخت یافته را به برنامه‌های فرعی منتقل کرد.

فرم کلی دستور Type در بخش تعاریف بصورت زیر است:

```
Type     شناسه   =   تعریف داده ساخت یافته
```

برای مثال یک ارایه از نوع صحیح را تعریف می‌کنیم:

```
Type        No = array [1...100] of integer ;
```

در اینصورت تعریف آرایه به، No نسبت داده می‌شود. اگر بخواهیم متغیری از نوع آرایه بالا تعریف کنیم، آنرا از نوع No تعریف می‌کنیم:

```
Var
        Number: No ;
```

در اینصورت متغیر Number یک آرایه ۱۰۰ عنصری از نوع صحیح خواهد بود.

برای ارسال آرایه‌ها به توابع و روال‌ها نیز مثل بالا عمل می‌کنیم.

مثال ۲۲-۱۱ برنامه‌ای بنویسید که یک آرایه حداکثر ۱۰۰ عنصری را از ورودی دریافت کرده، سپس عدد را از ورودی خوانده، توسط تابعی بنام Search محل وقوع عدد در آرایه را نمایش دهد.

```
Program    Example22 ;
Type       a = array [1.. 100]  of   integer ;
Var
           B: a ;
           N : Word ;
Function    Search ( b:a ; N: Word ; x: integer ) : Word ;
Var
      I , Index: Word ; Flag: Boolean ; { local variable }
 Begin
       Index:= 0 ;
       Flag:= TRUE ;
       While  ( I <= N ) And ( flag )  do
           Begin
                If  b[ I ] = x  Then
                   Begin
                        Index:= I ;
                        Flag:= FALSE ;
                   End ;
                Inc ( I ) ;
           End ;
        Search:= Index ;
 End ;
{*************************************************************}
Begin  { Main }
      Writeln ( ' Enter  Number ' ) ;
      Readln ( N ) ;
      For   I:= 1   To   N    Do
            Readln ( b[ I ] ) ;
      Writeln ( ' Enter  Number ' ) ;
      Readln ( x ) ;
       If   Search ( b , N , x ) = 0  Then
            Writeln ( ' Not  Found ' ) ;
        Else
            Writeln ( Search( b , N , x )  ) ;
 End. { End  Of Program }
```


همانطور که ملاحظه می‌کنید، با دستور Type توانستیم یک آرایه بعنوان پارامتر به تابع ارسال کنیم.

مثال ۲۳-۱۱ برنامه‌ای بنویسید که یک عدد از ورودی دریافت کرده، سپس توسط روالی بنام CreatePrime اعداد اول قبل از عدد خوانده شده را، تولید و در یک آرایه قرار دهد و در نهایت آرایه حاصل را در برنامه اصلی نمایش دهد.

```
Program    Example23 ;
Type       p = array [1.. 100]  of   integer ;
Var
           Prime : p ;
           I , M ,N : Integer ;
Procedure     CreatePrime ( Var prime: p ; Var  k: integer ; N: Integer ) ;
Var
      I , J : Integer ; Flag: Boolean ; { local variable }
 Begin
       Prime [1]:= 2 ;
       Prime [2]:= 3;
       K:= 3 ; Flag:= TRUE ;
       For   I:= 4  to   N   do
          Begin
               For   J:= 2  to  ( I Div 2 )  do
                   If   I   Mod  J  = 0  Then
                       Flag:= FALSE ;
                If    Flag  Then
                   Begin
                        Prime [k]:= I ;
                        Inc ( k ) ;
                    End ;
                Flag:= TRUE ;
          End ;
 End ;
{*************************************************************}
Begin  { Main }
     Writeln ( ' Enter  Number ' ) ;
     Readln ( N ) ;
     For   I:= 1   To   100    Do
           Prime [ I ]:= 0 ;
     M:= 0 ;
     CreatePrime ( Prime , M , N ) ;
     Writeln ( ' The Prime Number  befor  N ' ) ;
      For   I:= 1   To   M    Do
           Writeln ( Prime[ I ] ) ;
End. { End  Of Program }
```

۶-۱۱- ارائه چند مثال از این فصل

دراین بخش قصدداریم، تعدادی برنامه در مورد کاربرد توابع و روالها ارائه دهیم

مثال ۲۴-۱۱ برنامه‌ای بنویسید که اسم حداکثر ۱۰۰ دانشجو را از ورودی دریافت کرده، سپس توسط روالی بنام sort اسامی را بر اساس حروف الفبا مرتب کرده و نتیجه را در برنامه اصلی چاپ نماید.

```
Program    Example24 ;
Type       Na  = array [1.. 100]  of    string ;
Var
           Name: Na ;
           I , M  ,N : Integer ;              { Global  Variables }
Procedure      Sort ( Var  Name: Na  ;  N: Integer ) ;
Var
       I , J : Integer ;  { local  variable }
       Temp: String ;
Begin
       For   I:= 1   to    N -1    do
          Begin
               For   J:= I + 1   to   N    do
                   If   Name[I] > Name[J]   Then
                     Begin
                          Temp:= Name[ I ] ;
                           Name[ I ]:= Name[ J ] ;
                           Name[ J ] :=  Temp ;
                      End ;
 End ;
{************************************************************}
Begin   { Main }
     Writeln ( ' Enter  Number ' ) ;
     Readln (  N ) ;
     For    I:= 1    To    N     Do
            Readln ( Name [ I ] ) ;
      Sort ( Name , N ) ;
     Writeln ( ' The  Sorted  List  ' ) ;
      For    I:= 1    To    M     Do
            Writeln (  Name[ I ]  ) ;
End. { End   Of  Program }
```

در برنامه بالا روال sort یک آرایه N عنصری (N<=100) را بر اساس حروف الفبا مرتب نموده و نتیجه را به برنامه اصلی ارسال می‌کند.

مثال ۲۵-۱۱ برنامه‌ای بنویسید، که یک ماتریس ۵×۵ را از ورودی دریافت کرده، سپس توسط روالی بنام Change، بیشترین و کمترین مقدار ماتریس را پیدا کرده آنها را با هم جابجا نماید و در نهایت ماتریس حاصل را در برنامه اصلی چاپ نماید.


```
Program    Example25 ;
Type       a  = array [1.. 5 , 1...5]  of    integer ;
Var
           Mat: a  ;
           I , J : Integer ;              { Global  Variables }
Procedure        Change ( Var  Matrix: a  ) ;
Var
       Temp , Mini , Minj , Maxi , Maxj , max , Min : Integer ;  { local  variable }
       I , j : Integer ;
Begin
       Mini:= 1 ;  Minj:= 1 ;
       Min:= Matrix[ 1 , 1 ] ;
       Maxi:= 1 ; Maxj:= 1 ;
       Max:= Matrix [ 1 , 1 ] ;
       For   I:= 1   to   5   do
            For   J:= 1   to   5   do
                Begin
                     If   Matrix[i,j] > Max   Then
                       Begin
                            Max:= Matrix[i,j] ;
                            Maxi:= I  ;
                            Maxj:= J  ;
                        End
                     Else If   Matrix[i,j] < Min   Then
                       Begin
                            Min:= Matrix[i,j] ;
                            Mini:= I  ;
                            Minj:= J  ;
                        End ;
       Temp:= Matrix[ Maxi ,j] ;
       Matrix [Maxi , Maxj]:= Matrix [ Mini , Minj ] ;
       Matrix [ Mini , Minj ]:=  Temp ;
 End ;
{************************************************************}
Begin   { Main }
     Writeln ( ' Enter  Matrix ( 5 * 5 ) ' ) ;
     For    I:= 1    To    5     Do
                For    j:= 1    To    5     Do
                       Readln ( Mat[ I , J] ) ;
      Change ( Mat ) ;
      Writeln ( ' The Resualt  Matrix  ' ) ;
      For    I:= 1    To   5     Do
      Begin
          For    I:= 1    To    5     Do
                 Write (  Mat[ I ,J ]: 5  ) ;
          Writeln;
       End ;
End. { End   Of  Program }
```

مثال ۲۶-۱۱ برنامه‌ای بنویسیدکه اطلاعات مربوط به حداکثر ۱۰۰ دانشجو که عبارتند از:

فامیلی Fam
معدل Ave
ارزشیابی Value

Value از ورودی دریافت نمی شود.

الف) توسط روالی بنام scan از ورودی دریافت شود.(فامیلی و معدل)

ب) توسط روالی بنام Initial آرایه ارزشیابی را بصورت زیر مقداردهی نماید:

معدل	ارزشیابی
16 – 20	A
14 – 16	B
12 – 14	C
10 – 12	D
0 – 10	F

ج) توسط تابعی بنام Count تعداد دانشجویانی که نمره اول (A) را کسب نموده‌اند محاسبه نماید.

د) میانگین معدل کلاس را توسط تابعی بنام CalcAve محاسبه کند.

```
Program    Example26 ;
Type
        Family   = array [1.. 100]  of   String ;
        Average  = array [1.. 100]  of   Real ;
        Value    = array [1.. 100]  of   Char ;
Var
           Fam: Family ;
           Ave: Average ;
           Val: value ;
           I , N : Integer ;              { Global Variables }
{*********************************************************}
Procedure      Scan ( Var  Fam: Family  ; Var Ave: Average ;  N: Integer ) ;
Var
       I : Integer ; { local variable }
 Begin
       For   I:= 1   to   N    do
          Begin
               Writeln ( ' Enter Family ' ) ;
               Readln (  Fam[ I ]  ) ;
               Writeln ( ' Enter Average ' ) ;
               Readln (  Ave[ I ]  ) ;
          End;
End;
{*********************************************************}
Procedure      Initial ( Var  Val: Value  ; Ave: Average ;  N: Integer ) ;
Var
```



```
       I : Integer ; { local variable }
Begin
       For   I:= 1   to   N    do
          If    ( Ave[i] >= 16 )  And ( Ave[i] <= 20 )  Then
                Val[i]:= 'A'
          Else If    ( Ave[i] >= 14 )    Then
                Val[i]:= 'B'
          Else If    ( Ave[i] >= 12 )    Then
                Val[i]:= 'C'
          Else If    ( Ave[i] >= 10 )    Then
                Val[i]:= 'D'
          Else
                Val [i]:= 'F' ;
End;
{*********************************************************}
Function       Count  (Val: Value  ;  N: Integer ) : Integer ;
Var
       I , P : Integer ; { local variable }
Begin
       P:= 0;
       For   I:= 1   to   N    do
          If    ( Ave[i] = 'A' )   Then
                Inc ( p ) ;
       Count:= p ;
End ;
{*********************************************************}
Function       CalcAve  (  Ave: Average  ;  N: Integer ) : Real ;
Var
       I : Integer ; { local variable }
       Sum: Real ;
Begin
       Sum:= 0;
       For   I:= 1   to   N    do
             Sum:= Sum  +  Ave [i]  ;
       Count:= Sum / N  ;
End ;
{*********************************************************}
Begin  { Main }
     Writeln ( ' Enter  Number  Of  Data ' ) ;
      Readln (  N ) ;
      Scan ( Fam , Ave , N );
      Initial ( Val , Ave , N ) ;
      Writeln ;
      Writeln ( '   Family          Average          Value ') ;
      For   I:= 1    To    N    Do
           Writeln( Fam[i] , Ave [i]:8:2 , Val [i]: 8 ) ;
      Writeln ;
      Writeln ( ' The  Number  Of  Best  Average ') ;
      Writeln ( Count ( Val , N ) );
      Writeln ( ' The Average Of  Class  ');
      Writeln ( CalaAve( Ave , N ): 8:2 );
End. { End  Of Program }
```

در برنامه فوق خود مساله، تمام توابع و روال‌های لازم را مطرح کرده بود. ولی در حالت کلی وقتی مساله ای مطرح می‌شود، باید برنامه‌نویس بتواند، مساله را تجزیه و تحلیل نماید. منظور از تجزیه و تحلیل، تشخیص راه حل، تعداد توابع، تعداد روال‌ها، حتی تعداد متغیرها و اسامی متغیرها می‌باشد. و تا زمانیکه یک مساله یا در حالت کلی یک سیستم، تجزیه و تحلیل نشود نمی‌توان برنامه اصولی برای سیستم مطرح شده، نوشت.

نکته: همانطور که قبلاً گفتیم، برنامه‌های فرعی، قطعات یک برنامه هستند. در نوشتن توابع تا جایی که امکان داشته باشد، سعی می‌کنند، کاملاً مستقل از برنامه نوشته شود. تا بتوانند در برنامه‌های دیگر نیز از تابع نوشته شده استفاده کنند. (اصطلاحاً قطعات با قابلیت استفاده مجدد باید ساخت که در مهندسی نرم‌افزار بحث می‌شود).

مثال ۲۷-۱۱ برنامه‌ای بنویسید که عدد صحیحی را از ورودی دریافت کرده توسط تابع بازگشتی fib جمله N ام سری فیبوناچی را تولید نماید.

```
Program    Example27 ;
Var
          N : Integer ;              { Global Variables }
Function       Fib ( N: Integer ) :  integer ;
Begin
       If   N <= 2  Then
            Fib:= 1
        Else
             Fib:= Fib ( N – 1 )  +  Fib ( N – 2 ) ;
End ; { End Of Function }
Begin
       Writeln ( ' Enter  Number ' ) ;
        Readln ( N ) ;
         Writeln ( Fib ( N ) ) ;
End. {  End   Of   Program }
```

مثال ۲۸-۱۱ روال بازگشتی برای جستجوی دودوئی (binary search) بنویسید.

ساختار داده‌های این تابع بصورت زیر است:

first ⟵ ابتدای آرایه

last ⟵ انتهای آرایه

list ⟵ آرایه

element ⟵ عنصر مورد جستجو


```
Procedure      Search ( Var  List: IntArray  : Element : integer
                        First , Last: Integer ;
                        Var  Index: Integer ;
                         Var  Found: Boolean  ) ;
Var
       middle  : Integer ;  {  local  variable  }
 Begin
        middle:= ( First  +  Last )  Div  2 ;
        If  First  >  Last   Then
              Found:= FALSE
        Else if   Element = List [ middle]  Then
          Begin
                Found:= TRUE ;
                 Index:= middle ;
          End
        Else if   Element < List [ middle]  Then
                  Binary ( list , element , first, index-1 , found )
        Else     Binary ( list , element , index+1,last, found) ;
End ; {  End  Of     Function  }
```

۷-۱۱- تمرینات

۱- روال Down را در نظر بگیرید:

```
Procedure      Down ( N: Integer ) ;
Begin
        While N > 0 do
         Begin
              Write ( N: 3 ) ;
               Dec ( N ) ;
         End ;
End ;
```

الف) وقتی روال بصورت down (5) فراخوانی شود، چه چیزی چاپ می‌شود.

ب) مقدار پارامتر واقعی N بعد از اجرای روال چیست؟

۲- قسمتی از یک برنامه با روال‌های تودرتو در زیر آمده است. حوزه عمل هر شناسه را همراه با نام روال‌ها مشخص نمائید.

```
Program    Ex2 ;
Const    pi = 3.14159 ;
Procedure          A ( Var  X: Real  ) ;
Var
        B , C : Integer  ;
Procedure          D ( Var  S: Real  ) ;
       Const  Star ='*';
        Begin
              ..............
              ..............
        End ; { D }
 Begin  { A }
         ...........
         ...........
 End ; { A }
Procedure          F ( Var  X , Y: Real  ) ;
  Var
         D: integer ;
    Begin
          ..........
          ..........
     End ; { f }
Begin  { Main }
       ...........
       ...........
End.
```


۳- خروجی برنامه زیر را تعیین کنید:

```
Program    Ex3 ;
Var
        a , b , c : Integer  ;              { Global  Variables  }
Procedure       test  ( Var  b: integer  ;  a: Integer ) ;
Var
        c : Integer ;  {  local  variable  }
Begin
        C:= 10 ;
        B:= a + c ;
        C:= C + 1 ;
        A:= b + c ;
         Writeln ( a , b , c ) ;
 End ;
{*********************************************************}
Begin   { Main  }
        A:= 1 ;
        B:= 2 ;
        C:= 3;
        Test ( a , b, c ) ;
        Writeln ( a , b , c  ) ;
End. { End   Of  Program }
```

۴- تابع بازگشتی زیر را در نظر بگیرید:

```
Function       MyStery  (  M , N: Integer ) :  integer  ;
Begin
      If   N =  1   Then
           MyStery:= 1
       Else
           MyStery:= M *  MyStery ( M , N - 1 )  ;
End ; { End Of  Function }
```

الف) تابع بازگشتی بالا را به ازاء مقادیر N=4 , M=5 فراخوانی کنید.

ب) تابع بازگشتی بالا را به ازاء مقادیر N=5 , M=3 فراخوانی کنید.

۵- تابع بازگشتی زیر را در نظر بگیرید:

```
Function       test   (  N: Integer ) :  integer  ;
Begin
      If   N <= 2   Then
           Test := 1
       Else
           Test := N *  Test (  N – 2 )  ;
End ; { End Of  Function }
```

تابع بالا را بازاء N=9 , N=12 اجرا نمائید.

۸-۱۱- تمرینات برنامه‌نویسی

۱- برنامه‌ای بنویسید، که عددی را از ورودی دریافت کرده سپس :
الف) توسط روالی بنام prime اول بودن عدد را بررسی نماید.
ب) توسط روالی بنام compelete کامل بودن عدد را بررسی نماید.
ج) توسط‌روالی‌بنام‌fib بررسی‌کند که آیاعددخوانده شده‌جزء سری فیبوناچی‌هست یانه؟

۲- برنامه‌ای بنویسید که اطلاعات حداکثر ۱۰۰ دانشجو که عبارتند از:

Name	اسم
Family	فامیلی
St_No	شماره دانشجوئی

را از ورودی دریافت کرده سپس:
الف) توسط روالی بنام sort اطلاعات را برحسب شماره دانشجوئی مرتب کند.
ب) توسط تابعی بنام search شماره دانشجوئی فردی را از ورودی دریافت نماید، در صورتی که شخص موردنظر در لیست باشد، سایر اطلاعات آنرا نمایش دهد.

۳- برنامه‌ای بنویسید که با اطلاعات حداکثر ۵۰ دانشجو که عبارتند از:

Name	اسم
St_No	شماره دانشجوئی
Mark1	نمره درس ۱
Mark2	نمره درس ۲
Mark3	نمره درس ۳

را از ورودی دریافت کرده سپس با استفاده از برنامه‌های فرعی موارد زیر را در برنامه اعمال نماید:
الف) بالاترین نمره در هر درس به چه فرد یا افرادی تعلق دارد، اطلاعات آنها را در برنامه اصلی چاپ نماید.
ب) میانگین نمره هر درس را محاسبه نماید و با پیغام مناسب چاپ نماید.

ج) اسامی افراد را بترتیب معدل آنها (درسه درس) محاسبه و چاپ نماید.
د) میانگین معدل کلاس و نزدیک‌ترین معدل به میانگین را محاسبه و چاپ کند.
هـ) معدل افراد را در سه طبقه ۰ تا ۹، ۹ تا ۱۲، ۱۵ تا ۲۰ دسته‌بندی کرده و هیستوگرام آن را رسم نماید.

۴- آرایه‌ای از نوع صحیح با حداکثر ۱۰۰ عنصر را در نظر بگیرید.
برنامه‌ای بنویسید که ابتدا آرایه را از ورودی دریافت کرده سپس با استفاده از یک تابع بازگشتی بیشترین مقدار آرایه را محاسبه نماید.

۵- دو آرایه مرتب حداکثر ۱۰۰ عنصری که شامل اسامی افراد می‌باشد را درنظر بگیرید.
برنامه‌ای بنویسید که نخست دو آرایه را از ورودی دریافت نموده سپس توسط روالی بنام merge این دو آرایه در هم ادغام نموده و در آرایه سومی قرار دهد.
۶- اطلاعات مربوط به حداکثر ۱۰۰ استاد که عبارتند از:

Name	اسم
Family	فامیلی
Salary	حقوق

را در نظر گرفته، برنامه‌ای بنویسید، که نخست اسم و فامیلی و حقوق پایه اساتید را از ورودی دریافت نماید سپس:

الف) اطلاعات را بر اساس فامیلی و اسم مرتب کند.
ب) حقوق اساتید بر اساس سابقه تدریس بصورت زیر محاسبه نماید.

سابقه تدریس	ضریب حقوق
0 – 2	%5
2 – 5	%10
5 – 7	%20
7 – 10	%30
10 – 30	%50

ضریب حقوق + حقوق پایه = حقوق اصلی

ج) برای محاسبه حقوق دریافتی اساتید تابعی بنویسید که اعمال زیر را انجام دهد.

(کسرمالیات)

حقوق	کسورات قانونی
0 – 1700000	0
1700000 – 2000000	%10
2000000 – 4000000	%15
4000000 – 6000000	%20
6000000	%50

دقت کنید که درصدهای تعیین شده باید از حقوق اصلی کسر شده مابقی بعنوان حقوق پرداختی نمایش داده شود.

فصل ۱۲

مجموعه‌ها و داده‌های شمارشی

هدفهای کلی

- مفهوم مجموعه و داده‌های شمارشی در زبان پاسکال
- مجموعه و داده‌های شمارشی به عنوان متغیر
- استفاده از مجموعه‌ها و داده‌های شمارشی در برنامه

هدفهای رفتاری

دانشجو پس از مطالعه این فصل باید بتواند:

- مفهوم مجموعه و داده‌های شمارشی را درک کند.
- بتواند در صورت لزوم از مجموعه و داده‌های شمارشی استفاده کند.

مقدمه

تعداد محدودی از داده‌ها که از نظر نوع یکسان هستند، در غالب مجموعه و یا گونه‌های شمارشی نگهداری می‌شوند که مفهومی شبیه در ریاضیات دارند. برای استفاده در مواردی خاص نظیر روزهای هفته و یا نوع ماشینها و... که ترتیبی هستند و یا یک مجموعه از داده‌های پشت سر هم،می‌باشند، استفاده از این ساختارها کار را بسیار راحت می‌کند. هر چند وجود آنها به عنوان ساختارهای داده‌ای، الزامی‌نیست. بهرحال به عنوان ابزارهایی از زبان پاسکال هستند که در مواقعی، ضروری بنظر می‌رسند و مسئله را به صورتی قابل‌فهم و راحت حل می‌کند.

۱-۱۲- مجموعه‌ها (Sets)

در زبان پاسکال مجموعه، مفهومی شبیه به مفهوم مجموعه در ریاضیات جدید دارد. متغیری است که شامل لیستی از اعداد صحیح، کاراکتر، بولین و یا از نوع شمارشی می‌باشد که دارای تعداد عناصر محدود به حداکثر ۲۵۶ تا می‌باشد. از این جهت بسیار شبیه به یک آرایه در زبان پاسکال است که شامل داده‌هایی از یک نوع می‌باشد، ولی آرایه دارای عناصر محدودی نیست و در ضمن مانند آرایه تعریف نمی‌شود.

۱-۱-۱۲- تعریف مجموعه

برای تعریف یک مجموعه از کلمات کلیدی set of استفاده می‌کنیم که در زیر نمونه‌هایی از تعاریف مجموعه‌های مختلف آمده است:

```
Type
Digit_type = 0..9;
Alpha_type = 'a'..'z';
Capital_type = 'A'..'Z';
Bool_type = False..True;
```

```
Type
    Digit = set of Digit_type;        ⟶  مجموعه‌ای از کاراکترهای ۰ تا ۹
    Alpha = Set of Alpha_type;        ⟶  مجموعه‌ای از کاراکترهای a تا z
    Capital = Set of Capital_type;    ⟶  مجموعه‌ای از کاراکترهای A تا Z
    Bool = Set of Bool_type;          ⟶  مجموعه دو عضوی از F و T
```



```
Var
    d: Digit;                         ⟶  d متغیری از مجموعه Digit می‌باشد.
    a1,a2: Alpha;                     ⟶  a1 و a2 متغیرهایی از مجموعه Alpha است.
    c: Capital;                       ⟶  c متغیرهای از مجموعه Capital است.
    b: Array [1..10] of Bool;         ⟶  b آرایه ۱۰ عضوی از Bool است.
```

حال باتوجه به تعاریف فوق درداخل برنامه می‌توانیم مقداردهی‌های زیر راداشته باشیم:

```
d:=[2,4,6,8];
d:=[3..6,8,9];
a1:=[a,b,c];
a2:=[x,y,z];
c:=[u,v,w,x,y,z];
b[1]=[true];
b[2]:=[flase];
b[3]:=[];
b[4]:=[true,flase];
b[5]:=[flase,true];
b[6]:=[false,false,true,false];
```

با توجه به تعریف مجموعه در ریاضیات، مجموعه تهی مجموعه‌ای است که هیچ عضوی نداشته باشد. مانند b[3] که با علامت [] نشان داده می‌شود. تکرار عضوها در مجموعه‌ها تاثیری ندارد و فقط یک عضو درنظر گرفته می‌شود. مثلاً در b[6] عضو false سه بار تکرار شده است و فقط یکی معتبر است.

همچنین جابه‌جایی عضوها تاثیری ندارد، در نتیجه b[6] و b[4] و b[5] سه مجموعه مساوی هستند زیرا دارای اعضای یکسان، هر چند جابه‌جا می‌باشند ولی جابه‌جایی عضوها تاثیر ندارد و همچنین دارای عضوهای تکراری نیز هستند. با توجه به موارد گفته شده، کلیه موارد زیر با توجه به تعاریف فوق غلط هستند و خطای کامپایلری وجود دارد:

```
d:=3;
d:=['3','4'];
d:=[a,b,c];
a:=[B,C,D,E];
c:=[1,2,a,A,b];
c:=[A,B, a,b,c,d];
b[1]:=[1,2];
b[2]:=false;
b[3]:=true,false;
```

همانطور که قبلاً گفته شد، تعداد عناصر یک مجموعه به ۲۵۶ تا ختم می‌شود که حداکثر تعداد عضوهای یک مجموعه می‌باشد و لذا از تایپهایی نظیر byte, boolean, char, و شمارشی استفاده می‌گردد. زیرا دارای تعداد عناصر محدود به تا ۲۵۵ دارند، که معادل ۲۵۶ تا می‌باشد. پس بدلیل گستردگی تایپهایی نظیر real, integer, string نمی‌توان مجموعه‌ها را از آن نوع‌ها تعریف کرد. تعاریف زیر را درنظر بگیرید که همگی درست می‌باشند و در پاسکال مورد تأیید است، زیرا محدود به ۰ تا ۲۵۵ یعنی ۲۵۶ عضو هستند.

```
Var
    ch: set of char;        ⟶   #0..#255
    bool: set of boolean;   ⟶   false..true
    num: set of byte;       ⟶   0..255
    x: set of 100..200;
    y: set of 'a'..'z';
    z: set of '0'..'9';
```

و همچنین تعاریف زیر بدلیل گستردگی بیشتر از ۲۵۶ عضو و یا خارج از حد ۰ تا ۲۵۵ و یا عدم ترتیب منطقی دارای خطای کامپایلری در زبان پاسکال هستند:

```
var
    r: set of real;
    i: set of integer;
    s: set of  string;
    num1: set of -100..100;
    num2: set of 0..1000;
    x: set of 6..0;
    y: set of 'z'..'u';
    z: set of True..False;
    u: set of 'A'..'d';
```

۲-۱-۱۲- عملیات روی مجموعه‌ها

در ریاضیات عمل عضویت وجود دارد که به این معنی است که متغیری عضو مجموعه می‌باشد یا خیر. این عمل در زبان پاسکال با کلمه کلیدی in صورت می‌گیرد. اگر عضویت صحیح باشد جواب True وگرنه False می‌باشد. همچنین دو مجموعه را می‌توان با علامات شرطی =، <>، =< و >= مقایسه کرد که همگی دارای خروجی درست یا غلط می‌باشند. ولی‌علامات شرطی > و < در مورد مجموعه‌ها کاربردی ندارد.

مثال ۱-۱۲ :

```
Var
    a: set of byte;
Begin
    a:= [1..3,4];
    if  3 in a then write('3 is in set a')
    else write('3 is not in set a');
    write(3 in a);
    write(8 in a);
    write([1,2,3,4,] = a);
    write([1,2,3,4] = [2,2,1,1,1,3,4,4]);
    write([1,2] = [1,2,3]);
    write([1,2] <= [1,2,3]);
    write([] >= [1]);
End.
```

چاپ می‌شود.

```
True
False
True
True
False
True
False
```

در ریاضیات می‌توانیم مجموعه‌ها را با هم اجتماع، اشتراک و تفاضل کنیم که این عملیات در پاسکال با عملگرهای +، * و - به ترتیب می‌باشد. اجتماع دو مجموعه ترکیبی از همه عضوهای آنها است و اشتراک یعنی عضوهایی که در هر دو مجموعه مشترک است و تفاضل یعنی اعضایی که در مجموعه اول می‌باشد و در مجموعه دوم وجود ندارد. به مثال زیر توجه کنید:

مثال ۲-۱۲

```
Var
    a: set of byte;
    b: set of 0..10;
    c,d,e: set of byte;
Begin
    a:=[0,1,2,3];
    b:=[2,3,4,5];
    c:= a+b;
    d:= a*b;
    e:= a-b;
End.
```

```
c=[0,1,2,3,4,5]=[0..5]
d=[2,3]
e=[0,1]
```

در هنگام برنامه‌نویسی توجه داشته باشید که آرگومان ورودی روالها می‌تواند مجموعه باشد که قبلاً در تایپ تعریف شده باشد، نه اینکه مستقیم در روال به عنوان آرگومان بیاید. ولی خروجی یک تابع نمی‌تواند از نوع مجموعه باشد. برای نوشتن یا خواندن مجموعه‌ها باید عضو به عضو عملیات صورت بگیرد و مستقیماً توابع Write، Read روی آنها کار نمی‌کنند. لذا تعاریف زیر صحیح نیستند:

```
Type
      ch = set of char;
      function F1(var ch1: set of char): integer;
      procedure F2(ch2: set of 0..9);
      function F3(x:integer): ch;
```

ولی تعاریف زیر صحیح است:

```
function F4(ch3:ch): char;
procedure F5(var ch4: ch; ch5: ch);
```

مثال ۳-۱۲ برنامه‌ای بنویسید که تعداد محدودی عدد مختوم به ۱- را از ورودی بخواند و در یک مجموعه از اعداد صحیح قرار بدهد. سپس اعضای این مجموعه را با توجه به مجموعه ساخته شده در خروجی چاپ کند:

```
Program test;
Var
      num,temp: set of byte;
      i,c: integer;
Begin
      Writln('Enter numbers: ');
      Readln(i);
      Num:= [];
      c:= 0;
      while i  > -1 do begin
                c:=c+1;
                temp:=[i];
                num:=num+temp;
                Readln(i);
      end;
      for  i:= 0 to 255 do
                if i in num then write(i:5);
End.
```

در مثال فوق num مجموعه‌ای است که قرار است اعداد خوانده شده بین ۰ تا ۲۵۵ در آن نگهداری شوند و temp یک مجموعه کمکی است. هر عدد که خوانده می‌شود به صورت تکی در آن نگهداری می‌شود. c تعداد اعداد خوانده شده را در خود دارد و در انتها با توجه به اینکه اعداد داخل num بین ۰ تا ۲۵۵ هستند، لذا با شرط وجود آنها در num چاپ می‌شوند.

مثال ۴-۱۲ برنامه‌ای بنویسید که مجموعه‌ای از کاراکترهای a تا z را در برگیرد و تعداد اعضای آنرا بدست آورد و در خروجی چاپ کند؟


```
Var
      ch: set of 'a'..'z';
      i: char;
      count: integer;
Begin
      Ch:= ['a','c','d','p'..'x','z'];  ⟶  مجموعه نمونه برای مثال
      count:=0;
      for i:= 'a' to 'z' do
                if i in ch then count:= count + 1;
      write ('The number of characters in set is: ' , count);
End.
```

۲-۱۲- داده‌های شمارشی (Enumeration)

داده‌های شمارشی، یک مجموعه مرتب از اعداد است که در برنامه، هر عدد دارای نام بخصوصی است. این نامها در داخل دو پرانتز باز و بسته قرار می‌گیرند و بترتیب از صفر مقدار می‌گیرند مگر اینکه برنامه‌نویس به آنها مقدار مخصوصی بدهد. به مثال زیر توجه کنید که در آن گونه‌های ماشینها را لحاظ می‌کند:

```
Type
      Cars_type = (Peykan, Pride, Pegout, PK);

Var
      Cars: cars_type;
```

حال در برنامه cars می‌تواند مقادیر داخل داده‌های شمارشی را بگیرد:

```
cars:= Pride;
cars:= PK;
```

و موارد زیر با توجه به تعریف غلط می‌باشند:

```
cars:= 'Pride';  ⟶  رشته Pride مورد نظر است.
cars:= 2;
cars:= Pride + PK;
```

همانطور که گفته شده است داده‌های شمارشی بترتیب از مقدار صفر شماره‌گذاری می‌شوند. لذا داده‌های شمارشی از نوع ترتیبی است و کلیه توابع ترتیبی نظیر succ، pred و ord در مورد آنها صادق است. یادآوری می‌شود که succ یعنی مورد بعدی، pred یعنی مورد قبلی و ord مقدار داده را برمی‌گرداند. باتوجه به مثال زیر داریم:

```
Type
      Days_type = (sat, sun, mon, tue, wed, thu, fri);
Var
      Day: Days_type;
Begin
      Day:= Mon;
      i:= ord (sat) + ord (day) + (succ(sat));
      j:= ord ( pred (sun) + ord (pred (fri));
      writeln(i:5, j:5);
End.
```

برای i و j داریم:

i = 0+2+1 = 3
j = 0+5 = 5

لذا اعداد ۳ و ۵ در خروجی چاپ خواهند شد.

توجه داشته باشید succ(fri) و pred(sat) تعریف نشده هستند و خطای کامپایلری وجود دارد. همچنین توجه داشته باشید که متغیر داده‌های شمارشی را نمی‌توان با Read و یا Write استفاده کرد، یعنی مستقیماً نوشت و یا خواند.

اسامی داده‌های شمارشی ساده شبیه اسامی متغیرها در زبان پاسکال هستند و با " و" نمی‌توان بکار برد. همچنین برخلاف رکوردها که می‌توانند، فیلدهای مشابه در رکوردهای مختلف داشته باشند، در اینجا این کار ممکن نیست و نمی‌توان از یک اسم برای چند متغیر داده شمارشی استفاده کرد.

متغیر داده شمارشی شبیه مجموعه‌ها حداکثر دارای ۲۵۶ عضو می‌تواند باشد که از صفر تا ۲۵۵ شماره‌گذاری می‌شود و لذا یک بایت حافظه را اشغال می‌کند.

تمام موارد فوق در مثال زیر آمده است:

```
Type
      Cars_type = (Peykan, Pegout, Samand, Pride, Patrol, PK);
      Iran_khodro = Peykan..Samand;
      Saipa = Pride..Patrol;
      Pars_khodro = PK..Patrol;          خطای کامپایلری وجود دارد
Var
      car: cars_type;
      Ik: Iran_khodro;
      Sp: Saipa;


Begin
```



```
      Read(sp);  ------>  خطای کامپایلری به علت خواندن Sp
      Writeln(IK);  ------>  خطای کامپایلری به علت نوشتن
      Sp:= praid;  ------>  درست می‌باشد
      IK:= samand;  ------>  درست می‌باشد
      Write(ord (Sp) + ord(IK));  ------>  3+2=5
End.
```

۱-۲-۱۲- عملیات روی داده‌های شمارشی

داده‌های شمارشی همانند هر نوع تایپی می‌توانند در type برنامه تعریف شود و به عنوان آرگومانهای روالها و یا خروجی توابع می‌تواند درنظر گرفته شود ولی حتماً باید در type تعریف شده باشد و مستقیماً نمی‌توان بکار برد. این عمل مشابه مجموعه‌ها و آرایه‌ها نیز می‌باشند، لذا ابتدا در type تعریف می‌شود، سپس به عنوان ورودی و یا خروجی روالها استفاده می‌شود وگرنه خطای کامپایلری پیش خواهد آمد. همچنین می‌توان در آرایه‌ها از داده‌ها شمارشی استفاده کرد، زیرا مثال اندیس‌های آرایه داده‌های شمارشی هستند.

مثال ۴-۱۲

```
Type
      days_type = (sat,sun,mon,tue,wed,thu,fri);
Var
      day,odd,even: days_type;

      A: array[1..10] of days_type;  ------>  آرایه ۱۰ سلول دارد
      B: array[sun..thu] of char;  ------>  آرایه ۵ سلول دارد
      C: array[daystype] of daystype;  ------>  آرایه ۷ سلول دارد
      D: array[sat..fri]of integer;  ------>  آرایه ۷ سلول دارد
function F1 (A,B:days_type): days_type;
procedure F2 (var c:days_type);
procedure F3 (odd: (sun,tue,thu));------> خطا، زیرا مستقیماً نمی‌تواند تعریف کرد

Begin
      odd:= sun;
      even:= sat;
      A[2]:= wed;
      A[3]:= fri;
      C[sat]:= mon;
      B[mon]:= 'A';
```

```
B[sat]:= 'B';                                          خطای آرایه
D[wed]:=10;
D['sat']:= 6;                                          خطای آرایه
B[wed]:=7;
Writeln (ord(odd) + ord(A[2]) + D[wed]);   ⟶  ۱+۴+۱۰=۱۵
Writeln (B[mon] + C[sat]);                 ⟶  خطای جمع
End.
```

مثال ۵-۱۲ مشابه مجموعه‌ها نمی‌توان داده‌های شمارشی را مستقیماً Read یا Write کرد. برنامه‌ای بنویسید که اعداد صفر تا شش را به مراتب دریافت کرده و به کمک داده‌های شمارشی روزهای متناظر با آنها را از شنبه تا جمعه را حساب کند.

```
Program test:
Type
      days_type = (sat,sun,mon,tue,wed,thu,fri);
Var
      day: days_type;
      i: byte;
Begin
      Write('Enter your days number: ' );
      Readln(i);
      While (i>=0) and (i<=6) do begin
            Case I of
                  0: day:= sat;
                  1: day:= sun;
                  2: day:= mon;
                  3: day:= tue;
                  4: day:= wed;
                  5: day:= thu;
                  6:day:= fri;
               end;
            case day of
                  sat: writeln ('your day is sat');
                  sun: writeln ('your day is sun');
                  mon:writlen ('your day is mon');
                  tue:wrteln (' your day is tue');
                  wed: writeln ('your day is wed');
                  thu: writeln ('your day is thu');
                  fri: writeln ('your day is fri');
            end;
            Readln(i);
      end; {while}
End.
```


۳-۱۲- تمرینات

۱- با توجه به مجموعه‌های تعریف شده در مورد درستی و نادرستی هر عبارت زیر بحث کنید.

آرایه‌های تعریف شده هر یک چند تا سنول دارند؟

```
Type
      itype = set of 1..100;
      btype = set of byte;
      ctype = set of char;
      diype = set of integer;
Var
      i: itype;
      b: btype;
      c: ctype;
      d: array[boolean] of itype;
      e: array[bytpe] of btype;

Function F1 (x:btype):char;
Function F2(y:btype; var z:ctype):ctype;
Procedure P1(x: set of 1..10);
Procedure P2(x:btype; var y:ctype);
Begin
      i:=[0,1,2,3];
      b:=[1,2,3,4];
      c:=['a','b','c',A,B,C];
      e:=['A'..'Z','0'..'9','a'..'z'];
      i:=[2,5..10];
      B:=[2,3,4,5,6,7,8,9];
      Writelr(i <= b);
      Writeln( b = i);
      Writeln(i+b*I = b-i);
      Writeln(b+i+b-i <= b+i-b-i);
      Writeln(5 in i);
      Writeln(60 in b);
      Writeln('A' in e);
End.
```

راهنمایی: در مثال فوق توجه داشته باشید که اولویت عملگرهای مجموعه به قرار زیر است:

الف: * اشتراک

ب: + و - اجتماع و تفاضل

ج: in , = , <= , >= , <>

۲- برنامه‌ای بنویسید که مجموعه دلخواهی از کاراکترها را تهی کند.

۳- با توجه به داده‌های شمارشی تعریف شده در زیر، در مورد درستی یا نادرستی هر عبارت بحث کنید. آرایه‌های تعریف شده، هر یک چند تا سلول دارند؟

```
Type
        colors_type= (green,blue,red,white,black);
        days_type= (sat,sun,mon,tue,wed,thu,fri);
        numbers = (0..300, 400..500);
Var
        day, odd, even: days_type;
        color: colors_type;
        A, B: Array[1..3] of colors_type;
        C: Array[blue..black] of char;
        D: Array[days_type] of colors_type;
Function F1(A: days_type; var B: colors_type): char;
Funetion F2(var C: (green,blue,red)): colors_type;
Procedure P1(var D: colors_type);

Begin
        day:= sat;
        odd:= sun;
        even:= sat;
        Writeln(day+odd+wed);
        Writeln(ord (day) + ord (odd));
        Wrteln(succ(fri) + ord(mon));
        A[1]:= red;
        B[2]:= green;
        Writeln(ord(A[1]) + ord(pred(B[2])));
End.
```


۱۲-۴- تمرینات برنامه‌نویسی

۱- برنامه‌ای بنویسید تا مجموعه ای از کاراکترهای کوچک را گرفته و به یک مجموعه از کاراکترهای حروف بزرگ متناظر با آن تبدیل کند.

۲- به کمک داده‌های شمارشی، برنامه ای بنویسید که نام دانشجویان یک کلاس را در برگیرد و هر دانشجو نیز یک شماره داشته باشد. سپس با دریافت نام او شماره او در نمایشگر چاپ شود.

۳- برنامه‌ای بنویسید که مجموعه دلخواهی از اعداد را دریافت کرده و مجموع داده‌های داخل آن را بدست آورید بطوریکه:

الف: مجموعه اصلی از بین نرود.

ب: مجموعه اصلی از بین برود و تهی شود.

فصل ۱۳

رکوردها (Records)

هدفهای کلی

- مفهوم رکورد و اجزای آن
- انواع رکوردها و استفاده از آن در برنامه
- معرفی مزایای رکوردها در برنامه

هدفهای رفتاری

دانشجو پس از مطالعه این فصل باید بتواند:

- مواقع لزوم از رکورد را تشخیص دهد.
- از رکوردها در برنامه‌اش استفاده کند.
- برنامه‌های بزرگ و با داده‌های زیاد بنویسد.

مقدمه

نوع داده ساخت‌یافته‌ای که در اینجا مطرح می‌شود، رکورد نام دارد که جهت نگهداری داده‌های مختلف نظیر نام، نام خانوادگی، سن و آدرس برای یک دانشجو بکار می‌رود. رکوردها برخلاف آرایه‌ها که دارای عناصر از یک جنس و نوع هستند، دارای عناصر از انواع مختلف می‌باشند. اطلاعات ازنوع مختلف را نمی‌توان در آرایه نگهداری کرد چرا که دارای جنس مشابه نیستند و لذا از ساختاری به نام رکورد استفاده می‌شود.

رکوردها موارد استفاده بسیار دارند. مثلاً اگر شما بخواهید اطلاعات دانشجویان را ذخیره کنید، برای آنها برنامه بنویسید مجبور خواهید بود، رکوردی از دانشجویان بسازید و اطلاعات دانشجویان را در آرایه‌ای از رکوردهای دانشجویان نگهداری کنید زیرا هر دانشجو مشخصاتش با دانشجویان دیگر فرقی نمی‌کند ولی مشخصات فردی‌اش با هم فرق می‌کند. یعنی نوع اسم او با سن او فرق دارد ولی تمام دانشجویان دارای اسم و سن و... هستند.

۱-۱۳- تعریف رکوردها

در اینجا با مثال بالا کار را آغاز می‌کنیم. درنظر بگیرید می‌خواهیم رکوردی از دانشجویان فراهم کنیم. هر دانشجو مشخصات فردی نظیر اسم و فامیلی دارد که می‌تواند از نوع String باشد. همچنین سن او می‌تواند از نوع integer باشد. آدرس خانه هر دانشجو از نوع String باشد و...

برای ساختن رکورد، از کلمه کلیدی Record استفاده می‌شود که مطابق زیرمی‌باشد:

```
Type Student = Record
        Name: String [10];
        Family: String [15];
        Age: integer;
        Address: String;
End;
```

Record نیاز به begin ندارد ولی حتماً end; برای اتمام آن مورد نیاز است. در این تعریف Student از نوع Record است که با تساوی در type آمده است. سپس هر کدام از اجزاء رکورد که فیلد نامیده می‌شوند، همانند تعریف متغیرها می‌آید. در پایین تعریف رکورد یک end; لازم است که اتمام تعریف فیلدها را نشان می‌دهد. حال می‌توان در

var برنامه تعریف کرد:

```
Var
        S: Student;
```

چون Student یک type می‌باشد و نظیر دیگر typeها و یا نوع‌های استاندارد پاسکال است، می‌توان یک یا چند و یا حتی آرایه ای از آن تعریف کرد. به عبارتی می‌توان هر عملیاتی نظیر بقیه typeها روی آن انجام داد. در بالا S یک متغیر از نوع Student تعریف می‌شودکه S دارای چهارفیلد Name، Family، Age و Address می‌باشد.

همچنین نوع فیلد داده یعنی نوع Name و... در بالا می‌تواند ساده یعنی نظیر integer، real و... و یا پیچیده یعنی آرایه و یا رکورد باشد. در واقع می‌تواند هر نوع type باشد.

می‌توان Recordها را مشابه بقیه نوع‌های داده‌ای مستقیماً در Var تعریف نمود. یعنی مشابه زیر:

```
Var S: Record
        Name: String [10];
        Family: String [15];
        Age: integer;
        Address: String;
End;
```

در اینجا مستقیماً متغیر S تعریف می‌شود و دیگر typeی وجود ندارد. S یک متغیر است که از نوع Record است و چهار تا فیلد دارد. در Var برای تعریف متغیر از ':' استفاده می‌شود. این نوع تعریف سفارش نمی‌شود زیرا اگر در جایی از برنامه به رکورد نیاز داشته باشیم، دوباره باید رکورد تعریف شود. کلاً توصیه می‌شود تمامی‌تعاریف با نامهای خوب و آشنا در type تعریف شده و سپس متغیرهایی از آنها درنظر گرفته شود.

بطور کلی تعریف نوع رکورد در زیر آمده است:

```
Type
    Rec-type = Record
        Field1-list = type1;
        Field2-list = type2;
        ..
        ..
        Fieldn-list = typen;
End;
```

توضیح اینکه در بالا یک رکورد در حالت خیلی کلی در type با نام rec-type تعریف شده است که دارای n فیلد field1-list تا fieldn-list می‌باشد که هرکدام از fieldi-lits نیز می‌توان لیستی از نوع‌های مشابه باشد که با ',' جدا شده اند. Typei می‌تواند هرنوع type از پیش تعریف شده استاندارد و یا type تعریف شده برنامه‌نویس باشد. در زیر مثالی از یک رکورد آمده است:

```
Type list = array [1... 10] of Boolean;
Numbers = Record
        X, y, z: Real;
        P, q: integer;
        A, B, C, D: Array [1... 10] of Char;
        F, T: Boolean;
        K: list;
End;
Var
     R1, R2: Numbers;
 L1, L2, L3: list;
```

همانطور که می‌بینید رکورد Numbers دارای ۵ سری فیلد می‌باشد و هر فیلد آن یک list از متغیرهاست. مثلاً فیلد اول شامل سه متغیر x, y, z است و متغیر k شامل یک آرایه است از list که ده عنصر از نوع Boolean است.

۱-۱-۱۳- دسترسی به فیلدهای رکورد

برای دسترسی به فیلدهای رکورد ازعلامت '.' استفاده می‌شود. یعنی بصورت‌زیر:

نام فیلد. نام متغیر رکورد

بعد از تعریف رکورد، در داخل برنامه می‌توانید با علامت ' .' مطابق فوق به فیلدها دسترسی پیدا کرده و عملیات لازمه را روی آنها انجام داد.

مثال ۱-۱۳ برنامه‌ای بنویسید که رکوردی از نوع اعداد ایجاد کرده و مقادیر آنرا مقدارهی کند.


```
Type
        Numbers = record
                a, b, c = integer;
                x, y, z = Real;
        end;
Var
        Num = Numbers;
begin
        Num. a: = 2;
        Num. b: = 3;
        Num. x: = 2.5;
        Num. y: = 3.5;
        Num. z: = 10;
        Write (Num.a + Num.b + Num.x + Num.y + Num.z);
End.
```

در اینجا رکوردی از اعداد صحیح و حقیقی ایجاد شده است و سپس مقادیری به آنها داده است. این عمل توسط علامت ' .' صورت گرفته است که به فیلدها دسترسی پیدا کرده ایم و حالا دیگر هر فیلد نظیر یک متغیر معمولی است و هر کاری می‌توان با آن انجام داد.

مثال ۲-۱۳ برنامه‌ای بنویسید که رکورد یک دانشجو را داشته باشد و با توجه به جنس او کلمه آقا و یا خانم را به همراه نام و نام خانوادگی‌اش را چاپ کند.

```
Type
   Student = record
        Id: integer;
        Name: string[10];
        Family: string[15];
        Sex: char;
        Age: integer;
   End;
Var
   Stu: student;
Begin
   Stu.id:= 80132;
   Stu.name:='Ali';
   Stu.family:='Ahmadi';
   Stu.sex:='M';
   Stu.sge:=18;
  Case sex of
        'M': write('Mr.');
        'F': write('Mrs.');
  end;
  write(stu.name, ' ', stu.family);
End.
```

در برنامه فوق فیلد Sex از نوع کاراکتر بوده و می‌تواند مقدار F یعنی زن و یا M

یعنی مرد باشد. چون M است لذا ابتدا Mr چاپ شده و سپس نام و فامیلی دانشجو یعنی Mr Ali Ahmadi چاپ می‌شود.

۲-۱-۱۳- بدست آوردن حجم یک رکورد

برای بدست آوردن فضای اشغال شده توسط رکورد ابتدا باید فضای اشغال شده توسط تمامی‌فیلدها را بدست آورده و سپس باهم جمع کنیم. مثال زیر را درنظر بگیرید:

مثال ۳-۱۳

```
Type
        List1 = Array [1... 5] of integer;
        List2 = Array [1... 5] of char;
        Rectype = Record
        A, B: Real;              →        ۲*۶
        C, D: String [10];       →        ۱+۱۰
        F: Array [1... 10] of Boolean;  →  ۱۰*۱
        G: list1;                →        ۵*۲
End;
Var
        x: Rectype;
```

با تعریف‌های فوق در type هیچ فضایی اشغال نمی‌شود ولی وقتی X در Var تعریف می‌شود، رکورد فوق ساخته می‌شود و همانطور که دیده می‌شود حافظه در نظر گرفته شده، که معادل ۴۳=۱۰+۱۰+۱۱+۱۲ بایت فضا اشغال می‌شود.

توجه شود که فیلدهای یک رکورد نمی‌توانند از همانگونه رکورد تعریف شود. رکورد زیر را در نظر بگیرید.

```
Type
    Rec = record
        C: Char;
        x: Real;
        A: Rec;
        B: integer;
End;
```

در بالا همه اعلانها درست است فقط در A: Rec خطای کامپایلری وجود دارد زیرا Rec همان رکوردی است که A متعلق به آن می‌باشد و لذا خطا وجود دارد.

به کمک دستور و کلمه کلیدی with-do می‌توان از نام رکوردها در دسترسی به

فیلدهای آن صرفنظر کرد. به مورد زیر توجه کنید.

مثال ۴-۱۳

```
Type
        Rec = record
            i, j: integer;
            c: char;
        End;
Var
        X: Rec;
Begin
    With x do
        Begin
          i: = 3;
          j: = 2;
          c: = 'A';
        end;
End.
```

در اینجا با with-do که مخصوص رکوردها می‌باشد، عملیات فاکتورگیری از نام رکورد انجام شده است و لذا فرقی با حالات زیر نمی‌کند:

```
x. i: = 3;
x. j: = 2;
x. c: = 'A';
```

۲-۱۳- رکوردهای تودرتو

همانطور که قبلاً نیز گفته شد، فیلدهای یک رکورد می‌توانند از هر نوعی باشد، از جمله می‌توانند از نوع رکورد دیگری باشند. در اینجا نیز مشابه قبل دسترسی به همان صورت می‌باشد فقط به تعداد رکوردهای تودرتو، '.' پیش می‌آید. به مثال زیرتوجه‌کنید.

مثال ۵-۱۳

```
Type
        Rec = type
        a, b: integer;
        c: char;
        x: Record
           p: integer;
           q: integer;
        End;
End;
Var
        r: Rec;
```

دراینجا یک رکورد تو در تو به نام Rec تعریف شده است که متغیر r از آن نوع تعریف شده است. سپس برای دسترسی به فیلدهای a , b ,c می‌توان به صورت زیر عمل کرد:

```
r.a
r.b
r.c
```

ولی برای دسترسی به فیلدهای p ,q چون متعلق به رکورد x نیز هستند داریم:

```
r.x.p
r.x.q
```

اگر بخواهیم از دستور with-do استفاده کنیم:

```
With r do
  With x do
  Begin
     a:=2;
     b:=3;
     c:='c';
     p:=4;
     q:=5;
  End;
```

و یا بصورت دیگر زیر داریم:

```
With r, x do
  Begin
     a:=2;
     b:=3;
     c:='c';
     p:=4;
     q:=5;
  End;
```

همچنین فضای اشغال شده برای چنین رکوردهایی، از مجموع همه فیلدها بدست می‌آید که برابر ۹=۲+۲+۱+۲×۲ بایت می‌باشد.

مثال ۶-۱۳ رکورد دانشجوی قبل را کاملتر می‌کنیم. در نظر بگیرید که هر دانشجو نام، نام‌خانوادگی، شماره دانشجویی و سن به صورت روز، ماه و سال دارد.


```
Type
   Rec_age = record
         Day: 1..31;
         Month: 1..12;
         Year: interegr;
   End;
   Student = record
         Name: string[10];
         Family: string[15];
         Id: integer;
         Age: Rec_age;
   End;
Var
   Stu: student;
```

در اینجا یک رکورد تودرتو برای یک دانشجو ساخته شده است که اگر بخواهیم مقداردهی در داخل برنامه داشته باشیم، خواهیم داشت:

```
Stu. Name: = ' Ali ';
Stu. Family: = ' Ahmadi ';
Stu. ID: = 81243;
Stu. age. Day: =3;
Stu. age. Month: = 6;
Stu. age. Year: = 61;
```

۳-۱۳- آرایه‌ای از رکوردها

هنگامی‌که ما تعدادی داده مشابه داریم ولی در هریک، داده‌های مختلفی وجود دارد می‌توانیم یک رکورد تعریف کرده، سپس آرایه‌ای از آن تعریف کنیم. مثلاً در یک دانشگاه تعدادی دانشجو با خصوصیات مشابه وجود دارد، ولی هر دانشجو دارای مشخصات مختلفی نظیر رکورد قبلی می‌باشد.

```
Type
   Student = record
         Name: string[10];
         Id: integer;
         Age: integer;
   End;
   Arr_stu: array[1..10] of student;
Var
   S: Arr_stu;
```

در بالا ابتدا یک رکورد دانشجو تعریف شده است، سپس به تعداد ۱۰ نفر دانشجو تحت آرایه Arr_stu شکل گرفته است و سرانجام متغیری به نام S از نوع آن

تعریف شده است. مشابه ساختار آرایه، چیزی عوض نشده است و فقط هر عنصر آرایه، یک رکورد می‌باشد که دارای سه فیلد مطابق جدول فوق می‌باشد. تعریف فوق مدلی برای طراحی جدول است. برای دسترسی به جدول فوق داریم:

```
S[1].name
S[1].id
S[1].age
S[2].name
S[2].id
..
..
..
```

و در حالت کلی می‌توان با یک حلقه for به تمام خانه‌های جدول دسترسی پیدا کرد:

```
For i:=1 to 10 do
  With S[i] do
   Begin
        Name:= …;
        Age:= …;
        Id:= …;
   End;
```

مثال ۷-۱۳ یک برنامه کامل بنویسید که جدولی از کتابها (۴ کتاب) تهیه کرده و عناوین آنها را دریافت کرده و سپس در خروجی چاپ کند.

```
Program test;
Const
        No=4;
Type
        Book_type=record
                Name:string;
                Id: integer;
        End;
        Book_arr: array[1..no] of Book_type;
Var
        Book: Book_arr;
        i: integer;
Begin
   For i:=1 to no do
      With Book[i] do
            Readln(Name, id);
   For i:=1 to no do
       Writeln(i, ':', Book[i].name, ' ',Book[i].id);
End.
```


در این برنامه ابتدا مقدار ثابت ۴ کتاب در no تعریف شده است. سپس یک کتاب بصورت یک رکورد با دو فیلد ID و Name درنظر گرفته شده است. به علت وجود ۴ کتاب همانند، آرایه‌ای از رکورد فوق تعریف شد. در ابتدای برنامه ابتدا به هر چهار کتاب توسط حلقه for مقدار اولیه داده شد و سپس هر چهار کتاب در خروجی چاپ می‌شود. طریقه استفاده از آرایه و رکورد به همان سادگی فصلهای قبل می‌باشد و فقط تعاریف کلی‌تر می‌شود ولی نوع استفاده عوض نمی‌شود. در رکورد فوق می‌توان از رکوردهای تو در تو نیز استفاده کرد و در واقع چیزی عوض نمی‌شود بلکه تنها نحوه دسترسی به فیلدها فقط عوض می‌شود.

۳-۱۳- ارسال رکورد به زیربرنامه‌ها

رکوردها را می‌توان مشابه انواع تعاریف ساده دیگر به صورت پارامترهای متغیر و مقدار به زیربرنامه ارسال کرد. ولی نوع برگشتی توابع نمی‌تواند از نوع رکورد باشد، یعنی حتماً باید از نوع ساده نظیر integer، char و... باشد و از انواع ترکیبی نظیر رکورد، آرایه، مجموعه و فایل نمی‌تواند باشد.

اگر بخواهیم رکوردی را بصورت پارامتر به زیربرنامه ارسال کنیم ابتدا باید آنرا در type تعریف کرده و سپس ارسال شود وگرنه کامپایلر خطا صادر می‌کند. مثال زیر را درنظر بگیرید.

مثال ۸-۱۳ می‌خواهیم برنامه‌ای بنویسیم که رکورد stu را برای ۳۰ دانشجو ایجاد کرده، به کمک زیربرنامه مقداردهی شده و چاپ شوند.

```
Program test;
Const    no   30;
Type
        Stu-rec = Record
                Name: string [10];
                ID: integer;
                Age: record
                    Day: integer;
                    Month: integer;
                    Year: integer;
                End;
        End;

        Stu_arr=array[1..no] of stu_rec;
Var
        Stu: stu_rec;
Procedure ReadStu(Var S: Stu_rec);
Var
        i: integer;
Begin
  for i:= 1 to no do
        Readln(S[i].name, S[i].id, S[i].age.day, S[i].age.month, S[i].age.year);
End;
Procedure WriteStu(S: Stu_rec);
Var
        i: integer;
Begin
  for i:= 1 to no do
        Writeln(S[i].name,' ', S[i].id, ' ', S[i].age.day,' ', S[i].age.month,' ',
        S[i].age.year);
End;
Begin
  ReadStu(Stu);
  WriteStu(Stu);
End.
```

در این برنامه آرایه‌ای از رکورد دانشجو ساخته شده است. سپس توسط دو روال Readstu و Writestu به ترتیب دانشجویان خوانده و نوشته می‌شوند. ولی نکته در اینجاست که پارامتر Readstu بدلیل تغییرات ایجاد شونده در پارامتر ورودی از نوع Var است ولی درWritestu چون فقط نوشته خواهندشد وتغییرنمی‌یابد، ازنوع مقدارمی‌باشد.

مثال ۹-۱۳ برنامه‌ای به کمک زیربرنامه‌ها بنویسید که شماره دانشجویی، نام و نمره درسی n دانشجویان یک کلاس را داشته باشد و سرانجام معدل کلاس را بدست آورد.


```
Program test;
Const
        no = 100;
Type
        Stu-rec = Record
                Name: string [10];
                ID: integer;
                Mark: integer;
        End;
        Stu-Arr = Array [1... n] of stu-Rec;
Var
        Stu: stu-Arr;
        Count: integer;
        A: Real;
Procedure ReadStu(var S:Stu_arr);
Var
        i: integer;
Begin
   i:=1;
   repeat
        write('Enter Student Names, ID and Mark: ');
        readln(S[i].name, S[i].id, S[i].mark);
        i:=i+1;
   until (S[i].id =0) or (i>n);
   count:= i-1;
End;
Function WriteStu(S:Stu_Arr):real;
Var
        i: integer;
        G: real;
Begin
   G:= 0;
    For i:=1 to count do begin
        Writeln(S[i].id,' ', S[i].mark);
        G:= G+ S[i].mark;
     End;
     WriteStu:= G / count;
End;
Begin
   ReadStu(Stu);
   A:=WriteStu(Stu);
   Writeln('The Average Mark of Students is: ', A);
End.
```

در این مثال تعداد دانشجویان معلوم نیست ولی حداکثر ۱۰۰ نفر درنظر گرفته شده است. سپس از تعریف آرایه‌ای از ۱۰۰ دانشجو، دو زیربرنامه نوشته شده است که اولی روالی است که مشخصات دانشجویان را می‌خواند و در آرایه ذخیره می‌کند. و چون آرایه تغییر می‌کند لذا از نوع Var می‌باشد. تابع Writestu تغییری در آرایه ایجاد

نمی‌کند لذا آرایه را به صورت مقدار می‌گیرد. در تابع Readstu هرگاه تعداد دانشجویان یعنی شمارنده حلقه Repeat از ۱۰۰ تا بیشتر شود و یا ID یک دانشجو صفر وارد شود حلقه خاتمه می‌پذیرد و تعداد دقیق دانشجویان کلاس در متغیر سراسری Count ذخیره می‌شود. در تابع Writestu بعد از نوشتن دانشجویان با ID و Mark متناظر، میانگین نیز محاسبه شده و در قسمت اصلی برنامه در انتها چاپ می‌شود. توجه کنید که Count و A متغیرهای Global هستند و در داخل روالها مقدار آنها تغییر می‌یابد.

۱۳-۴- تمرینات

۱ برای گونه‌های حیوانات یک پارک نظیر پرندگان وحشی رکوردی شامل نام، دسته، محیط زندگی، سن بسازید. در نظر بگیرید که در پارک ۱۰۰ گونه از یک پرنده وحشی وجود دارد (راهنمایی: آرایه‌ای از رکورد با فیلدهای گفته شده بسازید).

۲- هریک از رکوردهای زیر چقدر فضا اشغال می‌کند:

```
Type
        Rec1 = Record
                x, y: integer;
                s1, s2: string[10];
                a, b, c: 1..30;
        end;

        Rec2 = Record
                A, B: Array [1..5] of boolean;
                C: Record
                    x, y: Word;
                    z: Real;
                end;
        end;

Var
        x: Rec1;
        y: Rec2;
```

۳- در تعاریف زیر اشکالاتی وجود دارد و یا صحیح هستند. آنرا مشخص کنید.

الف)

```
Var
        t: array [1... 10] of record;
                a: array [1...10] of Char;
                b: word;
        end;
```

ب)

```
Type
        r = Record
                a: Array [1... 10] of r ;
                b: r ;
        end;
```

ج)

```
Type
        r: Record
                r: word;
                x: char;
```

د)

```
end;
```

ه)

```
Type
    r = Record
            x: char;
            y: word,
        end;
    end;

var
    A: Array[1..10] of record
            x: char;
            y: A;
            z: record
                    x:A;
                    y:Z;
            end;
    end;
```

خ)

```
var
    rec = Record
            A,B: char;
            x,y,z: real;
            p, q: Array[1..n] of record
                    m,n: integer;
            end;
            k,l: array[1..10] of Rec;
            u,v,w: Boolean;
    end;
```


۱۳-۵- تمرینات برنامه‌نویسی

۱- برنامه‌ای بنویسید که برای نگهداری تاریخ به صورت روز، ماه و سال برای ۱۰۰ سال بکار رود. در واقع به صورت تقویم باشد و جمله‌ای در مورد آن تاریخ را نگهداری کنید؟ (راهنمایی: آرایه‌ای ۳۶۶ تایی از روز بهمراه آرایه‌ای از ۱۲ ماه و یک سال و یک رشته را به صورت یک رکورد در نظر گرفته سپس برای ۱۰۰ سال بصورت یک آرایه صدتایی در نظر بگیرید.)

۲- برنامه‌ای بنویسید که تعداد نامشخصی از مشخصات دانشجویان (شامل شماره، نام، نام‌خانوادگی، سن و سه نمره درسی) را از ورودی دریافت کرده و سپس منویی بسازد که دارای انتخابهای زیر باشد:

الف) اضافه کردن یک دانشجو

ب) حذف یک دانشجو

ج) یافتن میانگین نمرات برای هر دانشجو

د) یافتن میانگین نمرات برای کل دانشجویان

ه) مرتب‌کردن دانشجویان بر حسب نام‌خانوادگی

۳- برنامه‌ای بنویسید که جهت نگهداری مشخصات کتب موجود در یک کتابخانه بکار رود. کلیه عملیات موجود در کتابخانه مورد نظر می باشد.

فصل ۱۴

فـــایـلها (Files)

هدفهای کلی

- مفهوم فایل و انواع آن
- موارد استفاده از فایلها

هدفهای رفتاری

دانشجو پس از مطالعه این فصل باید بتواند:

- در برنامه‌اش از فایلها استفاده کند.
- برای برنامه‌های با داده‌های زیاد از فایل استفاده کند.
- با فایلهای داده‌ای و بایزی برنامه‌نویسی کند.

مقدمه

برای ذخیره دائمی داده‌ها از ساختاری به نام فایل (File) استفاده می‌کنیم. تاکنون همه عملیات لازم در حافظه اصلی انجام می‌گرفت که گذرا و فقط به زمان اجرای برنامه و روشن بودن کامپیوتر بستگی داشت. ولی در فایلها چنین نیست، بلکه داده‌ها در فایلهایی قرار دارند که حتی بعد از خاموش کردن کامپیوتر، بعدها قابل دسترسی است.

درنظر بگیرید شما در حال اجرای برنامه‌ای هستید که ۱۰۰ عدد بصورت ورودی دریافت می‌کند و ۲۰۰ عدد در خروجی تولید می‌کند. حال با اجرای هربار برنامه، باید تعداد ۱۰۰ عدد بصورت داده‌های ورودی به برنامه، وارد کنید تا در خروجی ۲۰۰ عدد ببینید که بسیار خسته‌کننده است. لذا بهتر است که این داده‌ها یکبار در فایلی ذخیره شوند و فایل به صورت ورودی داده شود تا فایل خروجی نیز تولید شود و همچنین داده‌ها ماندگار شوند و از بین نروند.

در زبان پاسکال فایلها به ۳ طریق تعریف می‌شوند. این طبقه بندی به نوع فایلها از نظر ساختار داخلی می‌باشد که در زیر آمده است:

۱-۱۴- فایلهای متنی (Text)

یک فایل متنی از تعداد کاراکتر تشکیل شده است که با یک اسم در روی دیسک ذخیره شده است. چون فایل از نوع متنی است می‌توانید داده‌های داخل آنرا مشاهده کنید.

می‌توانید کلیه داده‌هایی که بعنوان ورودی یک برنامه لازم است و یا در خروجی تولید می‌شود در فایلهای متنی جداگانه ای ذخیره کنید. همانطور که گفته شد، محتوی فایلهای متنی قابل مشاهده می‌باشد که شما می‌توانید با ویرایشگرهای مختلف آنرا مشاهده کنید.

ساختار داده‌ها در یک فایل متنی بدین صورت است که تعدادی خط وجود دارد که به علامت EOLN ختم می‌شوند و در انتهای فایل نیز علامت EOF قرار دارد. طول خطوط نامشخص است و اندازه فایل ممکن است بسیار بزرگ یا کوچک و یا تهی باشد.

۱-۱-۱۴- طریقه خواندن اطلاعات از یک فایل متنی

در نظر بگیرید که یک فایل متنی به اسم ' Test.dat ' به صورت زیر با ویرایشگری ایجاد شده است:

```
Money is for:

10      12      14
2       3       'Ali'
```

با استفاده از تایپ text می‌توان یک فایل متنی تعریف کرد.

```
Var
        f: Text;
```

با این تعریف، f یک متغیر از نوع فایلهای متنی می‌باشد. حال در برنامه با دستورات Read و Readln و Write و Writeln می‌توانیم همانند قبل عملیات خواندن و نوشتن روی این فایلها انجام می‌گیرد. Readln بعد از خواندن داده‌های مورد نظر به سطر بعد می‌رود. این عمل برای Writeln نیز چنین می‌باشد. ولی قبل از استفاده از این دستورات باید فایل متنی برای خواندن یا نوشتن باز شود. دستور Reset فایل متنی را برای خواندن باز می‌کند و مکان نما را ابتدای فایل می‌برد. دستور Rewrite فایل متنی را برای نوشتن در آن باز کرده، و مکان نما را به ابتدای آن می‌برد. حال باتوجه به فایل متنی 'test.dat' می‌خواهیم اعداد داخل فایل را که پنج تا هستند باهم جمع کرده و بگوییم شخص مورد نظر (علی) چقدر پول دارد:

```
Program usefile;
Var
        f: Text;
        str1, str2: string;
        a, b, c, d, e: integer;
begin
        Assign (f, ' Test. dat ');
        Reset (f);
        Readln (f, str1);
        Readln (f, a, b, c);
        Readln (f, d, e, str2);
        Write (str2, ': ' , a+b+c+d+e);
        Close (f);
End.
```

خروجی این برنامه بصورت Money is for: Ali 41 می‌باشد که مجموع اعداد داخل فایل می‌باشد.

دستور Assign همانطور که می‌بینید، اسم فایل را که یک رشته می‌باشد به متغیر f شناسایی می‌کند و f در واقع فقط به ' test. dat ' رجوع می‌کند. در واقع این دستور یک رشته اسمی که نام فایل متنی است را به متغیر فایل text نسبت می‌دهد. سپس باتوجه به نوع مسئله، فایل f برای خواندن توسط Reset باز می‌شود. رشته موجود خط اول خوانده می‌شود و در str1 ذخیره می‌شود. اعداد a, b, c از فایل f توسط Readln خوانده شده و مکان نما به خط بعد می‌رود. همینطور اعداد e, d و رشته 'Ali' از فایل f خوانده شده و برنامه خروجی مجموع اعداد را به همراه دو رشته روی مونیتور چاپ می‌کند. در انتها فایل باز شده، توسط close بسته می‌شود.

در مثال بالا، اینطور برآمد که برنامه‌نویس باید به طور دقیق از ساختار داخل فایل متنی خبر داشته باشد، چون باید بداند در کجا چه بخواند و یا بنویسد.

تمام قواعد مربوط به توابع Read و Write و Readln و Writeln در مورد چگونه نوشتن اعداد صحیح، حقیقی و... در اینجا نیز صادق است. به موارد زیر توجه کنید:

```
Read (f, a);
Readln (f, a, b);
Write (f, a: 2);
Writeln (f, a:3, b:3:2, s:4:2, str:10);
```

که f متغیر فایل در کلیه موارد بوده و a , b در دو مورد اول صحیح هستند و در سومی می‌تواند صحیح یا حقیقی باشد. در چهارمی a , b , c حقیقی بوده و str از نوع string می‌باشد.

۲-۱-۱۴- مثالها

در زیر مثالهایی آمده است که در هرکدام بسادگی موارد استفاده توابع مربوط به فایلهای متنی در آنها آمده است.

مثال ۱-۱۴ برنامه‌ای بنویسید که از ورودی ۱۰ عدد دریافت کند و آنها را به ترتیب هرکدام دریک خط ازیک فایل خروجی بریزد و در انتها تعداد و مجموع آنها را بنویسد.


```
Var
        f: Text;
        A: Array [ 1.. 10] of integer;
        Sum, i: integer;
Begin
        Sum: = 0;
        Writeln (' Enter 10 numbers: ');
        For i: = 1 to 10 do begin
                Read (A [i]);
                Sum: = sum + A[i];
        end;
        Assign (f, 'out.dat');
        Rewrite (f);
        For i: = 1 to 10 do
                Writeln (f, A[i]);
        Writeln (f, Sum);
        Close (f);
End.
```

در این مثال ابتدا ۱۰ عدد از ورودی دریافت شده و در آرایه ده تایی از A قرار می‌گیرد و همزمان جمع آنها در Sum می‌رود. f به نام ' out.dat ' توسط Assign نسبت داده شده و سپس توسط Rewrite روی حافظه جانبی ایجاد می‌گردد و مکان‌نما به ابتدای آن جهت نوشتن می‌رود. سپس اعداد در آن نوشته شده و در انتهای فایل مجموع آنها نوشته می‌شود. تابع Close در انتهای برنامه لازم است چون داده‌ها در بافر قرار دارند و ممکن است با Close نکردن، فایل تشکیل نشود و داده‌ها در آن نوشته نشوند. این برنامه به طریقه دیگر در زیر آمده است:

```
Var
        f: Text;
        A: Array [ 1.. 10] of integer;
        Sum, i: integer;
Begin
        Sum: = 0;
        Writeln (' Enter 10 numbers: ');
        Assign (f, ' out.dat ');
        Rewrite (f);
        For i: = 1 to 10 do begin
                Read (A [i]);
                Sum: = sum + A[i];
                Writeln (f, A[i]);
        end;
        Writeln (f, Sum);
        Close (f);
End.
```

```
Const
        n=10;
Var
        f:text;
        i,j:integer;
Begin
        Assign(f,'out.dat');
        Rewrite(f);
        For i:=1 to n do begin
                For  j:=1 to n do
                        Write(f, i*j: 5);
                Writeln(f);
        end;
        Close(f);
End.
```

در این برنامه با n=۱۰ یک جدول ۱۰×۱۰ به صورت زیر در خروجی فایل متنی out.dat داریم.

```
1       2       ...     10
2       4       ...     20
..
..
10      20      ...     100
```

مثال ۴-۱۴ برنامه‌ای بنویسید که از یک فایل متنی شامل اعداد صحیح، تعداد آنها را بدست آورده و در مانیتور چاپ کند.

```
Var
        f: text;
        num, count: integer;
Begin
        Assign(f,'int.dat');
        Reset(f);
        Count:=0;
        While  not  eof(f) do begin
                While not eoln(f) do begin
                        Read(num);
                        count:=count+1;
                end;
                Readln(f);
        end;
        Write('The count of numbers in file is: ', count);
        Close(f);
End.
```

توجه کنید در مواردی که فایل با Reset باز می‌شود و سپس عملیات نوشتن صورت می‌گیرد، خطای حین اجرای I/O ظاهر می‌شود و به همین صورت خواندن از

مثال ۲-۱۴ برنامه‌ای بنویسید که یک فایل متنی شامل چند جمله را از ورودی دریافت کرده و یک کپی از فایل درخروجی بسازد.

```
var
        f1,f2: Text;
        str1,str2: string;
        ch: char;
begin
        write('Enter   Input file: ');
        readln(str1);
        write('Enter output (copy) file: ');
        readln(str2);
        assign(f1,strl);
        reset(f1);
        assign(f2,str2);
        rewrite(f2);
        while not eof(f1) do begin
                while not eoln(f1) do begin
                        read(f1,ch);
                        write(f2,ch);
                end;
                readln(f1);
                writeln(f2);
        end;
        Close(f1);
        Close(f2);
End.
```

در این برنامه ابتدا دو فایل به نامهایشان نسبت داده می‌شوند و ترتیب برای خواندن و نوشتن باز می‌شوند. سپس توسط eof شرط خاتمه فایل ورودی یعنی f1 بررسی می‌شوند و در یک حلقه تو در تو while شرط خاتمه خطوط نیز توسط eoln بررسی شده و کاراکترها از فایل ورودی خوانده شده و در فایل خروجی نوشته می‌شوند. مواقعی که eoln است برای دو فایل ترتیب Readln و Writeln داریم که سطر عوض شود.

مثال ۳-۱۴ برنامه‌ای بنویسید که یک جدول ضریب n*n را در یک فایل خروجی ایجاد کند.

یک فایلی که با Rewrite باز شده است.

همانطور که از مثالهای فوق دیدید، توابع EOLN و EOF از نوع Boolean می‌باشد و برای استفاده فایل‌ها می‌باشد. تابع EOF در مورد فایلهای متنی فقط با دستور Reset معنی پیدا می‌کند.

۱۴-۲- فایلهای که دودویی و نوع‌دار(Binary & Typed)

این نوع فایل‌ها در زبان پاسکال از تایپهای مختلف char, integer, read, array, record و ... تشکیل شده است که نیاز به پردازش متنوع شبیه مرتب‌سازی، جستجو، حذف و... دارند. این فایلها سپس از ایجاد توسط برنامه، قابل رؤیت توسط ویراشگرها نیستند، بلکه به صورت کدهای اسکی می‌باشد یعنی دودویی می‌باشند. نحوه دسترسی به اطلاعات آنها نیز به صورت تصادفی (Random) است.

نحوه تعریف یک فایل دودویی نوع‌دار از انواع مختلف برای مثال در زیر آورده شده است:

```
Const  n=100;
Type
        Student = Record
                Name: string[10];
                Family: String[15];
                Age: integer;
                ID: integer;
        end;
        Sarray = Array [1..n] of student;
Var
 F1,f2: Text;                                  فایلهای متنی
 F3: Text;
 Bf1: file of char;                            فایل دودویی از نوع کاراکتر
  Bf2: file of integer;                        فایل دودویی از نوع صحیح
  Bf3: file of student;                        فایل دودویی از نوع رکورد دانشجو
  Bf4: file of Sarray;                         فایل دودویی از نوع آرایه صدتایی از دانشجویان
 Bf5: file of array[1..n] of Real;             فایل دودویی از نوع آرایه صدتایی از اعداد حقیقی
 Bf6: file of set of 'A'..'Z';                 فایل دودویی از مجموعه A تا Z
 Bf7: file of (sat,sun,mon,tue,wed,thu,fri);   فایل دودویی از شمارش شنبه تا جمعه
```


مثال ۱۴-۵ برنامه‌ای بنویسید که تعداد رشته (حداکثر ۱۰ تایی) از ورودی خوانده و در یک فایل دودویی بنویسید.

```
Const n= 10;
Var
        Bf: file of strin[10];
        Str: string [10];
        i: integer;
Begin
        Assign (Bf,'out.dat');
        Reset (Bf);
        For i:=1 to n do
        Begin
                Readln (str);
                Write (Bf,str);
        End;
        Close (Bf);
End.
```

۱۴-۳- فایلهای باینری بدون نوع (Binary & Untyped)

این فایلها مشابه فایلهای نوع از نوع دودویی بوده، یعنی با ویرایشگرها قابل مشاهده نیستند ولی برخلاف آنها از نوع خاصی نیز می‌باشند. در این فایلها طول رکوردها مساوی نیست و می‌توان هر نوع داده‌ای با طول و نوع مختلف در آن ذخیره کرد. در این نوع فایلها نیز خط معنی نداشته و فقط انتهای فایل دارای معنی است که کاراکتر با کد ۲۶ می‌باشد.

طریقه تعریف فایلهای بدون نوع ساده و به صورت زیر است:

```
Var
        Bf: file;
```

با این تعریف یک فایل بدون نوع تعریف شده است که می‌توان هر نوع داده‌ای و با هر مقدار و طول در آن نوشت. معمولاً این نوع فایلها بدلیل سرعت زیاد بیشتر در کارهای سیستمی نظیر حذف فایل تغییر نام فایل و... استفاده می‌شود.

در جدول صفحه بعد تمام روالهای کتابخانه‌ای پاسکال جهت کار با فایلها آورده است که به همراه توضیحات لازمه جهت کار با آنها می‌باشد. همچنین برای هر تابع و رویه مشخص شده است که مربوط به چه فایلی می‌باشد.

جدول ۱-۱۴ جدول توابع فایلها

ردیف	نام دستور	عملیات مربوطه	فایلهای متنی	فایلهای نوع دار	فایلهای بدون‌نوع
۱	Assign	نام فایل را به متغیر فایل نسبت می‌دهد.	+	+	+
۲	Reset	فایل مربوطه را برای خواندن باز می‌کند.	+	+	+
۳	Rewrite	فایل مربوطه را برای نوشتن باز می‌کند.	+	+	+
۴	Reatd	خواندن از فایل	+	+	+
۵	Readin	خواندن از فایل متنی	+	×	×
۶	Wrile	نوشتن در فایل	+	+	+
۷	Writeln	نوشتن در فایل متنی	+	×	×
۸	Eoln	مشخص نمودن انتهای خط	+	×	×
۹	Eof	مشخص نمودن انتهای فایل	+	+	+
۱۰	Append	بازنمودن فایل متنی جهت اضافه نمودن	+	×	×
۱۱	Close	بستن فایل	+	+	+
۱۲	Seek	انتقال وحرکت مکان‌نمادر فایل دودویی	×	+	+
۱۳	Seekoln	مشخص نمودن رسیدن به انتهای خط فایل متنی	+	×	×
۱۴	Seekeof	مشخص نمودن رسیدن به انتهای فایل متنی	+	×	×
۱۵	Filefos	شماره رکورد از انتهای فایل	×	+	+
۱۶	Filesize	تعداد رکوردهای فایل جاری را برمی‌گرداند	×	+	+
۱۷	Truncate	حذف یک رکورد از انتهای فایل	×	+	+
۱۸	Erase	برای حذف فایلها بکار می‌رود.	+	+	+
۱۹	Rename	برای تغییر نام فایلها بکارمی‌رود.	+	+	+
۲۰	Flush	محتویات بازفایل متنی را به دیسکت انتقال می‌دهد.	+	×	×
۲۱	SetTextbuf	اندازه بافر برای فایلهای متنی را برمی‌گرداند.	+	×	×
۲۲	Blockread	خواندن یک بلاک از فایل بدون نوع	×	×	+
۲۳	Blockwrite	نوشتن یک بلاک از فایل بدون نوع	×	×	+

۴-۱۴- مثالها

با توجه به مثالهای فایلهای متنی دیده می‌شود که همه موارد برنامه‌نویسی در اینجا نیز صادق است و فقط موارد ذخیره‌سازی از مانیتور به فایل یعنی حافظه جانبی تبدیل شده است.

مثال ۶-۱۴ برنامه‌ای بنویسید که تعداد خطوط یک فایل متنی را بدست آورد.

```
Var
        F: text;
        ch:char;
        str: string[20];
        count: integer;
Begin
        Write ('Enter the file  name: ');
        Readln(str);
        Assign(f,str);
        Reset(f);
        count:=0;
        While not eof(f) do begin
                While not eoln(f) do
                        Read(f,ch);
                Readln(f);
                count:=count+1;
        end;
        Write('the number of lines in file is: ',count);
End.
```

مثال ۷-۱۴ رکوردی از کتاب‌های در نظر گرفته، تعداد۱۰ کتاب را در یک فایل نوع دارد ذخیره کنید.

```
Const
        n   10;
Type
        Booktype = Record
                name: string[20];
                ID: integer;
        end;
Var
        Bf: file of Booktype;
        i,ID: integer;
        name: string[20];
        Book: Book type;
Begin
        Assign (Bf,'book.dat');
        Reafite (Bf);
        For i:=1 to n do
        begin
                Write ('Enter the name & ID of book: ');
                Readln (name, ID);
                Book.name:= name;
                Book.ID:= ID;
                Write (Bf, Book);
        End;
        Close (Bf);
End.
```

مثال ۸-۱۴ برنامه‌ای بنویسید تعدادی عدد صحیح را از ورودی خوانده و در فایل نوع‌دار ذخیره شود. سپس مجدداً از این فایل تمام اعداد به روی مانیتور نشان داده شوند. فرض کنید انتهای اعداد ورودی با عدد صفر باشد.

```
Var
        Bf: file of integer;
        i: integer;
Begin
        Assign (Bf, 'out.dat');
        Rewtite (Bf);
        Repeat
                Readln (i);
                Writeln (Bf,i);
        Until i=0;
        Reset(Bf);
        While not eof(Bf) do begin
                Read (Bf,i);
                Write (i:5);
        End;
        Close (Bf);
End.
```


۱۴-۵ تمرینات

۱- برنامه‌ای بنویسید که از یک فایل متنی از اعداد حقیقی که در سطرها و ستونها ی مختلف قرار دارند، میانگین داده‌ها را بدست آورده و در مانیتور چاپ کند.

۲- برنامه‌ای بنویسید که در انتهای یک فایل متنی، همان فایل متنی را اضافه کند.

۳- برنامه‌ای بنویسید که در انتهای یک فایل بی نوع، همان فایل را اضافه کند.

۴- برنامه‌ای بنویسید که در وسط یک فایل نوع‌دار از رکورد دانشجو، یک رکورد جدید اضافه کند.

۵- برنامه‌ای بنویسید که از ابتدای یک فایل نوع‌دار از آرایه، یک آرایه حذف کند.

۶- برنامه‌ای بنویسید که یک فایل بدون‌نوع را ابتدا تغییر نام دهد، سپس حذف کند.

۱۴.۶ تمرینات برنامه‌نویسی

۱- برنامه‌ای بنویسید که در یک فایل نوع دار از کاراکترها، اطلاعات آماری هر کاراکتر را بدست آورده، یعنی تعداد هر کاراکتر را بدست آورد. مثلاً کاراکتر 'A' به تعداد ۴۵ تا و کاراکتر 'B' به تعداد ۳۶ تا و...

(راهنمایی: از ساختار روبرو استفاده کنید: Count: Array['A'..'Z'] of integer)

۲- برنامه‌ای بنویسید که در انتهای یک فایل نوع‌دار، همان فایل را اضافه کند.

۳- برنامه‌ای بنویسید که رکوردهای مشخصات تاریخ مسئله ۱ تمرینات برنامه‌نویسی فصل ۱۱ را به کمک فایلها پیاده سازی کند. فایلهای ذخیره شده باید توسط برنامه در اجرای بعدی قابل خواندن باشد.

۴- برنامه‌ای بنویسید که رکوردهای مشخصات دانشجویان مسئله ۲ تمرینات برنامه‌نویسی فصل ۱۱ را به کمک فایلها پیاده‌سازی کند. فایلهای ذخیره شده باید توسط برنامه در اجرای بعدی قابل خواندن باشد.

۵- برنامه‌ای بنویسید که رکوردهای مشخصات کتابهای مسئله ۳ تمرینات برنامه‌نویسی فصل ۱۱ را به کمک فایلها پیاده‌سازی کند. فایلهای ذخیره شده باید توسط برنامه در اجرای بعدی قابل خواندن باشد.

فصل ۱۵

تحلیل الگوریتمها

هدفهای کلی

- مفهوم و تعریف الگوریتم
- مفهوم کارایی یک الگوریتم
- مرتبه یک الگوریتم

هدفهای رفتاری

دانشجو پس از مطالعه این فصل باید بتواند:

- برنامه خود را تحلیل زمانی نماید.
- مرتبه الگوریتم را بدست آورد.
- الگوریتمهای بازگشتی را تحلیل نماید.

مقدمه

تحلیل یک الگوریتم یعنی ارزیابی روشهای مختلف حل آن مسئله، بررسی و محاسبه بهترین و بدترین حالتها، بصورتی که با توجه به شرایط بهترین حالت را بتوان انتخاب کرد. الگوریتم در واقع تعداد محدودی از دستورالعملها هستند که بترتیب اجرا می‌شوند و هدف خاصی را دنبال می‌کنند و دارای پنج خصوصیت زیر است:

۱- ورودی: می‌تواند صفر یا چندین ورودی داشته باشد.

۲- خروجی: حداقل یک خروجی وجود دارد.

۳- قطعیت: باید عاری از ابهام باشد یعنی قاطع باشد.

۴- محدودیت: باید خاتمه‌پذیر باشد یعنی محدود باشد.

۵- کارایی: یعنی کارا باشد و سریع به جواب برسد.

در تحلیل الگوریتم سعی بر اینست که الگوریتم کارا باشد زیرا چهار مورد اول شروطی هستند که هر الگوریتم باید داشته باشد وگرنه ناقص است ولی مورد پنجم سرعت و کارایی الگوریتم را بالا می‌برد.

۱-۱۵- تعریف مرتبه یا پیچیدگی الگوریتم (O بزرگ)

برای بدست آوردن بدترین حالت اجرای یک الگوریتم یا مرتبه پیچیدگی الگوریتم از O بزرگ استفاده می‌کنیم و بنا به تعریف عبارتست از $f(n)=O(g(n))$ و می‌خوانیم $f(x)$ از مرتبه $g(x)$ می‌باشد، اگر و تنها $\exists c>0, \forall n>n0 \Rightarrow f(x)<c.g(x)$

مثال ۱-۱۵ $f(n)=2n+1$ می‌توان با در نظرگرفتن $g(n)=n$ و $c=3$ به واقعیت زیر رسید:

$f(n)=2n+1 <= cg(n)=3n \qquad n>=1$

لذا می‌توان گفت، پیچیدگی زمانی $f(n)$ از مرتبه $O(n)$ می‌باشد.

مثال ۲-۱۵ $f(n)= 10n^2+n+1$ می‌توان با در نظرگرفتن $g(n)=n^2$ و $C=11$ به واقعیت زیر رسید:

$f(n)=10n^2+n+1 < 11n^2$

به ازای تمام $n>2$ لذا داریم $f(n) = O(n^2)$

بطور کلی برای توابع خطی از درجه k داریم، پیچیدگی زمانی از مرتبه درجه بزرگتر می‌باشد یعنی $O(n^k)$ می‌باشد.

مثال ۳-۱۵

$f1(n) = 100n^3+n^2+n = O(n^3)$

$f2(n) = n^2+n+1 = O(n^2)$

$f3(n) = \log n + 3n^2 = O(n^2)$

$f4(n) = n^2 \log e + n^3 = O(n^3)$

$f5(n) = 2n \log n + \log n = O(n \log n)$

$f5(n) = n^2 \log n + 2^n + n^2 + n = O(2^n)$

در مثال فوق توجه کنید که منظور از Log همان لگاریتم در مبنای ۲ می‌باشد.

بطورکلی درجه توابع پیچیدگی زمانی در زیر بترتیب از چپ به راست آورده شده است که بترتیب بیشتر می‌شود:

$O(1) < O(\log n) < O(n) < O(n \log n) < O(n^2) < O(n^3) < ... < O(2^n) < O(n!)$

توجه کنید که تابع سمت راست از سمت چپ بزرگتر است و هر تابع سمت چپ از سمت راست کوچکتر می‌باشد، یعنی در واقع درجه تابع سمت راست از سمت چپ بیشتر است.

۲-۱۵- بدست آوردن مرتبه الگوریتمها

برای بدست آوردن مرتبه اجرای الگوریتمها باید به دقت بررسی شود و تعداد تکرار آن الگوریتم بدست آورده شود. یعنی در واقع تعداد تکرارها بدست آورده شود و سپس با هم جمع شده و مطابق مثالهای قبلی به پیچیدگی زمان واقعی رسید. برای این منظور از الگوریتمهای ساده شروع کرده و به پیچیده می‌رسیم.

فرض کنید الگوریتمی فقط دارای جملاتی نظیر x:=x+1 می‌باشد، حال ممکن است تعداد آنها زیاد باشد. چون این جملات فقط یک بار تکرار می‌شوند لذا اگر c جمله باشد c*1 یعنی c تکرار داریم که c ثابت است و لذا از O(1) می‌باشد.

فرض کنید الگوریتمی دارای حلقه‌های ساده به شکل زیر باشد:

```
x:= 0;
for i:=1 to n do
        x:=x+1;
```

که n ورودی می‌باشد. یعنی حلقه for بستگی بر پارامتر n دارد. همان طور که از تعریف حلقه for برمی‌آید تعداد n-۱+۱ تکرار یعنی n تکرار وجود دارد و لذا از O(n) می‌باشد.

حال درنظر بگیر ید از نوع حلقه‌های فوق چند تا باشد مثلاً در زیر دو حلقه for داریم:

```
x:= 0;
y:= 0;
For i:=1 to n do
        x:= x+1;
For  j:=1 to n do
        y:= y+1;
```

و لذا برای هر یک از حلقه ها داریم n تکرار، یعنی n+n=۲n تکرار داریم. ولی باز هم مرتبه ۲n=O(n) می‌باشد یعنی تعداد حلقه‌های فوق در مرتبه تأثیری ندارد. ولی دقت شود که حلقه‌های فوق مستقل از هم بوده و تودرتو نیستند.

حال فرض کنید الگوریتمی دارای حلقه تودرتوی زیر باشد:

```
x:= 0;
for  i:=1 to n do
   for  j:=1 to n do
        x:=x+1;
```

طریقه اجرای این حلقه تودرتو در پاسکال بدین نحو است که ابتدا i:=1 شده و برای j:=1 تا j:=n عملیات انجام می‌پذیرد و سپس i:=2 شده و عملیات برای j:=1 تا j:=n انجام می‌پذیرد و ...

پس به ازای هر i داریم n تکرار، لذا چون تعداد i ها برابر n می‌باشد در نتیجه n*n تکرار داریم یعنی n^2 تکرار و لذا از $O(n^2)$ می‌باشد، یعنی مرتبه اجرایی حلقه فوق برابر n^2 است.

مثال ۴-۱۵ مرتبه تابع زیر را بدست آورید:

```
function fact (n: integer): integer;
var
        f,i: integer;
begin
   f:=1;
   for i:=1 to n do
        f:=f*i;
   fact:=f;
end;
```


تابع فوق فاکتوریل عدد n را بر می‌گرداند. همانطور که می‌بینید تابع شامل یک حلقه for ساده بوده که از یک تا n متغیر بوده و لذا n بار تکرار می‌شود. ۲ جمله ساده نیز وجود دارد ولذا تعداد تکرارها برابر است با n+۲ که از O(n) می‌باشد.

مثال ۵-۱۵

```
function max(a,b:integer): integer;
begin
   if a>b then  max:=a
   else  max:=b;
end;
```

تابع فوق از بین دو عدد ورودی، عدد بزرگتر را بعنوان ماکزیمم برمی‌گرداند. همانطور که مشاهده می‌شود، تنها یک جمله if- else وجود دارد و تکراری وجود ندارد و لذا از O(۱) می‌باشد.

مثال ۶-۱۵

```
function sum(n:integer): integer;
var
   s,i: integer;
begin
   s:=0;
   for  i:=1 to n do
        s:= s+A[i];
   sum:= s;
end;
```

این تابع آرایه A به طول n از اعداد صحیح را دریافت کرده و جمع عناصر آنرا محاسبه کرده و برمی‌گرداند. مشابه مثال ۱، این تابع دارای یک حلقه for بوده که از یک تا n می‌باشد لذا دارای n تکرار بوده که از O(n) می‌باشد.

مثال ۷-۱۵

```
Procedure bubble(n: integer);
Var i,j: integer;
Begin

      For i:=1 to n-1do

         For  j:=i+1 to n do
            if  A[i]>A[j] then Swap (A[i], A[j]);
End;
```

که آرایه A از ۱ تا n را دریافت کرده و به روش bubble sort بصورت صعودی مرتب می‌کند.

(تابع Swap یک جابه‌جایی است و دو مقدار ورودی را جابجا یا عوض می‌کند.)

همانطور که مشاهده می‌کنید این تابع از یک حلقه تودرتو For تشکیل شده است که حلقه اول از ۱تا n-1 و حلقه دوم از i+۱ تا n می‌باشد و چون عملیات داخل حلقه‌ها تکرار ندارد، داریم:

لذا از $O(n^2)$ باشد. توجه کنید که تابع swap از مرتبه O(1) می‌باشد وگرنه در $O(n^2)$ ضرب می‌شد.

مثال ۸-۱۵

```
x:=0;
i:=n;
while i>1 do
begin
   i:=i div 2;
   x:=x+1;
end;
```

برنامه شامل یک حلقه while می‌باشد و در داخل آن i یعنی متغیر حلقه while هر بار نصف می‌شود و لذا طبق محاسبات ریاضی می‌توانیم داشته باشیم $n = 2^x$ یعنی $x=\log n$ و لذا چون x تعداد تکرار حلقه while می‌باشد. (x مانند شمارنده می‌باشد و بعد از هر عملیات یکی بیشتر می‌شود) پس مسئله از O(logn) می‌باشد. برای نمونه $n = ۱۶$ داریم:

۱	۱۶
۲	۸
۳	۴
۴	۲
-	۱

همانطور که می‌بینید به ازای $n = ۱۶$ تعداد ۴ یعنی log۱۶ تکرار داریم که همان مرتبه الگوریتم می‌باشد.

مثال ۹-۱۵

```
function fact(n:integer):integer;
begin
   if n=1 then fact:=1
   else fact:=n*fact(n-1);
end;
```

این تابع یک تابع بازگشتی بوده که فاکتوریل عدد ورودی n را حساب می‌کند. در واقع تابع بدین صورت عمل می‌کند که بعد از دریافت ورودی، اگر آن معادل یک بوده عدد یک یعنی ۱=!۱ را برمی‌گرداند و اگر بزرگتر بود آن را در else یعنی n را در مقدار fact(n-1) ضرب می‌کند که fact(n-1) همان n-1 در fact(n-1) می‌باشد و لذا مطابق ریاضیات fact:=n*(n-1)*(n-2)*...*fact(1) محاسبه می‌کند و چون این شبیه یک حلقه for از ۱ تا n می‌باشد (مثال ۲) و لذا از O(n) می‌باشد.

مثال ۱۰-۱۵ به ازای ورودی n مقدار خروجی تابع زیر و پیچیدگی زمانی آنرا را بدست آورید:

```
Function test (n:integer):integer;
Begin
        If n=1 then test:=2
        else test:= test(n-1)+1;
End;
```

این تابع، یک تابع بازگشتی بوده و داریم:

$T(1) = 2$
$T(n) = T(n-1) + 1$

معادله بازگشتی فوق را می خواهیم حل کنیم. داریم:

$$T(n) = T(n-1) +1= [T(n-2)+1]+1=[[T(n-3)+1]+1]+1=\ldots=$$
$$T(1)+1+1+\ldots+1=2+n-1=n+1$$

همانطور که می بینید با n بار شکستن مسئله فوق به جواب رسیده ایم، لذا از مرتبه $O(n)$ می باشد. برای خروجی تابع به ازای ورودی n داریم n+1 یعنی T(n)=n+1

۱۵-۲- تمرینات

۱- پیچیدگی زمانی توابع زیر را بدست آورید:

1. $f1(n) = n + 2n^2 + 100$
2. $f2(n) = n^3 + n^{\log n} + n + 2$
3. $f3(n) = 2^n + 5^n + \log n! + n^n$
4. $f4(n) = \log n + \sqrt{n} + 1$
5. $f5(n) = n^2 \log n + n^{\log n} + (\log n)!$

۲- پیچیدگی زمانی تابع زیر را بدست آورید:

```
Function test(n:integer): integer;
var
        i,x: integer;
Begin
        x:=0;
        for i:=1 to n do
                x:=x+1;
        x:=x+1;
        for i:=1 to n do
          for j:=1 to n do
                x:=x+1;
        test:=x;
End.
```

(جواب از $O(n^2)$ می باشد)

۳- پیچیدگی زمانی تابع زیر را بدست آورید:

```
Function test (n:integer):integer;
Begin
        If n=1 then test:=1
        else test:= test(n-1)+2;
End;
```

به ازای ورودی n خروجی تابع را بدست آورید.

(خروجی برابر ۲n-۱ و جواب از $O(n)$ می باشد)

۴- پیچیدگی زمانی تابع زیر را بدست آورید:

```
Function test (n:integer):integer;
Begin
        If n=1 then test:=1
        else test:= 2*test(n-1)+1;
End;
```

به ازای ورودی n خروجی تابع را بدست آورید.

(جواب از $O(n)$ می باشد)

۵- پیچیدگی زمانی تابع زیر را بدست آورید:

```
Function test (n:integer):integer;
Begin
        If n=1 then test:=1
        else test:= 2*test(n/2)+ 1;
End;
```

به ازای ورودی n خروجی تابع را بدست آورید.

(جواب از $O(\log n)$ می باشد)

۶- پیچیدگی زمانی تابع زیر را بدست آورید:

```
Function test (n:integer):integer;
Begin
        If n=1 then test:=1
        else test:= test(n-1)+ test(n-1);
End;
```

به ازای ورودی n خروجی تابع را بدست آورید.

(جواب از $O(2^n)$ می باشد)

۷- پیچیدگی برنامه را بدست آورید.

```
i:=n,;
X:=0;
For i:=1 to n do
        While i > 1  do begin
                i:=i div 2;
                x:=x+1;
        End;
```

(جواب از $O(n \log n)$ می باشد)

۸- پیچیدگی تابع زیر را بدست آورید.

```
Function bs(l,h:integer; x:integer):integer;
Var
        mid: integer;
Begin
        While l<=h do begin
                mid:= (l+h) dir 2;
                if A[mid] = x then bs:=A[mid]
                else if A[mid] > x then h:=mid-1
                else l:= mid+1;
        end;
        bs:= -1;
End;
```

A آرایه‌ای مرتب شده از اعداد صحیح می‌باشد.

این تابع چه عملیاتی روی آرایه ورودی A انجام می‌دهد.

(جواب از O(log n) می باشد)

۹- روال زیر را درنظربگیرید. پیچیدگی زمانی آنرا بدست آورید.

```
Procedure  test(n:integer);
Var
        i,j,k,x: integer;
Begin
        X:=0;
        For i:=1 to n do
          For  j:=i  to n do
                x:=x+1;
        If  x > n  then
                For  i:=1 to n do
                        x:=x+1;
        x:=x+1;
        x:=x+1;
        For  i:=1 to n do
          For  j:=1 to n do
                For  k:=1 to n do
                        x:=x+1;
        If x=n*n  then x:=x+1;
End;
```

(جواب از $O(n^3)$ می باشد)

۱۰- تابعی بنویسید که آرایه‌ای از اعداد صحیح را از ورودی دریافت کرده و حاصلضرب آنها را برگرداند. این تابع را به دو روش بازگشتی و غیربازگشتی نوشته و سپس پیچیدگی زمانی هر دوتابع را بدست آورده و با هم مقایسه کنید.

۱۱- تابعی بنویسید که عدد صحیح و بزرگ n را از ورودی دریافت کرده و تعداد ارقام آنرا بدست آورد. تابع را تحلیل زمانی کرده و مرتبه انرا بدست آورید.

۱۲- تابعی بنویسید که رشته عددی را از ورودی دریافت کرده و آنرا به معادل عدد صحیح آن تبدیل کرده و به مقدار تابع برگرداند. تابع را تحلیل زمانی کرده و مرتبه آنرا بدست آورید.

۱۳- تابعی بنویسید که ماکزیمم عناصر یک آرایه را بدست آورده و به خروجی تابع برگرداند. این تابع را به دو روش بازگشتی و غیربازگشتی نوشته و سپس پیچیدگی زمانی هر دو تابع را بدست آورده و با هم مقایسه کنید.

ضمیمه ۱

سئوالات چهار جوابی

۱- کدام نوع متغیر زیر جزو متغیرهای مجاز در پاسکال نیست؟

الف) float ب) integer ج) real د) string

۲- کدام شناسه صحیح نیست؟

الف) eof ب) last word ج) round د) true

۳- کدام شناسه در زبان پاسکال صحیح است؟

الف) 2a ب) x-y ج) to د) sqr

۴ - در کدام حالت می توان مقدار اولیه یک ثابت را در برنامه تغییر داد؟

الف) به هیچ وجه قابل تغییر نیست ب) در هر صورت می توان تغییر داد

ج) نوع آن اعلام شده باشد د) نوع آن اعلام نشده باشد

۵- کدامیک از گزینه های زیر نادرست است؟

الف) write(False); ب) write(minint);

ج) write(maxint) د) write(True);

۶- متغیرهای TRUE , ARRAY , NEW به ترتیب کدامند؟

الف) مجاز ، مجاز و غیرمجازند ب) غیرمجاز ، مجاز و غیرمجازند

ج) غیرمجاز ، مجاز و مجازند د) مجاز ، غیرمجاز و مجازند

۷- فرض کنید b,a دو متغیر صحیح باشند. تکه برنامه زیر چه تغییری بر محتـوای b,a را باعث می گردد؟

```
a:=a+b;
b:=a-b;
a:=a-b;
```

الف) همواره باعث جابجایی محتوای b,a می شود.

ب) گاهی اوقات باعث جابجایی محتوای b,a می شود.

ج) هیچ تغییری بر محتوای b,a نمی دهد.

د) همواره باعث یکسان شدن محتوای b,a می شود.

۸- در زبان پاسکال برای محاسبه a/b نوع متغیرهای b,a :

الف) باید real باشند. ب) می توانند char باشند.

ج) باید integer باشند. د) می توانند real یا integer باشند.

۱ - دستور صحیح در زبان پاسکال کدام است؟

الف) Const i:=12; ب) const i:=14.5;

ج) Var i:=integer; د) Var i:=integer;

۲- برای تعریف متغیر از نوع اعشاری کدام گزینه زیر صحیح است؟

الف) Byte ب) Word ج) Integer د) Real

۳- انواع داده ای در پاسکال استاندارد کدامند؟

الف) String , Boolean , Char , Real , Byte , Integer

ب) Boolean , Char , Single , Real , Byte , Longint , Integer

ج) Boolean , Char , Extended , Single , Real , Byte , Shortint , Integer

د) Boolean , Char , Real , Byte , Integer

۴- با توجه به دستور مقابل ثابت test از چه نوعی خواهد بود؟

```
Const test = TRUE;
```

الف) Boolean ب) char

ج) در برنامه مشخص خواهد شد د) در var مشخص خواهد شد

۵- کدام گزینه ، از کلمات رزرو شده پاسکال می باشد؟

الف) Integer ب) Readln ج) Var د) TRUE

۶- خروجی این برنامه چیست؟

```
Var
  Write : Integer
Begin
  Write := 3;
  Write ('Hello');
End.
```

الف) Hello ب) خطای کامپایلری ج) 3 د) خطا در هنگام اجرا

۷- در کدام یک از گزینه ها اعلام خطا دارد؟

```
Var A:longint;  B:byte;
Begin
   a:20000;
   b:300;
   b:a;
   Write(b);
End.
```

الف) b:=a; ب) b:=a; b:=300

ج) b:=300 د) a:=20000

۸- حاصل عبارت 16*5 DIV 3*2 MOD 3 کدام است؟

الف) 0 ب) 1 ج) 2 د) 3

۹- در دستورات زیر مقدار متغیر i چند خواهد بود؟

```
Var    I:Integer;  L:Longint;
Begin
    L:10*50;
    I:L*20+10 mod 12;
End.
```

الف) 2

ب) دستور انتساب I اشکال دارد و کامپایلر کدخطای انتساب نوع متفاوت راصادرمیکند

ج) 10010

د) 10002

۱۰- کدام گزینه درست و از نوع صحیح است؟

الف) A mod B*Div C ب) A(Div B mod C)

ج) A/B+C Div D د) A Div B mod C

۱۱- اولویت اجراء کدام عملگر بالاتر است؟

الف) AND ب) IN ج) OR د) NOT

۱۲- حاصل عبارت 3*5+3 MOD 3*5-4 کدام است؟

الف) 10 ب) 13 ج) 14 د) 18

۱۳- فرض کنید X=3.0 و Y=4.0 و Z=2.0 و FLAG=TRUE باشد در این صورت ارزش عبارتهای c , b , a به ترتیب کدام است؟

a) (X>Z) ZND (Y<Z)
b) NOT FLAG OR ((Y+Z)>=(X-Z))
c) NOT (FLAG OR ((Y+Z)>=(x-z)))

الف) درست ، درست ، نادرست ب) نادرست ، درست ، نادرست

ج) نادرست ، درست ، درست د) نادرست ، نادرست ، درست

۱۴- اگر z = 0.0 و x = 3.0 باشد در این صورت ارزش عبارتهای (z<x) AND (x<3.5) و (x = 1.0) OR (x = 4.0) به ترتیب چه خواهد شد؟

الف) True , True ب) False , True ج) True , False د) False , False

۱۵- اگر از ورودی به ترتیب مقدار 4 برابر متغیر A و مقدار 2 برای متغیر B دریافت شود ، خروجی برنامه کدام است؟

```
readln(A,B);    write(A>B);
```

الف) 4 ب) True ج) 4>2 د) False

۱۶- با اجرای برنامه زیر چه عملی انجام می شود؟

```
Var  x:Integer;
     y:real;
     z:Longint;
Begin
   x: = 1000;   y: = 60000;   z: = x+y;
    Writeln(z);
End.
```

الف) عدد 61000 چاپ می شود

ب) چون جواب (61000) از نوع Real می باشد ، برای دستور انتساب z خطای Type Mismatch صادر می شود.

ج) عدد 27233 -

د) خطای زمان اجرا بوجود می آید

۱۷- اولویت کدام عملگر بیشتر است؟

الف) AND ب) EQV ج) OR د) NOT

۱۸- کدام مورد پس از اجرای برنامه زیر در توربو پاسکال روی صفحه چاپ می شود؟

```
Var  X:Byte;
Begin
   X:50;
  Writeln(x*6);
End.
```

الف) از اجرای برنامه جلوگیری می شود ب) 50-

ج) 300 د) پیغام (Run Time Error)

۱۹- در عبارت منطقی زیر ، مقدار C چند باشد تا در پایان اجرای آن ، A بزرگتر از B شود؟

```
A: = 4;
B: = 7;
if (NOT (A<=B) OR (B>C-A)) AND (C>B-A)
  Then A: = B + A;
else
   A: = B – A ;
```

الف) 5 یا 7 ب) 5 ج) 7 د) 11

۲۰- خروجی برنامه زیر کدام است؟

```
program q1;
var
    x: boolean;
begin
    x: = 2 = 1;
   writeln(x);
end.
```

الف) FALSE ب) TRUE ج) 2 د) 1

۲۱- ارزش عبارت 5 DIV 3* 30 AND 56 کدام است؟

الف) 16 ب) 22 ج) 14 د) 58

۲۲- اگر A = 235 باشد ، حاصل A SHR 3 کدام است؟

الف) 705 ب) 29 ج) 78 د) 238

۲۳- خروجی برنامه زیر کدام است؟

```
I:= 3;
Write( x shr 2);
```

الف) FALSE ب) Run Time Error

ج) 3<3 د) خطای زمان ترجمه

۲۴- در زبان برنامه نویسی پاسکال کدام گزینه غلط می باشد؟(کارشناسی ارشد صنایع ـ دولتی ۷۸)

الف) 10.0/(10.0-1E1) ب) 42 DIV 26 (DIV 4 + 14)

ج) 20 + ((4.5/(0.5*10)) - 20 د) (24*(-2)) - 20) * (20 + 212 DIV 12)

۲۵- فرض کنید که متغیر FLAG یک متغیر منطقی است (که می تواند TRUE یا FALSE باشد) و A = 2 و B = 5 و C = 3. مقدار عبارت منطقی زیر با کدام گزینه برابر است؟ (کارشناسی ارشد صنایع ـ دولتی ۷۹)

الف) TRUE ب) FALSE

ج) TRUE تنها وقتی که FLAG = TRUE د) TRUE تنها وقتی که FLAG = FALSE

۲۶- معادل عبارت $\frac{F + \frac{A - B}{D}}{C*(C-1)}$ کدام گزینه است؟ (کارشناسی ارشد صنایع ـ دولتی ۷۹)

الف) (F + (A-B)/D)/C*(C-1) ب) F + ((A-B)/D)/(C*(C-1))

ج) D – C*D/(A + B/(A-D)) د) هیچکدام

۲۷- فرض کنید A = -1 , B = 2 , C = 0 و FLAG یک متغیر منطقی است. مقدار عبارت
(A<>B) AND ((B+A)>C) OR (NOT FLAG)
برابر است با ..... (کارشناسی ارشد صنایع ـ دولتی ۸۱)

الف) FALSE ب) TRUE

ج) TRUE تنها اگر FLAG = TRUE د) TRUE تنها اگر FLAG = FALSE

۲۸- حاصل عبارت زیر چیست؟ (کارشناسی ارشد صنایع ـ دولتی ۸۱)
NOT (-4.2<>3.0) OR NOT 10<20

الف) 4 ب) TRUE ج) FALSE د) INVALID

۲۹- عبارتهای منطقی (P AND P) و (P OR P) : (کارشناسی ارشد ریاضی ـ دولتی ۷۴)

الف) معادل یکدیگرند

ب) همواره مخالف یکدیگرند

ج) معادل یکدیگرند فقط وقتی که مقدار P نادرست است

د) معادل یکدیگرند فقط وقتی که مقدار P درست باشد

۳۰- فرض کنیم LVAL یک متغیر منطقی و متغیرهای I , J , M , N به ترتیب برابر ۲، ۳، ۴ و ۵ باشند. بعد از اجرای دستور زیر گزینه صحیح در زیر کدام است؟ (کارشناسی ارشد ریاضی ـ دولتی ۷۵)

LVAL: = NOT(I<J) OR (M<N)

الف) TRUE ب) FALSE

ج) این دستور غلط است د) این دستور اجرا نمی شود

۳۱- وضعیت تقدم اجرای عملگرهای رابطه ای با عملگرهای منطقی در پاسکال چگونه است؟ (کارشناسی ارشد ریاضی ـ دولتی ۷۵)

الف) عملگرهای منطقی از تقدم بالاتری برخوردارند

ب) عملگرهای رابطه ای از تقدم بالاتری برخوردارند

ج) این عملگرها هم تقدم هستند

د) بستگی دارد که در یک عبارت کدامیک در سمت چپ عبارت ظاهر شوند

۳۲- عبارتهای منطقی (P AND P) و NOT (P OR P) (که در آنها P یک متغیر منطقی است) (کارشناسی ارشد ریاضی ـ دولتی ۷۵)

الف) همواره مخالف یکدیگرند ب) معادل یکدیگرند اگر P نادرست باشد

ج) معادل یکدیگرند اگر P درست باشد د) معادل یکدیگرند

۳۳- فرض کنید که متغیر FLAG یک متغیر منطقی است و A =2 , B =5 , C =3. گزینه صحیح در مورد عبارت منطقی کدام است؟ (کارشناسی ارشد ریاضی ـ دولتی ۷۶)

(A<>B) AND (C<B) OR FLAG

الف) تنها وقتی درست است که مقدار FLAG نادرست (FALSE) باشد

ب) تنها وقتی درست است که مقدار FLAG درست باشد

ج) همواره درست (TRUE) است

د) همواره نادرست است

۳۴- معادل عبارت زیر کدام است؟ (کارشناسی ارشد ریاضی ـ دولتی ۷۶)

$$\frac{F + \frac{A - B}{D}}{C*(C-1)}$$

الف) F + ((A-B)/D)/(C*(C-1)) ب) (F+(A-B)/D)/C*(C-1)

ج) (F+(A-B)/D)/(C*(C-1)) د) هیچکدام از موارد

۳۵- در پاسکال ، کدامیک از شرط زیر بیان کننده آن است که متغیر num بین ۱ تا ۹ ولی برابر عدد ۴ نیست؟ (فرض کنید num یک متغیر صحیح است) (کارشناسی ارشد ریاضی ـ دولتی ۷۷)

الف) (num>1) and (num<10) and (num<>4)

ب) (num>=1) and (num<9) and (num<>4)

ج) (num>1) and (num<9) and (num<>4)

د) (num>1) or (num<9) and (num<>4)

۳۶- عبارت معادل با a<=(a>=b) ، اگر a , b از نوع منطقی باشند کدام است؟

الف) A XOR B ب) A AND B ج) A OR B د) TRUE

۳۷- حکم معادل حکم زیر کدام است؟

(A <= 5) And (B >= 6)

الف) (A>5) OR (B<6) ب) Not ((A>5) And (b<6))

ج) Not ((A>5) OR (B<6)) د) Not (A>5) And (B<6)

۳۸- عبارت منطقی (A OR A) AND (B OR B) همواره برابر است با : (کارشناسی ارشد صنایع - دولتی ۸۰)

الف) A OR B ب) A AND B ج) NOT (A OR B) د) NOT (A AND B)

۳۹- مقدار (NOT (A AND B)) OR (NOT (A OR B)) درست است اگر ..... (کارشناسی ارشد صنایع - دولتی ۸۰)

الف) اگر A یا B نادرست باشند ب) تنها اگر A , B هر دو درست باشند

ج) تنها اگر A , B هر دو نادرست باشند د) تنها اگر B درست و A نادرست باشند

۴۰- فرض کنید FLAG , B = FALSE , A = TRUE یک متغیر منطقی است. مقدار عبارت (NOT FLAG) AND ((NOT A) OR B) AND (A AND (NOT B)) برابر است با ..... (کارشناسی ارشد صنایع – دولتی ۸۰)

الف) FALSE تنها اگر FLAG = FALSE ب) TRUE تنها اگر FLAG = TRUE

ج) FALSE د) TRUE

۴۱- فرض کنید FLAG , C = 0 , B = 1 , A = -1 یک متغیر منطقی است. مقدار عبارت : (A<>B) AND (A+B=C) AND (NOT FLAG)

برابر است با ..... (کارشناسی ارشد صنایع -- دولتی ۸۰)

الف) TRUE ب) FALSE

ج) TRUE اگر مقدار FLAG درست باشد د) FALSE اگر مقدار FLAG درست باشد

۴۲ فرض کنید P یک متغیر منطقی است که می تواند مقدار TRUE یـا FALSE داشـته باشد. مقدار عبـارت (P AND P) AND (P OR P) OR (NOT P) برابـر اسـت بـا .....
(کارشناسی ارشد صنایع - دولتی ۸۱)

الف) TRUE ب) FALSE

ج) TRUE تنها اگر P برابر TRUE باشد د) TRUE تنها اگر P برابر FALSE باشد

۴۳- کدام گزینه زیر معادل a می باشد؟ (a , b متغیرهای بولین هستند)

الف) a XOR a ب) a AND (a OR b) ج) a OR (a AND b) د) گزینه ب و ج

۴۴- خروجی قطعه برنامه روبرو کدام است؟

```
Var A,B: Byte;
Begin
    B:= A or $FF;
    A:= (Not B) + 1;
    A:= A+B;
    Writeln(A);
End.
```

الف) 0 ب) $FF ج) $0F د) 1

۴۵- در زبان پاسکال حاصل عبارت آخر چه خواهد بود؟

```
Flag:= false;
x:= 3;      y:= 4;
not (flag or ((y+z) >= (x-z)));
```

الف) 0 ب) true ج) false د) قابل محاسبه نیست

۱- اگر A = 135.256 باشد و دستور روبرو اجرا شود نتیجه خروجی کدام است؟

```
writeln(A:5:2);
```

الف) 135.5 ب) 135.26 ج) 135.265 د) 13.5265

۲- خروجی دستور زیر چگونه است؟

```
write(351.236:4:3);
```

الف) 351.2 ب) 351.23 ج) 351.236 د) 351.24

۳- اگر a: = 68.951 باشد حاصل اجرای دستور write(a:5:1) کدام است؟

الف) 68.9 ب) 68.96 ج) 69.0 د) 69.1

۴- دستور زیر چه چیزی چاپ می کند؟

```
write(''' The End''); write('');
```

الف) The End" ب) 'The End'); write

ج) این دستور اشکال داشته و چیزی چاپ نمی شود د) 'The End"

۵- اگر a: = 12.6351 باشد حاصل خروجی دستور write(a:0:2) کدام است؟

الف) 0 ب) 126.35 ج) 12.6 د) 12.64

۶- اگر از ورودی abc را وارد کنیم خروجی کدام گزینه است؟

```
Var a:char;
Begin
   readln(a);
  write(a);
End.
```

الف) / ب) a

ج) 'a' د) خطا اعلام می شود چون 'abc' رشته است

۷- اگر از ورودی اطلاعات را به صورت 14↵ 12.5 وارد کنیم در برنامه زیر کدام مورد اتفاق می افتد؟

A:↵

```
Var a,b: Real;
   P,q: Char;
   x: Integer;
Begin
   Read(a,b,p,q,x);
```

الف) چون دستور read صادر شده ↵ بعد از 14 را نمی خواند و به خط بعد نمیرود.

ب) خطا اعلام می شود چون x یک متغیر عدد است و ما کاراکتر A را وارد کرده ایم .

ج) کاراکتر A در متغیر p قرار می گیرد و منتظر مقداردهی متغیر x,p است.

د) دومین مقدار وارد شده عدد صحیح 14 است که نمی تواند اعشار باشد.

۸- کدام گزینه در مورد متغیر a که از نوع Boolean می باشد نادرست است؟

الف) write(a); ب) a: = True; ج) a: = False; د) Read(a);

۹- اگر دستورات داده شده به ترتیب اجرا شوند و داده های ورودی به صورت زیر باشند مقدار متغیرهای P3,P2,P1 به ترتیب کدام است؟

123 ↵

456 ↵

الف) 3,2,1 ب) 4,2,1 ج) 5,2,1 د) 5,4,1

۱۰- فرض کنید عدد a برابر 17487.266 باشد ، با فرم چاپی a:5:1 نتیجه چگونه خواهد بود؟

الف) 17487 ب) 17487.2 ج) 17487.3 د) 1750.3

۱۱- در صورتی که ورودی مقدار 16- باشد ، دستور روبرو چه کاری انجام می دهد؟

الف) مقدار متغیر A برابر 16 - خواهد شد.

ب) مقدار متغیر A برابر 16.0- خواهد شد.

ج) مقدار متغیر A برابر 1600.0- خواهد شد.

د) دستور ، خطا دارد .

۱۲ در صورتی که بخواهیم پس از اجرای دستورات روبرو در متغیر A حرف x و در متغیر B مقدار ۵ قرار گیرد ، شکل ورودیها چگونه باید باشد؟

```
var   A:Char;
      B:Byte;
.
.
.
Read(A,B);
```

الف) x 5 ب) x 5 ج) 5x د) x5

۱۳- خروجی برنامه زیر کدام است؟

```
VAR   J,I: Integer;
      F: LongInt;
BEGIN
      I: 100;
      J: 1000;
      F: I*J;
      Writeln(f:4);
END.
```

الف) 0000 ب) 10000 ج) 100000 د) سرریزی رخ می دهد

۱۴- در دستور Read جهت وارد کردن داده ها از کدام کلیدها جهت جدا کردن استفاده می شود؟

الف) space , Enter ب) space Tab Enter

ج) space , ; د) ; Tab /

۱۵- اگر متغیر Pi یک متغیر از نوع Real باشد خروجی دستور write کدام خواهد بود؟

```
Pi: = 3.1416;
Write(Pi:0:3);
```

الف) 3.142 ب) 3.141 ج) 3 د) 0

۱۶- اگر x , y متغیرهای صحیح و b از نوع بولین باشند کدام دستور نادرست است؟

الف) Read(Input, x, y); ب) Read(b,x);

ج) write(b,y); د) write(output,b);

۱۷ - خروجی دستور روبرو کدام است؟

```
writeln($23);
```

الف) 23 ب) $23 ج) TRUE د) 35

۱۸- اگر c,b,a متغیرهای صحیح باشند و در جواب ۲ دستور زیر داده 7 ↵ 6 5 را وارد کنیم ، محتوای متغیرها چه می شود؟

الف) a = 5 , b = 6 شده و منتظر ورود عددی برای c می شود.

ب) a = 5 , b = 6 , c = 7 می گردد.

ج) a = 5 شده و منتظر ورود دو عدد برای b , c می شود.

د) هنگام اجرا پیام خطا صادر می شود.

۱۹- اگر a , b , c متغیرهای صحیح بوده و در جواب ۲ دستور زیر داده های مشخص شده را وارد کنیم ، محتوای متغیرها چه می شود؟

```
Read(a,b);
Readln(c);
```

2 ↵

3 4 5 ↵

الف) a = 2 , b = 0 , c = 3 ب) a = 3 , b = 4 , c = 5

ج) a = 2 , b = 3 , c = 4 د) a = 2 شده و سپس پیغام خطا صادر می شود

۲۰- اگر i متغیر صحیح ، C2,C1 متغیرهای کاراکتری و S متغیر رشته ای باشند و در جواب برنامه زیر دادهٔ ALI ↵ وارد شود ، خروجی چه خواهد شد؟

53 ↵

الف) ALI53 ب) I 35

ج) در هنگام اجرا پیام خطا صادر می شود د) 531

۲۱ اگر s متغیر رشته ای و i متغیر صحیح باشند و داده 789 ↵ برای دستورات زیر وارد شود محتوای متغیرها چه خواهد شد؟

```
Read(s);
Read(I);
```

الف) پیام خطا صادر می شود.

ب) رشته 789 در s ذخیره شده و کامپیوتر منتظر ورود داده برای i می شود.

ج) s برابر تهی و i برابر 789 می گردد.

د) رشته 789 در s و عدد 13 در i ذخیره می گردد.

۲۲- اگر i متغیری صحیح و C2,C1 متغیرهای کاراکتری باشند و در جواب برنامه زیر داده ↵

56t را وارد کنیم ، محتوای متغیرها چه می شود؟

```
Read(I);
Read(C1);      Read(C2);
```

الف) C2=#13 , C1=t , i=56

ب) I=56 شده و سپس منتظر ورود داده برای C2,C1 باقی می ماند

ج) C2=#10 , C1=#13 , i=56

د) پیام خطا صادر می شود

۱- فرض کنید T , X , Y , Z مقادیر حقیقی هستند. پس از اجرای دستورالعمل زیر :

```
T: = X;
IF T>Y THEN T: = Y;
IF T>=Y THEN T: = Z;
```

آنگاه ..... (کارشناسی ارشد ریاضی ـ دولتی ۷۴)

الف) همواره یکی از مقادیر X , Z در T قرار داده می شود

ب) مقداری که در T قرار می گیرد همواره مخالف X است

ج) کوچکترین مقدار در بین مقادیر X , Y , Z در T قرار داده می شود

د) بزرگترین مقدار در بین مقادیر X , Y , Z در T قرار داده می شود

۲- خروجی برنامه زیر چاپ کدام مورد است؟

```
Var
   I: integer;
Begin
  For i:= 0 To 3 do
   Begin
      Writeln(i=i);
   End;
End.
```

الف) دوبار عبارت TRUE ب) چهار بار عبارت i=i

ج) چهار بار عبارت TRUE د) عبارت 0=0 , 2=2

۳- خروجی برنامه زیر چاپ کدام مورد است؟

```
Var
   i: integer;
Begin
  for i:= 0 to 13 do
    Begin
        writeln(i);
        inc(i);
        if  i= 7 then halt
    End;
End.
```

الف) اعداد فرد از ۱ تا ۷ ب) اعداد فرد ۱ تا ۱۳

ج) اعداد زوج از صفر تا ۱۲ د) اعداد زوج از صفر تا ۶

۴- خروجی برنامه زیر چاپ کدام مورد است؟

```
Var
    i: integer;
    j: boolean;
Begin
    i:= 1;
    repeat
          writeln(i);
          i:1+2;
          j:= i >10
    until j
End.
```

الف) اعداد زوج صفر تا ۱۰ ب) اعداد فرد ۱ تا ۹

ج) دارای خطا است د) فقط عدد ۱

۵- خروجی برنامه زیر کدام است؟

```
For  x:= 1 to 2 do
Begin
       write(x);
       for y:= 1 to 2 do
             Write('K');
End;
```

الف) 12KK ب) 2KK1KK ج) 1KK2KK د) 1K2K

۶- خروجی برنامه زیر کدام است؟

```
Var       i: Byte;
Begin
      for i:= 6 downto 1 Do;
      write(i);
End.
```

الف) 6 ب) 123456 ج) 654321 ج) 1

۷- دستور چاپ زیر چند بار تکرار می شود؟

```
For  i:= 1 to 5 do
For  j:= 12 to 15 do
For  K:= 8 Downto 7 do
WriteLn(i*j*k);
```

الف) 0 ب) 15 ج) 30 د) 40

۸- در اثر اجرای برنامه روبرو در خروجی چه چیز مشاهده خواهد شد؟

```
Program;
   Var x,y : Integer
Begin
   x:= 0;
   Repeat
      y:= x*y;
   write(y);
   x:= x + 0.2;
   Until x< >1;
End.
```

الف) خروجی 0.8 0.6 0.4 0.2 0 ب) خروجی 0.64 0.36 0.16 0.04 0

ج) خروجی 1.0 0.64 0.36 0.16 0.04 0 د) برنامه خطا دارد

۹- حلقه i در برنامه زیر چند بار اجرا خواهد شد؟

```
Var   i,j: integer;
Begin
       For i:= 1 to 10 do
          For j:= 1 to 10 do
                 if  j>8 then
                  i:= i - 8;
       else
                 i:= i + 2;
End
```

الف) 6 ب) 10 ج) 18 د) بی نهایت (43694)

۱۰- کدام گزینه در مورد برنامه زیر صحیح است؟

```
i:= 1;
While 2<5 Do
Begin
      Write(i);
    i:= i + 1
End;
```

الف) خطای زمان ترجمه رخ میدهد

ب) حلقه while ، بی نهایت بار اجرا میشود

ج) حلقه while ، سه بار اجرا می شود

د) Run time Error رخ می دهد

۱۱- در برنامه زیر ، دستور writeLn(x) چند بار خواهد شد؟

```
   x: =1;
   While x<5 Do;
Begin
      WriteLn(x);
      x:= x + 1;
End;
```

الف) 0 ب) 1 ج) 4 د) 5

۱۲- فرض کنید که A , B متغیرهای منطقی (BOOLEAN) هستند که هر یک می تواند یکی از مقادیر درست (TRUE) یا نادرست (FALSE) را داشته باشد. مقدار عبارت منطقی (X OR X) AND (Y AND Y) همواره با کدام گزینه برابر است؟ (کارشناسی ارشد - دولتی ۷۹)

الف) XOR ب) X AND Y

ج) NOT (X OR Y) د) (X OR Y) OR (X AND Y)

۱۳- پس از اجرای قطعه برنامه زیر ، آخرین عددی که در خروجی نوشته می شود ، چیست؟ (کارشناسی ارشد صنایع - دولتی ۷۹)

```
EPS:= -1.0;
WHILES EPS<0.0 DO
   Begin
         Writeln(EPS);
         EPS:= EPS/2.0
End;
```

الف) صفر

ب) بزرگترین عدد منفی قابل نمایش در ماشین

ج) کوچکترین عدد منفی قابل نمایش در ماشین

د) PES به طوری که نمایش (1+EPS) در ماشین برابر ۱ است

۱۴- اجرای قطعه برنامه زیر (کارشناسی ارشد صنایع - دولتی ۸۰)

```
EPS:= 1.0;
WHILE(EPS>0.0) DO
```

```
EPS: = EPS/2;
WRITELN(EPS);
```

الف) منجر به نوشتن صفر در خروجی می شود

ب) منجر به نوشتن روند عدد یک در خروجی می شود

ج) منجر به نوشتن کوچکترین عدد مثبت قابل نمایش در کامپیوتر می شود

د) هیچگاه متوقف نمیشود و بنابراین خطای زمان اجرای بیش ازحد مجاز صادر میشود

۱۵- قطعه برنامه زیر را در نظر بگیرید.

```
M: = 1000;
SUM: = 0;
WHILE (M > 1) DO
BEGIN
      M: = M DIV 2;
      SUM: = SUM + M
END;
```

پس از اجرای این قطعه برنامه مقدار SUM برابر است با : (کارشناسی ارشد صنایع دولتی ۸۰)

الف) 993 ب) 994 ج) 995 د) 996

۱۶- قطعه برنامه زیر را در نظر بگیرید. (فرض کنید که M یک عدد صحیح مثبت است.)

```
READLN(M);
TERM: = 0;      MULT: = 1;
REPEAT
        TERM: = TERM + 1;
        MULT: = MULT * 2;
UNTIL (MULT > = M);
WRITELN(TERM);
```

پس از اجرای این قطعه برنامه ، مقدار نوشته شده در خروجی همواره ..... $\log_2 M$ (کارشناسی ارشد صنایع - دولتی ۸۰)

الف) بزرگتر از ب) کوچکتر از

ج) بزرگتر یا مساوی با د) کوچکتر یا مساوی با

۱۷- قطعه برنامه زیر را اجرا کنید (فرض کنید I $ J به معنی I به توان J است)

```
I    ;
```



```
PEPEAT
        I: = I + ۱
UNTIL ۲ $ I > = ۱۰۰۰
```

پس از اجرای این قطعه برنامه ، مقدار I برابر است با ..... (کارشناسی ارشد صنایع - دولتی ۸۰)

الف) ۹ ب) ۱۰ ج) ۱۱ د) ۱

۱۸- تعداد دفعاتی که (*) در قطعه برنامه زیر اجرا می شود ، از کدام مرتبه است؟ (کارشناسی ارشد صنایع - دولتی ۸۱)

```
FOR I: = ۱ TO N DO
      FOR J: = I TO N - ۱ DO
        A: = A + ۱;        (*)
```

الف) N^2 ب) N ج) N^3 د) $N \log N$

۱۹- قطعه برنامه زیر را در نظر بگیرید (فرض کنید A , B متغیرهای صحیح با مقادیر A=۲۱ و B=۶ هستند)

```
REPEAT
A: = A/۲;
B: = B - ۲ ;
UNTIL (A = B)
```

تعداد دفعات اجرای حلقه بالا کدام است؟ (کارشناسی ارشد صنایع - دولتی ۸۱)

الف) ۱۴ ب) ۱۵ ج) ۱۶ د) ۱۷

۲۰- اگر داده های ورودی ali's BOOK. باشد ، با اجرای برنامه زیر کدام عبارت چاپ می گردد؟ (کارشناسی ارشد صنایع - دولتی ۸۱)

```
Program P;
Var
    ch: = char;
Begin
      Repeat
      Read(ch);
      if (ch > = 'a') and (ch < = 'z') then
              Write (chr (ord(ch) - ord('a') + ord('A')));
       until    ch = '.';
end.
```

الف) ALIS ب) ALI'S booK ج) ALI'S BOOK د) ALI'S

۲۱- با اجرای دستورات زیر خروجی کدام گزینه خواهد بود؟ (از چپ به راست) (کارشناسی ارشد صنایع - دولتی ۸۱)

```
f1: = 0;      f2: = 1;      f: = 1;
Repeat
        write(t);
        f: = f1+f2;
        f1: = f2;
        f2: = f;
Until(f>20)
```

الف) 1,2,3,5,8,13 ب) 1,1,2,3,5,8,13,21

ج) 1,2,3,5,8,13,21 د) 1,1,2,3,5,8,13

۲۲- با اجرای برنامه زیر ، در چه صورتی برای n ، مقدار n چاپ می شود؟ (کارشناسی ارشد صنایع - دولتی ۸۱)

```
readln(n);
J: = 2;
while J<n -1 do
        if n mod J = 0 then
           halt
     else
           J: = J + 1;
if J<n then
       write (n);
```

الف) عدد فرد باشد ب) عدد کامل باشد

ج) عدد اول باشد د) عدد زوج باشد

۲۳- پس از اجرای قطعه برنامه زیر : (کارشناسی ارشد ریاضی - دولتی ۷۴)

```
T: = 1.0;
P: = 1.0 + T;
WHILE (P>1.0) DO
BEGIN
    T: = T/10.0;
    P: = 1.0 + T;
END;
```

آنگاه .....

الف) مقدار نهایی T ، صفر است.

ب) خطای زیر ریز (UNDERFLOW) ایجاد می شود.

ج) حلقه تکرار ، بدون توقف ، بی نهایت بار تکرار می شود.

د) پس از تکرار حلقه به دفعات متناهی ، مقدار P برابر 1 می شود.

۲۴- در قطعه زیر عبارتی که با * مشخص شده است چند بار اجرا می شود؟ (کارشناسی ارشد ریاضی - دولتی ۷۴)

```
FOR I: = 1 TO N DO
    FOR J: = 1 TO I DO
       X: = X + 1.0;      (*)
```

الف) N.I ب) N 2 ج) $N+N^2$ د) $\frac{N(N+1)}{2}$

۲۵- پس از اجرای قطعه برنامه زیر مقدار ذخیره شده در M کدام است؟ (کارشناسی ارشد ریاضی - دولتی ۷۴)

```
M: = 2;       K: = 1;     N: = 20;
WHILE K<N DO
   BEGIN
       M: = M + 2;
       K: = K + K
   END;
```

الف) 10 ب) 12 ج) 14 د) 32

۲۶- در قطعه برنامه زیر عبارتی که با (*) مشخص شده است ، چند بار اجرا می شود؟ (N عددی طبیعی است) (کارشناسی ارشد - دولتی ۷۵)

```
FOR I: = 1 TO N DO
    FOR J: = I TO (N-1) DO
       Y: = Y + 1.0;   (*)
```

الف) N(N-1) ب) N(N-1)/2 ج) N(N+1)/2 د) N

۲۷- پس از اجرای قطعه برنامه زیر ، گزینه صحیح در خصوص آخرین عددی که در خروجی نوشته می شود ، کدام است؟ (کارشناسی ارشد ریاضی - دولتی ۷۶)

```
EPS: = -1.0;
WHILE EPS<0.0 DO
BEGIN
      WRITELN(OUTPUT,EPS);
```

```
    EPS: = EPS/2.0
END;
```

الف) بزرگترین عدد منفی قابل نمایش در ماشین

ب) EPS بصورتی که نمایش 1-EPS در ماشین همان عدد 1 است

ج) کوچکترین عدد منفی قابل نمایش در ماشین

د) هیچکدام از موارد فوق

۲۸- با توجه به عبارات زیر چه چیزی در خروجی نوشته می شود؟ (کارشناسی ارشد ریاضی - دولتی ۷۷)

```
A: = 10;
B: = 5;
C: = 6;
D: = 11;
    if A<B then if C<D then write('I')
                       else write('2)
          else if C<D then write('3')
    else write('4');
```

الف) 1 ب) 2 ج) 3 د) 4

۲۹- در قطعه برنامه زیر دستور WRITE چند بار اجرا می شود؟ (کارشناسی ارشد ریاضی - دولتی ۷۸)

```
FOR I: = 0 TO 100 DO
FOR J: = I + 1 TO 101 DO
WRITE (I,J);
```

الف) 4949 ب) 5050 ج) 5151 د) 9494

۳۰- دستور IF و دستور WRITE در برنامه زیر به ترتیب چند بار اجرا می شوند؟ (کارشناسی ارشد ریاضی - دولتی ۷۸)

```
FOR I: = 1 TO 10 DO
BEGIN
FOR J: = 1 TO N DO
BEGIN
FOR K: = I TO 10 DO
BEGIN
IF (I=J) AND (J=K) THEN
    WRITE(I,J,K);
END
END
```



```
END;
```

الف) ۱۰۰ , ۵ ب) ۱۱۰ , ۵ ج) ۲۲۰ , ۱۰ د) ۱۰۰۰ , ۱۰

۳۱- قسمتی از یک برنامه به زبان پاسکال بصورت زیر است : (کارشناسی ارشد ریاضی - دولتی ۷۸)

```
M: = ۱;      K: = ۲;      N: = ۱۱;
Repeat begin
    Writeln(M);
    M: = M + K;
End
Until M>=N
```

با اجرای دستورالعملهای فوق مقادیر نوشته شده در خروجی را تعیین کنید.

الف) اعداد فرد از ۱ تا ۹ ب) اعداد فرد از ۱ تا ۱۱

ج) همه اعداد صحیح از ۱ تا ۹ د) همه اعداد صحیح از ۱ تا ۱۱

۳۲- خروجی برنامه زیر کدام است؟ (کارشناسی ارشد ریاضی - دولتی ۷۹)

```
X: = ۲;
for I: = X div X to sqr(X) do
for j: = i div X downto I mod X do
X: = succ(X);
writeln(X);
```

الف) ۶ ب) ۱۶ ج) ۱۲ د) ۸

۳۳- قطعه برنامه زیر را مطالعه کنید و گزینه صحیح برای مقدار TOTAL را تعیین کنید. (فرض کنید که N یک عدد صحیح مثبت و توانی از ۲ است.) (کارشناسی ارشد ریاضی - دولتی ۷۹)

```
TOTAL: = ۰;
WHILE(N<>۰) DO
Begin
    N: = N DIV ۲;
    TOTAL: = TOTAL + ۱
End;
```

الف) log N ب) log N-۱ ج) log N+۱ د) log (N+۱)

۳۴- اگر k متغیری اعشاری و حاوی عدد 14 باشد ، خروجی تکه برنامه زیر چه می‌شود؟

```
case int(k) of
      3: write ('3');
      4: write ('4');
end;
```

الف) پیام خطا صادر می شود ب) 3 ج) 4 د) 34

۳۵- خروجی قطعه برنامه زیر چه می باشد؟

```
M:=0;
N:= 8570;
While  N>0 do
    Begin
          T:= N mod 10;
          N:= N div 10;
          M:= M*10+T;
    End;
Writeln(M);
```

الف) ۸۵۷۰ ب) ۷۵۸۰ ج) ۸۵۷ د) ۷۵۸

۳۶- اگر M یک عدد صحیح و مثبت باشد جزء برنامه زیر چه مقدار چاپ می کند؟

```
S:= 0;
While  M>0 do
    Begin
        S:= S + M mod 10;
        M:= M div 10
    End;
    WRITE(S)
End.
```

الف) 0 ب) مجموع ارقام M ج) عدد M د) تعداد ارقام M

۳۷- اگر قسمتی از یک برنامه پاسکال به فرم زیر اجرا گردد چه چاپ خواهد شد؟

```
a := 4234;
b := 10;
while (a<>4) do
    begin
        d := a mod b;
        write(d:1);
        a := a div b;
    end;
```


الف) 4234 ب) 423 ج) 4324 د) 432

۳۸- قطعه برنامه زیر را اجرا کنید (فرض کنید J $ I به معنی I به توان J است)

```
I: = 1;
PEPEAT
      I: = I + 1
UNTIL 2 $ I>= 1000
```

پس از اجرای این قطعه برنامه ، مقدار I برابر است با ..... (کارشناسی ارشد صنایع - دولتی ۸۰)

الف) 9 ب) 10 ج) 11 د) 1

۳۹- تعداد دفعاتی که (*) در قطعه برنامه زیر اجرا می شود ، از کدام مرتبه است؟ (کارشناسی ارشد صنایع دولتی ۸۱)

```
FOR I: = 1 TO N DO
      FOR J: = I TO N - 1 DO
        A: = A + 1;        (*)
```

الف) N^2 ب) N ج) $N \log N$ د) $N \log^2 N$

۴۰- قطعه برنامه زیر را در نظر بگیرید (فرض کنید A , B متغیرهای صحیح با مقادیر $A \leq 2^6$ و $B = 2$ هستند)

```
REPEAT
A: = A/2;
B: = B - 2 ;
UNTIL (A=B)
```

تعداد دفعات اجرای حلقه بالا کدام است؟ (کارشناسی ارشد صنایع دولتی ۸۱)

الف) 14 ب) 15 ج) 16 د) 17

۴۱- اگر داده های ورودی ali' s BOOK. باشد ، با اجرای برنامه زیر کدام عبارت چاپ می گردد؟ (کارشناسی ارشد صنایع - دولتی ۸۱)

```
Program P;
Var
    ch: = char;
Begin
      Repeat
      Read(ch);
```

```
    if (ch >= 'a') and (ch <= 'z') then
        Write (chr (ord(ch) - ord('a') + ord('A')));
    until   ch = '.';
end.
```

الف) ALIS ب) ALI'S booK ج) ALI'S BOOK د) ALI'S

۴۲- با اجرای دستورات زیر خروجی کدام گزینه خواهد بود؟ (از چپ بـه راسـت) (کارشناسی ارشد صنایع - دولتی ۸۱)

```
f1: = 0;    f2: = 1;    f: = 1;
Repeat
        write(t);
        f: = f1+f2;
        f1: = f2;
        f2: = f;
Until(f>20)
```

الف) 1,2,3,5,8,13 ب) 1,1,2,3,5,8,13,21

ج) 1,2,3,5,8,13,21 د) 1,1,2,3,5,8,13

۱ برنامه زیر چه عددی را نشان می دهد ؟

```
Var
   a,b,i : integer;
    n: arry [1.. 10] of integer;
begin
   a: = 12;    b: = 2;    i: = 1;
   while a<>0 do begin
          n[i]: = (a mod b);
          a: = a div b;
          inc (i);
  end;
  writeln;
  for i: = i –1 downto 1 do write(n[i]);
end.
```

الف) 0011 ب) 136 ج) 631 د) 1100

۲- کدام گزینه زیر برای مشخص کردن نمرات ۵۰ دانشجو که هـر دانـشجو ۹ درس را امتحان داده صحیح می باشد؟

الف) STUDENT:Arry[50,9] ب) STUDENT:Array [50,9], Real

ج) STUDENT:Array [1..50,1..9] of Real د) STUDENT:array[9..50] of Real

۳- آرایه با تعریف زیر چند عضو دارد؟

```
Var   A: Array[0..2,2..3,3..4,4..5,5..6,6..7,7..8,8..9] of Integer;
```

الف) ۳۸۴ عضو ب) ۷۶۸ عضو

ج) ۱۹۲ عضو د) این آرایه قابل ایجاد نیست

۴- تعریف زیر چه نوعی ایجاد می کند؟

```
Var
    C: array [char] of 1..100;
```

الف) آرایه های با اندیس شماره کاراکترها ایجاد می کند که با مقادیر بین ۱ تـا ۱۰۰ پـر می شوند

ب) این تعریف اشتباه است

ج) آرایه ای با اندیس کاراکترهای اسکی ایجاد می شود که مقدار هـر عـضو بـین ۱ تـا ۱۰۰ می تواند باشد

د) آرایه ای ۱۰۰ خانه با اعضای از نوع کاراکتر ایجاد می کند

۵- آرایه تعریف شده بصورت Var c:Array[Boolean] of 1 .. 5; چه نوع آرایه ای است؟

الف) ۲ عضوی از نوع زیر محدوده ب) ۵ عضوی از نوع بولی

ج) ۱ تا ۵ عضوی از نوع بولی د) این معرفی معتبر نیست و خطا دارد

۶- آرایه X:Array[-7..5] of word; چند عضو دارد؟

الف) 2 ب) 5 ج) 12 د) 13

۷- با توجه به اینکه متغیر W بصورت Var W:Array['a'..'z'] of 'A'.. 'M'; تعریف شده است ، کدام مورد صحیح است؟

الف) W[z]: = 'A' ب) W['x']: = 'I' ج) W['B']: = 'A' د) W['a']: = 'a'

۸- در صورت صحت تعریف زیر ، چند بایت حافظه اشغال می شود؟

```
Var   Auto: Array [1..2] of Array [1..2] of Array [1..2] of
      Array [1..2] of Array [1..2] of array [1..2] of Real;
```

الف) ۶۴ بیت

ب) ۳۸۴ بایت

ج) در پاسکال فقط می توان آرایه تا چهار بعدی تعریف کرد و این تعریف مجاز نیست

د) ۲۵۶ بایت

۹- آرایه A: Array [-3..7] چند بایت در حافظه اشغال می کند؟

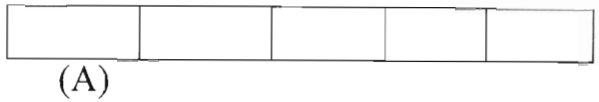
الف) 4A ب) 16 ج) 40 د) 66

۱۰- در جستجوی خطی آرایه باید به چه صورت باشد؟

الف) نزولی مرتب باشد ب) صعودی مرتب باشد

ج) به هر صورت که باشد د) مرتب شده باشد (چه صعودی و چه نزولی)

۱۱ اگر آرایه بصورت زیر باشد در برنامه زیر Swap چند بار تکرار می شود؟



الف) 7 ب) 3 ج) 4 د) 5

۱۲- در روش مرتب سازی انتخابی برای ۸ عنصر از یک آرایه ، در حالت عادی چند عمل مقایسه انجام می شود؟

الف) 49 ب) 20 ج) 28 د) 64

۱۳- قطعه برنامه زیر را در نظر بگیرید: (کارشناسی ارشد صنایع - دولتی ۷۹)

```
FOR I: = 1 TO N DO
FOR J: = 1 TO N DO
   BEGIN
        IF (I<>J) THEN A[I,j]: = 1;
      IF (I = J) THEN A[I,J]: = 1;
      ELSE A(I,J): = 0;
END
```

پس از اجرای این قطعه برنامه ، ماتریس A عبارت است از ماتریسی .....

الف) همانی ب) با همه درایه ها برابر با 1

ج) با همه درایه ها برابر با 0 د) پایین مثلثی با درایه های برابر با 1

۱۴- فرض کنید P(I) = I , I = 1,2,...,100 در درست است. تعداد دفعاتی که حلقه زیر اجرا می شود، چقدر است؟ (کارشناسی ارشد صنایع - دولتی ۷۹)

```
I: = 1;
WHILE (P[I] < 50) DO
BEGIN
      P[I+1]: = 4* P[I];
      I: = I + 1
END;
```

الف) 3 ب) 5 ج) 12 د) 15

۱۵- پس از اجرای قطعه برنامه زیر ، گزینه صحیح را انتخاب کنید. (کارشناسی ارشد صنایع - دولتی ۷۹)

```
FOR J: = 1 TO M DO
   BEGIN
         I: = M – J +1;
         IF (Y[I] <> X[J]) THEN Y[I]: = X[J]
END;
```

الف) Y = X

ب) X(J) = Y(J) , J = 1,...,M

ج) درایه های Y مرتب شده درایه های X هستند

د) درایه های Y همان درایه های X در جهت عکس هستند

۱۶- قطعه برنامه زیر را اجرا کنید (فرض کنید ماتریس A , N*N است) و گزینه صحیح را در خصوص ماتریس A انتخاب کنید. (کارشناسی ارشد صنایع - دولتی ۷۹)

```
FOR I: = 1 TO N-1 DO
   A[I-1,I]: = A[I,I-1];
```

الف) درایه های متناظر بالای قطر اصلی و زیر قطر اصلی A جا به جا می شوند.

ب) درایه های قطر فرعی A در درایه های متناظر قطر اصلی قرار می گیرند.

ج) درایه های بالای قطر اصلی A در درایه های متناظر زیر قطر اصلی قرار می گیرند.

د) درایه های زیر قطر اصلی A در درایه های متناظر بالای قطر اصلی قرار می گیرند.

۱۷- قطعه برنامه زیر را در نظر بگیرید: (کارشناسی ارشد صنایع – دولتی ۸۰)

```
FOR I: = 1 TO M DO
      FOR J: = I TO M DO
            IF (I>J) THEN
               BEGIN
                     A[I,J]: = 1.0;
                     A[J,I]: = A[I,J];
               END
            ELSE BEGIN A[I,J] = 0.0;       A[J,I]: = 0.0     END;
```

پس از اجرای این قطعه برنامه ، ماتریس A برابر با ماتریس .....

الف) همه درایه ها برابر صفر

ب) درایه های قطری برابر صفر و بقیه درایه ها یک

ج) درایه های قطری برابر صفر و درایه های زیر قطری برابر یک

د) متقارن با حداقل یکی از درایه های غیرقطری برابر یک

۱۸- فرض کنید P(I) = I , I = 1,2,...,10 در دست است. قطعه برنامه زیر را در نظر بگیرید:

```
I: = 1;
REPEAT
            P[I+1]: = P[I]*P[I] + P[I+1]
            I: = I + 1
UNTIL (P[I] > = 1000) OR (I > = 10)
```

پس از اجرای این قطعه برنامه ، مقدار I برابر است با ..... (کارشناسی ارشد صنایع – دولتی ۸۰)

الف) ۴ ب) ۵ ج) ۶ د) ۱۰

۱۹- قطعه برنامه زیر را اجرا کنید (فرض کنید B ، N*N , N بزرگتر از 2 است) :

```
FOR I: = 1 TO N-1 DO
    B[I,I+1]: = B[I+1,I];
```

الف) پس از اجرای این قطعه برنامه ، ..... (کارشناسی ارشد صنایع – دولتی ۸۰)

الف) درایه های قطر فرعی B در درایه های قطر اصلی B قرار می گیرند.

ب) درایه های متناظر بالای قطر اصلی و زیر قطر اصلی B جابجا می شوند.

ج) درایه های متناظر زیر قطر اصلی در درایه های متناظر بالای قطر اصلی B قرار می گیرند.

د) درایه های متناظر بالای قطر اصلی در درایه های متناظر زیر قطر اصلی B قرار می گیرند.

۲۰- قطعه برنامه زیر را اجرا کنید (فرض کنید N یک عدد صحیح مثبت است) :

```
FOR I: = N DOWNTO 1 DO
    BEGIN
          J: = N-1 + 1;
          IF (A[I] < > B[I]) THEN B[I]: = A[I];
          ELSE B[J]: = A[J]
    END;
```

پس از اجرای این قطعه برنامه ، ..... (کارشناسی ارشد صنایع – دولتی ۸۰)

الف) B مرتب شده A است.

ب) A[I] = B[I] , I = 1, ...,N, B[I] = A[I]

ج) ارتباط مشخصی بین درایه های A , B وجود ندارد.

د) مقادیر درایه های B برابر مقادیر درایه های A در جهت عکس هستند.

I

۲۱- فرض کنید P یک بردار صحیح است با P[I] = 2 برای I = 1,2, ...,10. تعداد دفعات اجرای (*) درحلقه زیر برابر است با ..... . (کارشناسی ارشد صنایع - دولتی ۸۱)

```
I: = 1;
WHILE P[I] < > P[I+1] DO
Begin
        P[I+2]: = P[I+2] – P[I+1];          (*)
        I: = I +1
end;
```

الف) ده ب) دو ج) یک د) صفر

۲۲- با توجه به برنامه مقابل مقادیر ماتریس K کدام است؟ (فرض کنید A یک ماتریس N*N باشد.) (کارشناسی ارشد صنایع - دولتی ۸۱)

```
FOR i: = 1 TO N DO
Begin
     FOR j: = 1 TO N DO
          Begin
               K[i+j]: = 0;
              IF (i+j) = N+1 then K[i,j]: = 1;
          end;
end;
```

الف) درایه های ماتریس همگی یک هستند.

ب) درایه های ماتریس همگی صفر هستند.

ج) درایه های قطر اصلی یک و بقیه درایه ها صفر است.

د) درایه های قطر غیراصلی یک و بقیه درایه ها صفر است.

۲۳- قطعه برنامه زیر را در نظر بگیرید: (کارشناسی ارشد صنایع – دولتی ۸۱)

```
FOR I: = 1 TO N DO
begin
    A[I,I]: = 1.0;
    FOR J: = 1      TO   N   DO
          A[I,J]: = 0.0;
end;
```

پس از اجرای این قطعه برنامه ، ماتریس A با کدام ماتریس برابر است؟

الف) همانی

ب) با درایه ها همگی برابر صفر

ج) با عدد یک روی ستون اول و صفر روی بقیه درایه ها

د) با عدد یک روی سطر اول و صفر روی بقیه درایه ها

۲۴- حاصل اجرای برنامه زیر در مورد ماتریس M چیست؟ (کارشناسی ارشد ریاضی- دولتی ۷۴)

```
FOR I: = 1 TO 10 DO
BEGIN
     FOR J: = 1 TO 10 DO
     BEGIN
    M[I,J]: = 0;
    IF (I+J) = 11 THEN
        M[I,J]: = 1
    END;
END;
```

الف) ماتریس M ، ماتریس صفر است.

ب) عناصر قطر غیراصلی یک و بقیه عناصر صفر است.

ج) عناصر ستون اول ماتریس M برابر صفر و بقیه عناصر ۱ است.

د) عناصر سطر اول ماتریس M برابر ۱ و بقیه عناصر صفر است.

۲۵- آرایه K[I] , I = 1,...,20 با ضابطه K[I] = I مفروض است. به ازای چه مقدار I حلقه زیر خاتمه می یابد؟ (کارشناسی ارشد ریاضی – دولتی ۷۴)

```
I: = 2;
WHILE K[I] < 20 DO
   BEGIN
        K[I]: = K[I-1]*3;
        I: = I +1;
   END;
```

الف) ۴ ب) ۵ ج) ۱۹ د) ۲۰

۲۶- فرض کنید A یک ماتریس N*N باشد. پس از اجرای قطعه برنامه زیر مقدار ذخیره شده در T کدام است؟ (کارشناسی ارشد ریاضی - دولتی ۷۵)

```
T: = 0.0;
FOR I: = 1 TO N-1 DO
T: = T + A[I,N-I] * A[I+1,I];
```

الف) ضرب داخلی بردار زیر قطر اصلی A و بردار قطر فرعی A

ب) ضرب داخلی بردار زیر قطر اصلی A و بردار زیر قطر فرعی A

ج) ضرب داخلی بردار بالای قطر اصلی A و بردار بالای قطر فرعی A

د) ضرب داخلی بردار بالای قطر اصلی A و بردار زیر قطر فرعی A

۲۷- پس از اجرای قطعه برنامه زیر ، که در آن ماتریس M*N و X بردار مفروض است:

```
FOR I: = 1 TO M DO
Begin
    FLAG: = TRUE;
   FOR J: = 1 TO N DO
          IF (X[J] < > A[I,J])
          THEN FLAG: = FALSE
End;
```

در متغیر FLAG مقدار نادرست (FALSE) تنها وقتی می گیرد که ..... (کارشناسی ارشد ریاضی - دولتی ۷۵)

الف) سطر M ام A مساوی X باشد.

ب) سطر M ام A مخالف X باشد (کل خانه ها مخالف باشند).

ج) همه سطرهای A مخالف X باشند.

د) درایه های از سطر M ام A با درایه نظیرش از X مخالف باشد.

۲۸- پس از اجرای قطعه برنامه زیر (کارشناسی ارشد ریاضی - دولتی ۷۵)

```
FOR I: = 1 TO N DO
begin
    J: = N-1 + 1;
    IF (Y[J] < > X[I]) THEN Y[J]: = X[I]
end;
```

الف) $X(I) = Y(I)$, $I = 1,\ldots,N$

ب) $Y(I) = X(N+1-I)$, $I = 1,\ldots,N$

ج) درایه های Y مرتب شده درایه های X هستند.

د) درایه های X به ترتیب در درایه های Y قرار گرفته اند.

۲۹- برنامه زیر چه عملی روی ماتریس A انجام می دهد (A ماتریسی N*N و مفروض است)؟ (کارشناشی ارشد ریاضی - دولتی ۷۵)


```
    Sum: = 0.0;
FOR I: = 1 TO N
begin
        FOR J: = I TO N
        Sum: = Sum + A[I,J]
end;
```

الف) درایه های روی قطر اصلی و همه درایه های مثلث بالای قطر اصلی A را جمع می کند.

ب) مجموع درایه های مثلث زیر قطر اصلی A را به دست می آورد.

ج) مجموع درایه های قطر اصلی و همه درایه های مثلث زیر قطر اصلی A را به دست می آورد.

د) همه درایه های مثلث بالای قطر اصلی ماتریس A را جمع می کند.

۳۰- بخشی از یک برنامه بصورت زیر را اجرا کنید و نتیجه بدست آمده را برای ماتریس M*M , A تعیین کنید. (کارشناسی ارشد ریاضی - دولتی ۷۶)

```
FOR I: = 1 TO M DO
FOR J: = 1 TO M DO
BEGIN
IF (I<>J) THEN A[I,J]: = 0.0;
        A[I,J]: = 1.0
END;
```

الف) یک ماتریس همانی ب) یک ماتریس با همه درایه های مساوی صفر

ج) یک ماتریس با همه درایه های مساوی یک د) هیچکدام از موارد فوق

۳۱- در برخی از زبانهای برنامه نویسی امکان استفاده از اندیس های منفی برای بردارها وجود ندارد. آیا راه حل هایی را که اندیس های منفی به کار می گیرند ، می توان به صورت برنامه أی معادل در هر زبان برنامه نویسی نوشت؟ (کارشناسی ارشد ریاضی - دولتی ۷۹)

الف) بستگی به کامپیوتر مورد استفاده دارد ب) تنها در برخی زبانهای برنامه نویسی امکان دارد.

ج) تنها گاهی اوقات می توان چنین کرد د) همواره با تغییراتی مناسب می توان چنین کرد

۳۲ - متغیرهای زیر چند بایت فضا اشغال می کنند؟

```
x: Array [boolean , char] of word;
y: Array [boolean] of Array [5..10] of Array [True .. True] of shortint;
```

الف) بایت x = 512 و بایت y = 24 ب) بایت x = 256 و بایت y = 36

ج) بایت x = 1024 و بایت y = 12 د) تعاریف فوق نادرست است

۳۳- کدام تعریف در قسمت Var درست است؟

```
i: Array [char] of Boolean;
```

الف) i [char (false)]: = true; ب) i [chr(3)]: = false;

ج) i [char(3)]: = false; د) هر سه گزینه

۱ مقدار عبارت زیر چند است؟

```
5+2/8*3-sqrt(8/2)+2 mod Round(6 div 4)
```

الف) 0.375- ب) 3.75 ج) 4.75 د) 5.25

۲- تکه برنامه زیر چه عددی را چاپ می کند؟

```
writeln (SQRT (ABS(-4)));
```

الف) خطا تولید می شود ب) 2- ج) 2 د) 4-

۳- کار دستور PRED('C') کدام است؟

الف) کد اسکی علامت C را برمیگرداند

ب) علامت قبل از حرف C را برمیگرداند

ج) علامت بعدازحرف C را برمیگرداند

د) علامت مربوط به کد اسکی C را برمیگرداند

۴- خروجی تکه برنامه زیر کدام است؟

```
writeln (CHR (ORD(UPCASE('a'))));
```

الف) A ب) 64 ج) a د) 96

۵- نوع خروجی کدام تابع نظیر نوع ورودی آن است؟

الف) SQR ب) SQRT ج) ODD د) TRUNC

۶- خروجی دستور روبرو کدام است؟

```
Writeln (trunc(EXP(3*Ln(2))));
```

الف) 6 ب) 5 ج) 8 د) 9

۷- نوع پارامتر ورودی و خروجی تابع odd(x) چیست؟

الف) اعشاری ، منطقی ب) اعشاری ، اعشاری

ج) صحیح ، منطقی د) صحیح ، صحیح

۸ - عملکرد دستورالعمل Round(x) معادل با کدام مورد زیر است؟

الف) از دستور Trunc می توان بجای Round نیز استفاده کرد

ب) برای X+ معادل با Trunc(X+0.5) و برای x- معادل Trunc(X-0.5) است

ج) این دستور معادل ندارد زیرا کار آن با دستورات دیگر متفاوت است

د) اگر مقدار اعشار X بزرگتر از 0.5 باشد فقط می توان از دستور Trunc استفاده کرد

۹- کدامیک از توابع پاسکال می تواند عدد 6.6 را به عدد 6 تبدیل نماید؟

الف) odd ب) round ج) trunc د) abs

۱۰- کدامیک از جملات زیر مقدار ASCII از حرف 'A' را چاپ می نماید؟

الف) writeln(Ch('A')) ب) writeln(pred('A'))

ج) writeln(succ('A')) د) writeln(ord('A'))

۱۱- نتیجه عبارت FRAC(99.42) کدام است؟

الف) 0.42 ب) 99 ج) 100.0 د) 99.4

۱۲- در عبارت زیر نتیجه W کدام است؟

```
If x>0 then    w: = trunc(x)   else   w: = trunc(x)-1
```

الف) |x| ب) [x] ج) x د) [x]-x

۱۳- اگر a = -2.6 باشد ، آنگاه مقدار عبارت زیر چه خواهد بود؟

```
trunc(a) + Round(a) + Int(a) + (a-2)
```

الف) -7.00 ب) -7 ج) -7.6 د) خطا دارد

۱۴- پس از اجرای دستور زیر مقدار x کدام خواهد بود؟

```
x: = Pred(3.5);
```

الف) 2 ب) 3 ج) 4 د) از تابع Pred استفاده غیرمجاز شده است

۱۵- مقدار عبارت زیر را مشخص نمائید؟


```
6+3/2*2+9 mod round(8 div 6)
```

الف) 10 ب) 6.75 ج) 9 د) 9.75

۱۶- در قطعه برنامه زیر فرض کنید که CH یک حرف انگلیسی است.

```
T: = ORD('A') – ORD('a');
IF (CH>='A') AND (CH<='Z') THEN CH: = CHR(ORD(CH)-T);
IF (CH>='a') AND (CH<='z') THEN CH: = CHR(ORD(CH)+T);
```

الف) پس از اجرای این قطعه برنامه ، CH برابر است با ..... (کارشناسی‌ارشد صنایع- دولتی ۸۰)

الف) یک حرف غیرالفبایی

ب) حرف الفبایی بزرگ متناظر با مقدار اولیه CH

ج) حرف الفبای کوچک متناظر با CH اگر CH اولیه حرف الفبای بزرگ باشد

د) حرف الفبای کوچک اگر CH اولیه حرف بزرگ و حرف الفبای بزرگ اگر CH اولیه حرف الفبای کوچک باشد

۱۷- اگر x = TRUE و y = FALSE باشند ؛ خروجی دستور زیر چه خواهد بود؟

```
write(Succ(x<y), '    ', Pred (y<-x));
```

الف) TRUE TRUE ب) FALSE FALSE

ج) TRUE FALSE د) FALSE TRUE

۱۸- خروجی برنامه زیر کدام است؟

```
F:= 2;
FOR C:= 1 TO 20 DO     F:= SQRT(F) + 2
F:= F – 2;
WRITELN(F:2:2);
```

الف) 0 ب) 2.00 ج) 4.00 د) OVERFLOW

۱۹- خروجی این تکه برنامه چیست؟ x , y متغیرهای صحیح و k متغیر کاراکتری می‌باشند.

```
for  x: = 3 to 4 do
begin
   write (x-1);
   for k:  'c' to 'd' do
   begin
         for y: = 5 to 6 do;
             write (y);
        write(pred(k));
    end;
end;
```

الف) 356b56c456b56c ب) 26b6c36b6c

ج) 256b56c356b56c ج) برنامه خطا دارد

۱- پس از تعریف ;'123' = Const S:String محتوای S[0] کدام است؟

الف) #1 ب) 3 ج) #3 د) موجود نمی باشد

۲- در صورت اجرای دستورات زیر چه عملی انجام می شود؟

```
Var   S:String[5];
Begin
   S[1] = 'a';   S[2]:= 'b'      S[3]:= 'c'
   Writeln(S);
End.
```

الف) عملی انجام نمی شود ، چون طول S صفر است

ب) کلمه abc روی صفحه چاپ می شود

ج) کلمه abc بعلاوه یک حرف دیگر که در اندیس صفر قرار دارد و نامشخص است

د) این برنامه اشتباه است ، زیرا نمی توان با رشته ها در پاسکال ، مانند آرایه کار کرد

۳- خروجی برنامه زیر چیست؟

```
Var    K: string;
Begin
   K: = 'B';
  write(pred(K));
End.
```

الف) 65 ب) A ج) 'A' د) خطای کامپایلری دارد

۴- کدام عدد بعد از اجرای برنامه زیر نمایش داده می شود؟

```
uses crt;
var s:string;
begin
     Clrscr;
     S: = '123.576';
     S[0]: = #5;
     write(S);
end.
```

الف) 5 ب) 123 ج) 123.5 د) 123.57

۵- خروجی برنامه زیر چاپ کدام مورد است؟

```
Var     s: string;
        i: integr;
begin
        for i: = 6 to 10 do
```

```
        S[i-5]  chr (i159);
    S[0]:  #5;
    write(s);
end.
```

الف) اعداد ۶۵ تا ۶۹ ب) عبارت ABCDE ج) عبارت ABCD د) عدد ۵

۶- جملات به زبان پاسکال را انتخاب کنید که یک آرایه packed با ۱۵ عنصر تعریف نماید به نام word و یک متغیر به نام myword از نوع word تعریف نمایید و سپس ورودی صفحه کلید را در این متغیر ارزش دهی نمایید.

الف)

```
type word = packed array [1..15] of char;
var word : myword;
begin
   readln(myword);
```

ب)

```
type word = packed array [1..15] of integer;
var myword : word;
begin
  readln(myword);
```

ج)

```
type word = packed array [1..15] of char;
var myword : word;
begin
   readln(myword);
```

د)

```
type word = packed array [1..15] of char;
var myword : word;
begin
  readln(word);
```

۷- از برنامه زیر :

```
program q2;
var
    x: array [1..2] of char;
begin
    x[1]: = 'A';
  x[2]: = 'B';
   x[3]: = 'C';
    writeln(x[1], x[2], x[3]);
end.
```

الف) اشتباهی گرفته نمی شود و برنامه به نحو صحیح کار می کند.

ب) اشتباهی گرفته نمی شود ولی ممکن است نتیجه درست نباشد.

ج) یک اشتباه دستوری در زمان ترجمه گرفته می شود.

د) یک اشتباه منطقی در زمان اجرا گرفته می شود.

۸- خروجی برنامه زیر چیست؟

```
VAR   T: STRING;
      I: BYTE;
BEGIN
CLRSCR;
      T: = 'AB CDE';
      I: = LENGTH(T);
      WHILE T[I] = CHR(32) DO DEC(I);
      WHILE T[I] <> CHR(32) DO
          BEGIN
              WRITE(T[I]);
              DEC (I);
          END;
END.
```

الف) 'DEC' ب) 'CDE' ج) 'CED' د) 'EDE'

۹- خروجی برنامه زیر کدام است؟

```
StrName1: = 'Ali';
StrName2: = 'Reza';
For i: = 1 to 4 do
     StrName1[i+3]: = StrNam2[I];
write(StrNam1);
```

الف) Ali ب) Alireza ج) Ali Reza د) Reza

۱۰- نتیجه کدام عبارت بولی برابر TRUE است؟

الف) 'TEST> 'test' ب) 'TEST' < 'test'

ج) TEST >= 'test' د) 'TEST' = 'test'

۱۱- خروجی برنامه زیر کدام است؟

```
StrNume1: = '4';
StrNume2: = '2';
StrNume1: = StrNum1 + StrNum2;
write(Strnum1);
```

الف) 6 ب) 24 ج) 42 د) غیرمجاز

۱۲ پس از اجرای قطعه برنامه زیر چه نتیجه ای برای ماتریس مربع M*M بدست می آید ، یک ماتریس ..... (کارشناسی ارشد ریاضی - دولتی ۷۸)

```
VAR K,L,X,Y : INTEGER;
M: ARRAY[1..10,1..10] OF INTEGER;
FOR K: = 1 TO 10 DO
FOR L: = 1 TO 10 DO
BEGIN
      X: = TRUNC(L/K);
      Y: = TRUNC(K/L);
      M[K,L]: = X*Y;
END;
```

الف) بالا مثلثی با درایه های مساوی یک ب) پایین مثلثی با درایه های یک

ج) سه قطری با درایه های مساوی یک د) همانی

۱۳- اگر P متغیری رشته ای باشد ، خروجی تکه برنامه زیر چیست؟

```
P: = 'Alireza';
P[0]: = chr(4);
if ord(P[0]) = Length (P) then writeln(P)
else writeln ('ERROR');
```

الف) Alir ب) Alireza ج) ERROR د) خطای کامپایلری دارد

۱۴- نتیجه اجرای حلقه زیر بر روی رشته St کدام است؟

```
While St [length(st)] = ' ' DO
    Delete (St, Length(St), 1);
```

الف) حذف کاراکترهای space از ابتدای رشته St

ب) حذف کاراکترهای space از انتهای رشته St

ج) حذف یک کاراکتر space از انتهای رشته St

د) حذف و پاک کردن تمام کاراکترهای رشته St

۱۵- اگر CH از نوع CHAR باشد جزء برنامه زیر چه کاری انجام می دهد؟

```
While  NOT  EOLN do begin
     READ(CH);
     If (CH >= 'A') AND (CH <= 'Z') then
        CH:= CHR(ORD (CH) – ORD ('A) + ORD ('a'));
WRITE(CH)
End;
```

الف) یک خط از متن را می خواند و می نویسد.

ب) یک خط از متن را می خواند و با حروف بزرگ می نویسد.

ج) یک خط از متن را می خواند ، فقط حروف آن را می نویسد.

د) یک خط از متن را می خواند و با حروف کوچک می نویسد.

۱ خروجی برنامه زیر کدام است؟

```
procedure m( x:integer; var y:integer);
begin
      x: = x+y;    y: = x-y;      x: = y*2;
end;
var x,y : integer;
begin
      x: = 5;      y: = 2;          m(x,y);
      write(x, ",y);
end.
```

الف) 5 5 ب) 2 4 ج) 2 2 د) 4 2

۲- خروجی تابع زیر اگر n = 4 فرض شود کدام است؟

```
Function Loop (n:Integer) : word;
Begin
       if n <= 2 then Loop: = 1
       else Loop: = Loop(n-1) + Loop(n-2);
End;
```

الف) 2 ب) 3 ج) 4 د) 5

۳- اگر x = 64 باشد ، خروجی تابع روبرو کدام است؟

```
Function sqp (x:word):word;
Var c,s,n:word;
Begin
    c: = 1;      s: = 0;    n: = 0;
   while s<x do
   Begin
    s: = s+c;
    inc(n);
    c: = c+2;
 End;
 sqp: = n;
End;
```

الف) 8 ب) 16 ج) 64 د) 1320

۴- اگر S: = '2135' , m: = 3 باشد خروجی تابع زیر کدام است؟

```
Function ver (s:string; m:Byte): string
Begin
       If m = 1 then ver: = S[1]
        else ver: = S[m] + ver(s,m-1);
End;
```

الف) 135 ب) 213 ج) 312 د) 531

۵ خروجی برنامه زیر کدام است؟

```
Function x(var a:Integer; b:Integer): Integer;
Begin
       a: = a*b;
       b: = a+b;
       x: = a;
End;
Var a,b,c : Integer;
Begin
       a: = 2;     b: = 3;      c: = 4;
       b: = x(a,c);
       writeln(a:3, b:3, c:3);
End.
```

الف) 8 12 4 ب) 4 8 8 ج) 8 12 4 د) 8 8 12

۶- در صورت اجرای برنامه زیر چه عملی انجام می شود؟

```
program test;
Var   i:integer;
procedure F1;
begin
        i: = i-2;
       F2;
end;
procedure F2;
begin
        i: = i+3;
end;
begin
     i: =1;
     F1;
  writeln(I);
end.
```

الف) برنامه دارای خطایی برای نحوه تعریف F2 بوده و اجرا نمی شود.

ب) برنامه دارای خطای محل تعریف i بوده و اجرا نمی شود.

ج) مقدار متغیر i که برابر 2 است چاپ می شود.

د) مقدار متغیر i که برابر 1 است چاپ می شود.

۷- خروجی پردازه زیر چیست؟

```
program sample (output);
var x,y :integer;
procedure godoit (x,y :integer);
    begin
       x: = y;          y: = 0;
```

```
    write(x,y);
  end;
begin
  x: = 1;    y:  2;    godoit (x,y);    writeln(x,y);
end.
```

الف) 2001 ب) 1020 ج) 2012 د) 1234

۸- برنامه زیر چه کاری انجام می دهد؟ (جوابی که به نظر صحیح تر است انتخاب کنید.)

```
PROCEDURE BLE(L:INTEGER;  VAR A:A1);
VAR PASS, I , TEMP: INTEGER;
      SWAP: BOOLEAN;
BEGIN
PASS: = 1;   SWAP: = TRUE;
WHILE (PASS<L) AND SWAP DO
BEGIN
     SWAP: = FALSE;
     FOR I: = 1 TO L-PASS DO
           IF A[I+1] < A[I]
           THEN BEGIN
               TEMP: = A[I];      A[I]: = A[I+1];
               A[I+1]: = TEMP;  SWAP: = TRUE
            END
END;
PASS: = PASS +1
END;
```

الف) عناصر آرایه A را با روش BUBBLE بصورت صعودی مرتب می کند.

ب) عناصر آرایه A را با روش INSERTION بصورت صعودی مرتب می کند.

ج) عناصر آرایه A را با روش BUBBLE بصورت نزولی مرتب می کند.

د) مقادیر هر زوج عنصر A(I) , A(I+1) را با یکدیگر مقایسه می کند و اگر اولی از دومی بزرگتر باشد ، جای آن دو را با هم عوض می کند.

۹- اگر ورودی تابع زیر 9 باشد خروجی آن کدام است؟

```
FUNCTION STP( A:BYTE) : LONGINT;
BEGIN
   IF A = 1 THEN
     STP: = 1
  ELSE
    STP: = A*STP(A-2);
END;
```

الف) 362880 ب) 384 ج) 945 د) 25

۱۰- خروجی برنامه زیر چیست؟

```
Var   a,b    : Integer;
      S      : String;
Begin
   a: = 19;   b: = 2;    s: = '';
   while a>0 do
   Begin
        s: = str(a mod b) +s;
        a: = a div b;
   end;
 writeln(s);
end.
```

الف) پیغام خطا ب) 10111 ج) 11001 د) 10011

۱۱- خروجی برنامه زیر چیست؟

```
Procedure   wo(s: string;   n: Byte);
Begin
          IF  n< Length (s) then
            wo (s,n+1);
          write (s[n]);
end;
Begin
      Wo('Reza',1);
end.
```

الف) Reza ب) azeR ج) R د) zeR

۱۲- خروجی تابع زیر اگر n = 40 باشد کدام است؟

```
Function   Ad(n: Byte):word;
Begin
         IF  n=1 then
             Ad:= 1
          Else
             Ad:= Ad(n-1)+n;
End;
```

الف) 420 ب) 1600 ج) 780 د) 820

۱۳- در صورتی که تابع زیر به فرم writeln(f(471)); احضار شود چاپ می شود؟

(کارشناسی ارشد صنایع – دولتی ۷۸)

```
Function f(x:integer) : integer;
   Var
      K:integer;
Begin
```

```
    K:  0;
    While(x>0) do
    Begin
        K:= k*10 + (x mod 10);
        X:= x div 10;
    End;
    F:= k;
End;
```

الف) 174 ب) 423 ج) 792 د) 452

۱۴- با احضار زیربرنامه مقابل در صورتی که ..... چاپ می شود. (کارشناسی ارشد صنایع – دولتی ۷۸)

```
Procedure P(m,n: Integer);
   Var
     i: integer;
   begin
     if not (m>n) then
        for i:= m to n do
            if not (odd(i)) then writeln(i);
end;
```

الف) m<n نباشد اعداد زوج m تا n ب) m>n باشد اعداد فرد m تا n

ج) m<n باشد اعداد فرد m تا n د) m>n باشد اعداد موجود بین m تا n

۱۵- فرض کنید که Y یک آرایه صحیح (به طول N) که یکی از مؤلفه های آن برابر مقدار صحیح X است. نتیجه فراخوانی تابع زیر ، یعنی مقدار برگردانده شده در F برابر است با ..... (کارشناسی ارشد صنایع – دولتی ۷۹)

```
FUNCTION TEST (N,X:INTEGER; VAR Y:ARRAYTYPE): INTEGER;
VAR I: INTEGER;
          I:= N;
          WHILE (X<>Y[I]) DO
                    I:= I-1;
          F:= I
END;
```

الف) I+1 به طوری که X=Y(I)

ب) کمترین مقدار I به طوری که X=Y(I)

ج) بیشترین مقدار I به طوری که X=Y(I)

د) تعداد دفعاتی که X در آرایه Y موجود است

۱۶ زیربرنامه زیر را در نظر بگیرید. سپس گزینه صحیح را در خصوص عملی که این زیربرنامه انجام می دهد ، انتخاب کنید. (کارشناسی ارشد صنایع – دولتی ۷۹)

```
Procedure Test (N:integer);
Var SUM:integr;
Begin
     SUM:= N;
     If SUM<10 then
          Write(SUM)
     Else begin
          Write(SUM MOD 10);
          SUM:= SUM DIV 10;
          Test (SUM)
        End
End;
```

الف) عدد N را می گیرد و ارقام آن را می نویسد.

ب) عدد N را می گیرد و جمع ارقام آن را می نویسد.

ج) عدد N را می گیرد و ارقام آن را در جهت عکس می نویسد.

د) عدد N را می گیرد و باقیمانده و خارج قسمت آن را بر ۱۰ می نویسد.

۱۷- فرض کنید A یک آرایه صحیح (به طول N) با حداقل یکی از درایه های آن برابر مقدار صحیح B است. تابع TRY را در نظر بگیرید: (کارشناسی ارشد صنایع – دولتی ۸۰)

```
FUNCTION TRY (B:INTEGER; VAR A:ARRAYTYPE):INTEGER;
VAR J:INTEGER;
BEGIN
      J:=1;
      WHILE (B<>A[J]) DO
            J:=J+1;
      TRY:= J;
END;
```

مقدار TRY پس از فراخوانی آن برابر است با .....

الف) J+1 به طوری که A[J]=B ب) J-1 به طوری که A[J]=B

ج) بیشترین مقدار J به طوری که A[J]=B د) کمترین مقدار J به طوری که A[J]=B

۱۸- زیربرنامه زیر را در نظر بگیرید: (کارشناسی ارشد صنایع – دولتی ۸۰)

```
PROCEDURE TEST (M: INTEGER; VAR DIGIT: INTEGER);
BEGIN
    DIGIT:= 0;
    WHILE (M>=10) DO
```

```
    BEGIN
        DIGIT:= DIGIT + M MOD 10;
        M:= M DIV 10
    END;
    DIGIT:= DIGIT + M
END;
```

با فراخوانی مناسب TEST ، مقدار DIGIT برابر است با .....

الف) حاصل جمع ارقام عدد M

ب) دو برابر حاصل جمع ارقام عدد M

ج) حاصل جمع ارقام عدد M با خود عدد M

د) حاصل جمع ارقام عدد M منهای یکان عدد M

۱۹- پس از فراخوانی مناسب تابع : (کارشناسی ارشد صنایع – دولتی ۸۰)

```
FUNCTION LOGIC (X,Y: INTEGER) : BOOLEAN;
BEGIN
    LOGIC:= X<>Y;
    IF (X=Y) THEN LOGIC:= TRUE;
END;
```

مقدار LOGIC .....

الف) همواره درست است.

ب) تنها به ازای X=Y درست است.

ج) تنها به ازای X#Y درست است.

د) به علت خطای منطقی در زمان اجرا نامعلوم است.

۲۰- پس از اجرای زیربرنامه ALFA ، مقدار R چیست؟(کارشناسی ارشد صنایع – دولتی ۸۱)

```
Procedure ALFA (A:KType; N:integer; VAR R:Real);
Var I:integer;
Begin
    R:= 0.0;
    FOR I:=2 TO N Do
        R:=R+A[I-1,I] * A[i-1]
END;
```

الف) ضرب داخلی زیر قطر اصلی و زیر قطر فرعی

ب) ضرب داخلی زیر قطر اصلی و بالای قطر فرعی

ج) ضرب داخلی بالای قطر اصلی و زیر قطر فرعی

د) ضرب داخلی زیر قطر اصلی و بالای قطر اصلی ماتریس

۲۱- فرض کنید Y یک آرایه صحیح است که یکی از مولفه های آن مقدار متغیر صحیح X را دارا می باشد. نتیجه فراخوانی تابع زیر کدام است؟ (مقدار برگردانده شده در F چیست؟) فرض کنید ARRAYTYPE به صورت ARRAY[1..100] OF INTEGER تعریف شده است؟ (کارشناسی ارشد ریاضی – دولتی ۷۴)

```
FUNCTION F(X:INTEGER; Y:ARRAYTYPE): INTEGER;
VAR   I : INTEGER;
BEGIN
   I:= 1;
   WHILE (X<>Y[I]) DO
       I:= I+1;
   F:= I
END;
```

الف) به طوری که X=Y[I]

ب) I+1 به طوری که X=Y[I]

ج) تعداد دفعاتی که X در آرایه Y موجود است

د) تعداد دفعاتی که X در آرایه Y موجود نیست

۲۲- پس از اجرای تابع زیر ، کدام گزینه در مورد مقدار TRY صحیح است؟ (کارشناسی ارشد ریاضی – دولتی ۷۴)

```
FUNCTION TRY(A,B: REAL) : BOOLEAN;
BEGIN
      TRY:= (A>=B)
END;
```

الف) تنها وقتی که A=B ، درست است.

ب) تنها وقتی که A<=B ، درست است.

ج) وقتی که A<=B ، درست و در غیر اینصورت نادرست است.

د) وقتی که A>=B ، درست و در غیر این صورت نادرست است.

۲۳- پس از اجرای تابع زیر ، کدام گزینه در مورد ارزش Javab صحیح است؟ (کارشناسی ارشد ریاضی – دولتی ۷۵)

```
FUNCTION Javab(A,B:REAL):BOOLEAN;
Begin
```

```
        Javab:=(A<B)
End;
```

الف) ارزش Javab درست است تنها وقتی که A=B

ب) ارزش Javab درست است تنها وقتی که A#B

ج) ارزش Javab درست است تنها وقتی که A<B

د) ارزش Javab درست است تنها وقتی که A بزرگتر یا مساوی B باشد.

۲۴- فرض کنید که X,Y آرایه های اعداد صحیح هستند. پس از اجرای تابع زیر به ازای آرایه های تعریف شده Y,X گزینه درست در خصوص مقدار برگردانده شده در نام تابع یعنی F را تعیین کنید. (کارشناسی ارشد ریاضی – دولتی ۷۶)

```
FUNCTION F(X:Xtype;  Ytype;  N:INTEGER):BOOLEAN;
VAR I: INTEGER;
BEGIN
      FOR I:= 1 TO N DO
      BEGIN
      F:= FALSE;
      IF (X[I] = Y[I]) THEN F:= TRUE
      END
   END;
```

الف) مقدار F وقتی درست است که درایه های N ام Y,X با هم برابر باشند.

ب) مقدار F همواره نادرست است.

ج) مقدار F درست است اگر هر یک از درایه های Y,X با هم برابر باشند.

د) مقدار F تنها وقتی درست است که همه درایه های Y,X با هم برابر باشند.

۲۵- پس از اجرای زیربرنامه TRY با پارامترهای ورودی مناسب ، مقدار محاسبه شده در R چیست؟ (کارشناسی ارشد ریاضی – دولتی ۷۶)

```
PROCEDURE TRY(VAR A: Atype;  N: INTEGER;  R: REAL);
VAR  I: INTEGER;
BEGIN
    R:= 0.0;
    FOR I:= 1 TO N-1 DO
    R:= R+A[I,I+1] * A[I+1,I]
END;
```

الف) ضرب داخلی زیر قطر اصلی و بالای قطر اصلی ماتریس A

ب) ضرب داخلی بالای قطر اصلی و زیر قطر فرعی ماتریس A

ج) ضرب داخلی زیر قطر اصلی و زیر قطر فرعی ماتریس A

د) ضرب داخلی زیر قطر اصلی و بالای قطر فرعی ماتریس A

۲۶- پس از اجرای زیربرنامه TRY با پارامترهای ورودی مناسب ، مقدار محاسبه شده در R کدام است؟ (کارشناسی ارشد ریاضی – دولتی ۷۷)

```
PROCEDURE  TRY(VAR A:Atype; N:INTEGER; R:REAL);
VAR I: INTEGER
BEGIN
R:= 0.0;
FOR I:= 2 TO N DO
  R:= R+A[I-1,I] * [I,I-1]
END;
```

الف) ضرب داخلی زیر قطر اصلی و زیر قطر فرعی ماتریس A

ب) ضرب داخلی زیر قطر اصلی و بالای قطر فرعی ماتریس A

ج) ضرب داخلی بالای قطر اصلی و زیر قطر فرعی ماتریس A

د) ضرب داخلی زیر قطر اصلی و بالای قطر اصلی ماتریس A

۲۷- در رابطه با استفاده از تکنیک برنامه سازی بازگشتی (Recursive) کدام یک از موارد زیر مورد نظر است؟ (کارشناسی ارشد ریاضی – دولتی ۷۹)

الف) تبدیل بهینه برنامه بازگشتی به زبان ماشین توسط مترجم زبان

ب) سهولت پیاده سازی برخی از الگوریتمها

ج) استفاده بهینه از حافظه در حین اجرا

د) سرعت بالای اجرای برنامه

۲۸- تابع زیر را در نظر بگیرید: (کارشناسی ارشد ریاضی – دولتی ۷۹)

```
FUNCTION SUM(N: INTEGER) : INTEGER;
BEGIN
      IF N=0 THEN SUM:= 1;
      ELSE IF N=1 THEN SUM:=2;
          ELSE SUM:= SUM(N-1)+SUM(N-2)
END;
```

چقدر SUM(4) چقدر است؟

الف) ۵ ب) ۶ ج) ۸ د) ۳

۲۹- تفاوت تابع (function) و رویه (procedure) در کدامیک از موارد زیر است؟ (کارشناسی ارشد ریاضی – دولتی ۷۹)

الف) دریافت داده ها از طریق پارامتر

ب) در نحوه فراخوانی و برگرداندن مقادیر پس از اجرا

د) در ایجاد حالت پیمانه ای بودن (modularity) برای برنامه

۳۰- رویه زیر را به زبان پاسکال در نظر بگیرید. فرض کنید که نوع VECTOR به صورت زیر در برنامه اصلی تعریف شده است: (کارشناسی ارشد ریاضی – دولتی ۷۹)

```
TYPE  ARRAYTYPE=ARRAY[1..100] OF INTEGER;
.
.
.
PROCEDURE NOW (VAR A:ARRAYTYPE; N:INTEGER);
VAR  I:INTEGER;
BEGIN
    f:=1;
    WHILE (I<=N-1) DO
     BEGIN
                IF (A[I]>A[I+1]) THEN A[I+1]:= A[I];
                I:=I+1;
    END;
END;
```

پس از فراخوانی و اجرای NOW ، گزینه صحیح را تعیین کنید.

الف) کوچکترین عدد در آرایه A در A[1] قرار می گیرد.

ب) بزرگترین عدد در آرایه A در A[1] قرار می گیرد.

ج) آرایه A به صورت صعودی مرتب می شود.

د) آرایه A به صورت نزولی مرتب می شود.

۳۱- تعیین کنید که تابع زیر به زبان پاسکال پس از فراخوانی چه چیز را محاسبه می کند : (کارشناسی ارشد ریاضی – دولتی ۷۹)

```
FUNCTION TRY(N:INTEGER) : INTEGER;
BEGIN
      IF (N=0) THEN TRY:=1
      ELSE
         TRY:= N*TRY(N-1)
END;
```

N-1 N

الف) 0 ب) N! ج) N د) N

۳۲- مقدار F(7) برای تابع زیر برابر است با :

```
FUNCTION F(M:INTEGER):INTEGER;
BEGIN
      IF(M<=2) THEN
           F:= M+M+2
      ELSE
           F:=F(M-2)-2
END.
```

الف) ۲- ب) ۰ ج) ۲ د) ۴

۳۳- خروجی برنامه زیر کدام است؟

```
Program sample (output)
   Var a,b : Integer;
   Procedure proc(x,y,z : Integer);
     Begin
          y:=y+1;
          z:=z+x;
      end
begin
     a:= 2;
     b:= 3;
     proc(a+b, a , a)
     write(a)
end.
```

الف) ۲ ب) ۳ ج) ۵ د) ۷

۳۴- با فراخوانی FUN(64) مقادیر نوشته شده در خروجی به ترتیب از چپ به راست کدام است؟

```
FUNCTION  FUN(N:INTEGR): INTEGER;
BEGIN
   IF N>0 THEN
       BEGIN
          N:= N DIV 2;
          FUN(N DIV 2);
          WRITE(N)
       END
END;
```

الف) 0 1 4 16 ب) 1 2 8 16 ج) 0 2 8 32 د) 1 2 4 8 16 32

۳۵- برنامه زیر را در نظر بگیرید.

```
Function F(x,n: Integer): Integer;
  Begin
    If n=1 then
      F:=x
    Else
      F:= x*F(x,n-1);
  End;
```

مقدار F(4,4) عبارت است از:

الف) ۸ ب) ۱۶ ج) ۶۴ د) ۲۵۶

۳۶- برنامه زیر را در نظر بگیرید.

```
Function Z(k,n: Integer): Integer;
    Begin
      If n=k then
        Z:=k
      Else
        If n>k then
        Z:= Z(k,n-k)
      Else
        Z:=Z(k-n,n)
    End;
```

۳۷- تابع زیر چه مقداری برگشت می دهد؟

```
Function TEST(X: REAL) : REAL;
Begin
    TEST:= X;
    If X<0 Then TEST:= -X
End;
```

الف) منفی x ب) مقدار X ج) قدر مطلق X د) تعداد ارقام x

۳۸- اگر برنامه زیر به فرم P(5.5, '+',W) احضار گردد مقدار W چه خواهد بود؟

```
Procedure P(x:real; OP:char; Var y:real);
Begin
    If (OP='*') then
        Y:= x*x*x
    Else if (OP='+') then
        Y:= x+x+x-10.0;
End;
```

الف) 6.5 ب) 16.5 ج) 27.75 د) معلوم نیست.

۳۹ عملکرد برنامه زیر کدام است؟

```
FUNCTION P(x:real; N:integer) :real;
Var
   Pr: real;
   Count: integer;
Begin
   Pr:= 1.0;
   For Count:= 1 to ABS(N) do
      Pr:= Pr*x;
   If N >=0 then P:= Pr
      Else
        P:= 1.0/Pr;
End;
```

الف) N بار مقدار x را در خودش ضرب می کند.

ب) N بار مقدار x را با خودش جمع می کند.

ج) ۳۲۷۶۸ بار X را با خودش جمع می کند.

د) ۳۲۷۶۸ بار x را در خودش ضرب می کند.

۱- خروجی برنامه زیر کدام است؟

```
Var C:char;
      K: Set of 'A'..'Z';
Begin
      C:='G';
      Writeln(C in K)
End;
```

الف) C in K ب) FALSE ج) TRUE د) G

۲- خروجی برنامه زیر کدام است؟

```
Var K: set of 1..10;
      M: set of 5..20;
Begin
      K:=[2..7];
      M:=[6..8];
      M:= K+M;
      Write (ord (2 in M));
End;
```

الف) 1 ب) 0 ج) TRUE د) FALSE

۳- خروجی برنامه زیر کدام است؟

```
Var A: set of  5..100;
     B: set of  1..100;
Begin
    A:= [1..100];
    B:= [4..8];
    B:= A*B;
   Write(ord(5 in B));
End;
```

الف) 1 ب) 0 ج) TRUE د) FALSE

۴- خروجی برنامه روبرو چیست؟

```
Type Long= [Small, Medium, Large];
     Stong= [Medium, Large];
Var I,J: Long;
      K: Integer;
      J:= [Medium];
Begin
       For K:= 1 to 3 Do
           Begin
               I:= I + [Small];
               K:= K + 1;
           End;
```



```
     J:= I*J;
Write(I,J in Stong);
End;
```

الف) False, [Small,Medium] ب) True, [Small,Medium]

ج) False, [Small,Small] د) برنامه خطا دارد.

۵- با تعریف زیر در توربو پاسکال چند بایت حافظه اشغال می شود؟

```
Var   X: Set of 'a' .. 'z' ;
```

الف) ۲۶ بایت ب) ۵۲ بایت ج) ۴ بایت د) ۶ بایت

۶- متغیر نوع مقابل چند بایت اشغال می کند؟

```
Var  m: set of char;
```

الف) 32 ب) 16 ج) 8 د) 64

۱ در پاسکال با تعریف زیر اندازه متغیر i چند بایت خواهد بود؟

```
Type sn:= (car1, car2, car3);
Var   i: record
        s: string[6];
        case  b:sn of
            car1: (n:Real);
            car2: (c:char);
            car3: (ch:array[1..8] of integer);
        end;
```

الف) ۲۳ بایت ب) ۳۱ بایت ج) ۲۴ بایت د) ۳۲ بایت

۲- اندازه ساختاری از نوع H چند بایت است؟

```
Type  V= (R,S,T);
       H= Record
          x,y: Real;
          Case  Z: v of
                S: (A: Integer);
                R: (B,C: Real);
                T: (D,E,F: char);
End;
```

الف) ۲۸ بایت

ب) اندازه این ساختار نامعلوم است و در حین اجرا مشخص می شود.

ج) ۲۴ بایت

د) ۲۵ بایت

۳- طول رکورد روبرو چقدر است؟

```
TYPE
EM= RECORD
N: STRING[10];
T: REAL;
C: CHAR;
END;
```

الف) 12 ب) 13 ج) 14 د) 18

۴- طول رکورد زیر چند بایت است؟

```
TYPE  LOOP = RECORD
          N: REAL;
          X: STRING[5];
         CASE A: BYTE OF
               1: (L,M:REAL);
```



```
               2: (J:REAL);
END;
```

الف) 27 ب) 24 ج) 23 د) 25

۵- اندازه حافظه اشغالی متغیر g کدام است؟

```
TYPE
   List= Record
   Case Integer of
    1: (Name : String[15]);
    2: (Number : Array[1..15] of Byte);
END;
VAR
    g: list;
```

الف) 15 ب) 16 ج) 17 د) 18

۶- طول رکورد زیر چقدر است؟

```
Type  stu = Record
       N: string[15];
       STN: word;
      Case d: Byte   of
         1: (a,b,c : Integer);
         2: (e: Real; f: Byte);
end;
```

الف) 23 ب) 26 ج) 24 د) 25

۷- با توجه به قطعه برنامه زیر متغیر Person چند بایت حافظه را اشغال می نماید؟

```
type
employee = Record
name: string[10];
id: integr;
tex: real
end;
var Person : employee;
```

الف) 14 ب) 19 ج) 18 د) 20

۸- کدام تعریف درست است؟

الف)
```
Var  M: Record
       a: byte;
       case i: char of
       'A': (x:word);
```

ب)
```
Var  M: Record;
       a: byte;
       case  i: char of
       'A': (x: word);
```

```
'B': (y:char);
case Boolean of
True: (p:Real);
False: (k: word);
End;
```

```
'B': (y: char);
    j: byte;
end;
```

ج)
```
Var M: Record
  m: byte;
  case char of
'A': (x:word);
'B': (y:char);
  end;
```

د)
```
Var  M:Record
    a : byte;
 case  char of
'A': (a:word);
'B': (y:char);
   else (z:byte);
 end;
```

۹- متغیر x چند بایت فضا می گیرد؟

```
Var x: Record
    a: shortint;
    b: Record
        c: char;     d: (bad,good);
    end;
    case byte of
       2: (e: (R,G,B));
       3: (f: shortint);
    end;
```

الف) ۴ بایت ج) ۵ بایت ج) ۳ بایت د) ۸ بایت

۱۰- مجموعه ای از فیلدهای مرتبط به هم را چه می نامند؟

الف) رکورد ب) بایت ج) پرونده د) کلمه

۱- روی فایل متن کدامیک از دستورات زیر را می توان استفاده کرد؟

الف) Seek ب) Filepose ج) Eoln د) Filesize

۲- برای بررسی وجود یک فایل متن از چه دستوری می توان استفاده کرد؟

الف) Reset(file1) ب) Assign(file1)

ج) Exist(file1) د) SEEKOF(File1)

۳- دستورات زیر برای یک فایل دارای نوع چه عملی انجام می دهند؟

```
Reset(file1);
Seek(file1,10);
Rec:=100;
Write(file1,Rec);
```

الف) یک رکورد دیگر به انتهای فایل (رکورد دهم) اضافه می کند و فایل دارای ۱۱ رکورد می شود.

ب) این دستورات موجب بروز خطا می شوند. زیرا نوشتن در فایل خواندنی غیرمجاز است.

ج) هیچ عملی انجام نمی شود زیرا فایل برای خواندن باز شده است.

د) یک رکورد به انتهای فایل اضافه می کند و اندازه فایل برابر ۱۰ رکورد خواهد شد.

۴- دستور .... Rewrite در مورد یک فایل دارای نوع چه عملی انجام می دهد؟

الف) فایل را برای نوشتن باز می کند ولی اشاره گر فایل را تغییر نمی دهد.

ب) اگر فایل موجود باشد آن را برای نوشتن باز می کند و در غیر این صورت خطای عدم وجود فایل بوجود می آید.

ج) فایل را برای نوشتن باز می کند و اشاره گر فایل را به ابتدای فایل منتقل می کند ولی محتوای فایل حذف نمی شود.

د) فایل را برای نوشتن / خواندن (ویرایش) باز می کند و اشاره گر فایل را به ابتدای فایل منتقل می کند و محتوای فایل حذف می شود.

۵- کدام گزینه درباره تکه برنامه زیر صحیح است؟

```
i: 0;
While not eof(file1) do begin
        Readln(file1);      Inc(i);
End;
```

الف) تعداد حروف موجود در یک فایل متن را بدست آورده و در متغیر i قرار میدهد.

ب) شکل استفاده از دستور Readln اشتباه است و برنامه خطا دارد.

ج) تعداد کاراکترهای Enter موجود در یک فایل را بدست آورده و در متغیر i قرار می دهد.

د) تعداد خطوط موجود در یک فایل متن را محاسبه کرده و در متغیر i قرار می دهد. (تعداد خطوط = i)

۶- اگر شماره آخرین رکورد فایل زوج باشد برای اینکه اشاره گر فایل Seek(t,k) را دقیقا وسط فایل قرار دهیم کدام گزینه باید به جای k قرار گیرد؟

الف) filesize(t) Mod 2　　ب) filepos(t) Div 2

ج) filesize(t/2)　　د) filesize(t) Div 2

۷- با اجرای خطوط زیر چه عملی انجام می شود؟

```
Reset (fptr);
Seek (fptr, 10);
MyRec:= 10;
Wrute(fptr, MyRec);
```

الف) این دستورات موجب بروز خطا می شوند زیرا نوشتن در فایلی که برای خواندن باز شده غیرمجاز است.

ب) چون فایل برای خواندن باز شده است رکوردی به فایل اضافه نمی شود.

ج) یک رکورد به انتهای فایل بدون نوع اضافه می کند و تعداد رکوردها برابر ۱۱ می شود.

د) یک رکورد دیگر به انتهای فایل (رکوردهم) اضافه می کند و فایل دارای ۱۰ رکورد می شود.

۸- در برنامه زیر شماره خطی که خطا تولید می کند کدام است؟

```
1) VAR A:TEXT;
   BEGIN
2)       ASSIGN(A,'TEMP');
3)        REWRITE(A);
4)        WRITELN(A,'SAMPLE');
5)        RESET(A);
6)       CLOSE(A);
   END.
```

الف) ۲　　ب) ۳　　ج) ۵　　د) هیچکدام

۹- کدام فرمان زیر فقط با فایل متنی قابل استفاده است؟

الف) Append　　ب) Read　　ج) Seek　　د) Rewrite

۱۰- اگر شماره آخرین رکورد فایل زوج باشد ، کدام فرمان اشاره گر فایل را به وسط فایل انتقال می دهد؟

الف) seek(fp, ftell(fp)/2)　　ب) seek(fp, filepos(fp) div 2)

ج) seek(fp, filesize(fp) div 2)　　د) seek(fp, filesize(fp)/2)

۱۱- فایلی نوع دار دارای ۲۵ رکورد و هر رکورد ۱۰ بایت است. می خواهیم ۵ رکورد آخر فایل را حذف کنیم. کدام گزینه این کار را انجام می دهد؟

الف)	Truncate(f)	Seek(f,210)
ب)	Truncate(f)	Seek(f,200)
ج)	Truncate(f)	Seek(f,21)
د)	Truncate(f)	Seek(f,20)

۱۲- برنامه زیر چه عملی را انجام می دهد؟

```
PROGRAM sample (data1, data2);
    TYPE line = PACHED ARRAY [1..80] Of char;
    Personal = RECORD
                    Name : line;
                    Address : line;
                    Phone : line;
               END;
VAR data1, data2 : FILE OF personal;
```

```
        Nameandaddress : personal;
Begin
        Reset(data1);
        Rewrite(data2);
While NOT eof (data1) DO
    BEGIN
            Read(data1, Nameandaddress);
            Write(data2, Nameandaddress);
    END;
END.
```

الف) کپی یک سری رکورد از یک فایل به فایل دیگر

ب) نوشتن یک سری رکورد اطلاعاتی

ج) خواندن یک سری رکورد اطلاعاتی

د) وارد کردن نام از صفحه کلید و جستجوی فایل برای رکوردی با همان اسم

فهرست منابع و ماخذ

1. TURBO PASCAL 5th Edition, Elliot B. Koffman 1998.
2. Steven Nameroff, Using quick PASCAL, Osborane Mc Graw- Hill, 1989.
3. Stephen O. Brien And Steven Nameroff, TURBO PASCAL 7, Mc Graw–Hill, 1993.
4. Geneva G. Belford And C.L. Liu, PASCAL, Mc Graw- Hill, 1986.
5. Douglas F.Riddle, Programming in Pascal, Maxwell Macmillan, 1991.
6. Elliot B. Koffman, Problem Solving And Structured Programming in PASCAL Addison–Wesley Publishing Company, 1985.

درست‌نامه

لطفاً موارد زیر را با توجه به صفحه مورد نظر اصلاح شود:

۱- صفحه ۸۵: مثال وسط صفحه

```
6  div  3 = 2        6  mod  3 =  0
8  div  3 = 2       -6  mod  4 = -2
3  div -3 = -1       7  mod -3 =  1
-6 div  3 = -2      -7  mod -3 = -1
```

۲- صفحه ۹۱: جدول ۸-۵

نوع	محدوده	اندازه بر حسب بایت
Byte	از ۰ تا ۲۵۵	۱
Shortint	از ۱۲۸- تا ۱۲۷	۱
Integer	از ۳۲۷۶۸- تا ۳۲۷۶۷	۲
Word	از ۰ تا ۶۵۵۳۵	۲
Longint	از ۲۱۴۷۴۸۳۶۴۸- تا ۲۱۴۷۴۸۳۶۴۷	۴

۳- صفحه ۱۰۷: پاراگراف دوم ☐☐ ۱۲۳

۴- صفحه ۲۸۹: پاراگراف اول

```
a1:=['a','b','c'];
a2:=['x','y','z'];
c:=['U','V','W','X','Y','Z'];
```

۵- صفحه ۳۰۶: مثال ۳-۱۳

C, D: String [10]; → ۲*۱۱

لذا: ۱۲+۲۲+۱۰+۱۰=۵٤

۶- صفحه ۳۴۱: مسئله ۷

```
For j:=1 to n do begin
    i:=n,;
   While  i > 1   do
       i:=i div 2;
 end;
```

خوانندهٔ محترم

این پرسشنامه به منظور ارتقای کیفیت کتابهای درسی و رفع نواقص آنها تهیه شده است. دقت شما در پاسخگویی به این پرسشنامه در پایان هر نیمسال ما را در تحقق این هدف یاری خواهد کرد.

نام کتاب ..... نام مؤلف/مترجم ..... سال انتشار .....

وضعیت پاسخگو: عضو علمی پیام نور ☐ عضو علمی سایر دانشگاهها ☐ رشتهٔ تخصصی ..... سابقهٔ تدریس .....

دانشجوی پیام‌نور ☐ دانشجوی سایر دانشگاهها ☐ رشتهٔ تحصیلی ..... ورودی سال .....

سؤال	خیلی زیاد	زیاد	کم	بسیار کم
۱. آیا از زمان تحویل و نحوهٔ دسترسی به کتاب راضی بودید؟				
۲. آیا حجم کتاب با توجه به تعداد واحد مناسب بود؟				
۳. آیا راهنماییهایی لازم برای مطالعهٔ کتاب منظور شده بود؟				
۴. آیا در ترتیب مطالب کتاب سلسله مراتب شناختی (آسان به مشکل) رعایت شده بود؟				
۵. آیا تقسیم‌بندی مطالب در فصلها و یا بخشها متناسب و بجا بود؟				
۶. آیا متن کتاب روان و ساده و جمله‌ها قابل‌فهم بود؟				
۷. آیا به روز بودن مطالب و آمارها رعایت شده بود؟				
۸. آیا مطالب تکراری داشت؟				
۹. آیا پیوستگی مطالب با درسهای پیش‌نیاز رعایت شده بود؟				
۱۰. آیا مثالها، شکلها، نمودارها، جدولها و... گویا بودند و در فهم مطلب تأثیر داشتند؟				
۱۱. مطالعهٔ هدفهای کلی، آموزشی/رفتاری تا چه اندازه به درک بهتر شما کمک کرد؟				
۱۲. آیا خودآزماییهای کتاب به گونه‌ای بود که تمام مطالب درسی را شامل شود؟				
۱۳. آیا پاسخ خودآزماییها و تمرینها کامل و گویا بود؟				
۱۴. چقدر با غلطهای املایی و اشکالهای چاپی مواجه شدید؟				
۱۵. کیفیت چاپ و صحافی کتاب چگونه بود؟				
۱۶. آیا طرح روی جلد کتاب مناسب بود؟				
۱۷. چنانچه از وسایل کمک‌آموزشی از قبیل نوار، فیلم، لوح فشرده و... استفاده کرده‌اید، آیا به درک بهتر شما کمک کرده است؟				
۱۸. تا چه اندازه این کتاب شما را از حضور در کلاس بی‌نیاز کرد؟				

لطفاً چنانچه با اشکالهای تایپی یا محتوایی و مطالب تکراری مواجه شده‌اید، فهرستی از آنها را با ذکر شمارهٔ صفحه ضمیمه کنید.

در مجموع کتاب را چگونه ارزیابی می‌کنید؟ عالی ☐ خوب ☐ متوسط ☐ ضعیف ☐

در صورت تمایل سایر پیشنهادهای خود را نیز بنویسید.

این پرسشنامه را پس از تکمیل از کتاب جدا کنید و به قسمت آموزش مرکز تحویل دهید یا مستقیماً به نشانی تهران ۱۹۵۶۹- صندوق پستی ۲۶۹۷ ۱۹۳۹۵ مدیریت تدوین ارسال فرمایید.

با تشکر

مدیریت تدوین