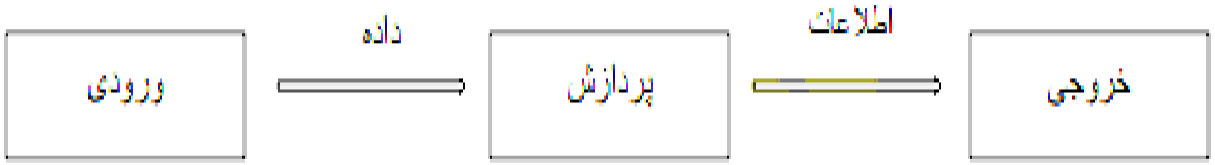
مبانی کامپیوتر و برنامه سازی

سیستم کامپیوتری

هر سیستم کامپیوتری از مجموعه ای از سخت افزارها و نرم افزارهایی تشکیل شده است که در جهت انجام کار خاصی با یکدیگر همکاری می کنند. تعریف دقیق تر آن بدین صورت است که هر سیستم کامپیوتری از وسایل الکترونیکی و الکترومکانیکی تشکیل شده است که داده هایی را به عنوان ورودی دریافت کرده و عملیات خاصی را طبق مجموعه ای از دستورالعمل ها بر روی داده ها انجام می دهد و نتایج حاصل از عملیات را به عنوان خروجی تولید می کند.

سیستم کامپیوتری



داده : مجموعه مطالبی که وارد کامپیوتر می شود داده گفته می شود. داده ها می توانند به صورت عدد، حرف، صدا، تصویر و ... باشد.

پردازش : به کلیه فعالیت های صورت گرفته برروی داده ها که منجر به پیدایش نتایج می شود پردازش داده ها گفته می شود.

اطلاعات : به خروجی که بعد از پردازش داده ها تولید می شود اطلاعات اطلاق می گردد.

سیستم کامپیوتری

الگوریتم : دستورالعمل هایی که برای کامپیوتر نوشته می شود را الگوریتم گوییم.

برنامه کامپیوتری : به تشریح الگوریتم ها برای کامپیوتر با استفاده از یک زبان برنامه سازی گفته می شود.

زبان برنامه سازی: زبانی است که برای کامپیوتر قابل فهم بوده و الگوریتم ها با استفاده از آن به کامپیوتر داده می شوند.

زبان ها

مهندسین نرم افزار تاکنون برای زبانها پنج نسل را در نظر گرفته اند:

زبان نسل اول: که به آن زبان ماشین نیز گفته می شود، مستقیما به زبان خود کامپیوتر(یعنی زبان صفر و یک) نوشته می شود و توسط کامپیوتر قابل اجرا می باشد. هر کامپیوتری زبان ماشین مخصوص به خود را دارد که وابسته به سخت افزار خود آن کامپیوتر است.

زبان نسل دوم: زبان اسمبلی است که حالت نمادین زبان ماشین است و در آن دستورات با استفاده از یک نماد بجای ۰و۱ بجای کد صفر و یک نوشته می شوند.

زبان ها

زبان نسل سوم : این نسل شامل زبان های سطح بالا است که از جمله زبان های این نسل می توان به زبان های C++، JAVA، FORTRAN، Basic، PASCAL، C#، Cو... اشاره کرد.برنامه نویسی به این زبانها بسیار نزدیک به زبان انسان هستند و از دستوراتی مشابه زبان طبیعی (زبان انگلیسی) تشکیل شده اند.

زبان نسل چهارم: این نسل شامل زبانهای بسیار سطح بالا هست که از جمله زبان های این نسل می توان به SQL اشاره کرد که زبانی است خاص منظوره و همچون زبان طبیعی که برای دریافت اطالعات از پایگاه داده بکار می رود.

زبان نسل پنجم: زبان های این نسل شامل محیط های گرافیکی مناسب و راحتی برای تولید نرم افزار ها هستند از جمله می توان به Visual C#، Visual C++و ... اشاره کرد.

انواع کامپیوتر ها

کامپیوتر ها از لحاظ قدرت پردازشی به گروه های مختلفی تقسیم می شوند:

- **ابر رایانه ها:** این نوع رایانه ها، قوی ترین و گرانترین نوع رایانه ها است که از قدرت اجرایی و سرعت بسیار بالایی برخوردار هستند و بیشتر در زمینه های نظامی، تحقیقاتی، علوم فضایی و پروژه های علمی بزرگ مورد استفاده قرار می گیرند.
- **کامپیوترهای بزرگ:** این کامپیوترها از سرعت و قدرت بالایی برخوردارند و معمولا دردانشگاهها و سازمانهای بزرگ و برای محاسبات سنگین استفاده می شوند. توان محاسباتی این رده نسبت به ابر رایانه ها کمتر است.

کامپیوترهای کوچک: از آنجا که قیمت کامپیوترهای بزرگ بسیار بالا بود، در اواخر دهه ۱۹۵۰کامپیوترهای کوچک وارد بازار شدند که توان محاسباتی کمتری داشتند و توسط سازمانهای کوچکتر مورد استفاده قرار می گرفتند.

- **ریزکامپیوتر:** در آغاز دهه ۱۹۸۰ریزکامپیوترها یا کامپیوترهای شخصی با قیمت پایین و حجم بسیار کوچک وارد بازار شدند و مورد استقبال مردم و افراد عادی قرار گرفتند.

اجزای کامپیوتر

هر کامپیوتر از دو قسمت اصلی تشکیل شده است :

- ✓ **سخت افزار :** کلیه دستگاههای الکتریکی، الکترونیکی و مکانیکی تشکیل دهنده یک کامپیوتر را سخت افزار آن می گوییم.

- ✓ **نرم افزار:** مجموعه برنامه هایی هستند که برای یک کاربرد خاص نوشته شده اند و بدون آنها سخت افزار قادر به کاری نیست.

سخت افزار

اجزای تشکیل دهنده سخت افزار کامپیوتر عبارتند از :

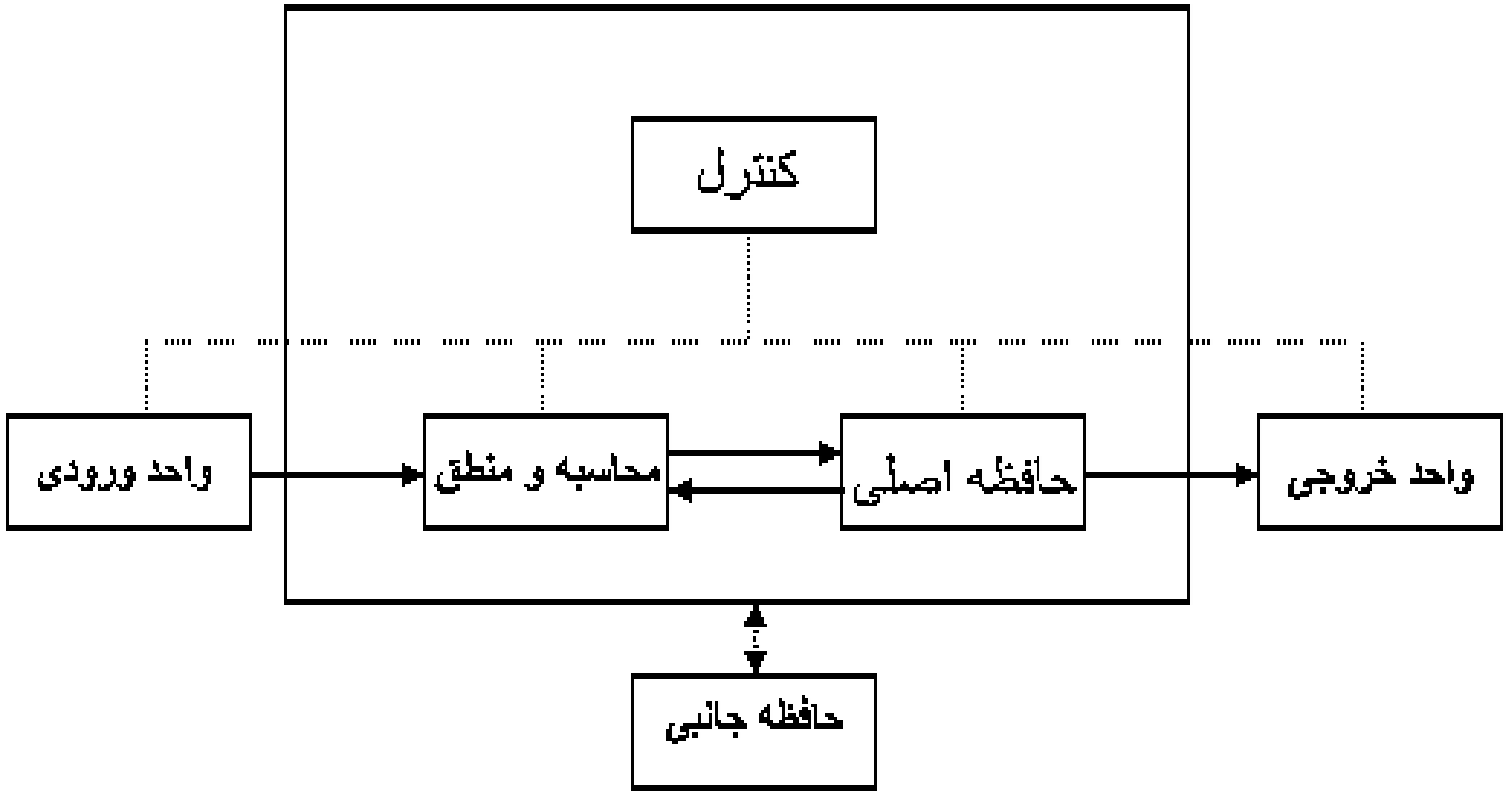
- واحد ورودی
- واحد خروجی
- واحد حافظه
- واحد محاسبه و منطق
- واحد کنترل و حافظه جانبی

سخت افزار

واحد ورودی: وظیفه این بخش دریافت داده ها از دستگاه های ورودی و انتقال آنها و تبدیل آنها به داده های قابل فهم برای کامپیوتر می باشد. دستگاههای ورودی مهم عبارتند از:صفحه کلید، ماوس، صفحه لمسی، قلم نوری، اسکنر و...

واحد خروجی: این بخش وظیفه انتقال اطلاعات از کامپیوتر به محیط خارج را بعهده دارد و مهمترین دستگاههای خروجی عبارتند از :صفحه نمایش، بلندگو، چاپگر و...

اجزای سیستم



برنویک مشیس کی هدنهد لیکشت یازجا

سخت افزار

- **واحد محاسبه و منطق:** واحدی است که تمامی عملیات ریاضی همچون جمع، ضرب، تفریق، تقسیم و منطقی همچون مقایسه دو مقدار و ... در آن انجام می پذیرد.

- **واحد کنترل:** این بخش نقش نظارت و کنترل بر ورود اطلاعات از طریق ورودی، ذخیره آنها در حافظه، انتقال اطلاعات از حافظه به واحد محاسبه و منطق و خروج اطلاعات از طریق واحد خروجی را دارد. بطور کلی وظیفه کنترل سایر بخشها را بعهده دارد و تصمیم میگیرد کدام عمل در چه زمانی صورت پذیرد و چه مداراتی فعال و یا غیر فعال گردند.

واحد کنترل بهمراه واحد محاسبه و منطق بخش مهم تشکیل دهنده واحد پردازش مرکزی یا CPUهستند. اجزای دیگر تشکیل دهنده CPUعبارتند از ثبات ها و حافظه نهان یا کش.

سخت افزار

واحد حافظه: این واحد وظیفه نگهداری اطلاعات (شامل داده ها و برنامه ها) را بصورت موقت و دائم بر عهده دارد. حافظه ها به دو دسته تقسیم می شوند:

1. **حافظه اصلی :** در واقع هر برنامه ای برای اجرا، ابتدا باید بهمراه داده های مورد نیاز وارد حافظه اصلی گردد. ویژگی اصلی حافظه اصلی آنست که از سرعت بسیار بالایی برخوردار است اما با قطع برق اطلاعات آن از بین می رود.حافظه اصلی به دو دسته اصلی تقسیم می گردد :

- **حافظه با دستیابی تصادفی :** این حافظه قابل خواندن و نوشتن می باشد و برای ذخیره اطلاعات کاربران بکار می رود.

- **حافظه فقط خواندنی:** این حافظه فقط قابل خواندن است و محتویات آن قابل تغییر نیست. این حافظه معمولا در کارخانه سازنده پر شده و حاوی دستورالعملهای لازم برای راه اندازی اولیه کامپیوتر می باشد.

2.حافظه جانبی : این حافظه نسبت به حافظه اصلی سرعت کم تری دارد ولی اطلاعات ذخیره شده در آن با قطع برق از بین نمی رود و حجم ذخیره سازی داده در آنها نسبت به حافظه اصلی بسیار زیاد است. حافظه جانبی انواع گوناگونی دارند که می توان CD،DVD، هارد دیسک ، Flash Memoryها و... را نام برد.

اصطلاحات

بیت (bit): حافظه از واحدهای کوچکی بنام بیت تشکیل شده است که هر بیت قابلیت نگهداری یک ۰ یا ۱ را در خود دارد.

بایت (Byte): به هر بیت یک بایت گفته می شود که واحد اندازه گیری حافظه است.

کاراکتر(character): در کامپیوترها کار با اطلاعات بر مبنای بیت دشوار است از این رو از کاراکترها استفاده می شود که بتوان اطلاعات را به صورت ارقام، حروف و نمادها نمایش داد. برای ذخیره سازی کاراکترها به هریک از آنها یک کد عددی نسبت داده شده است و در حقیقت کد عددی هر کاراکتر در کامپیوتر ذخیره می گردد. در گذشته پر کاربردترین کد مورد استفاده، کد ASCIIبود که برای نمایش هر کاراکتر از یک بایت استفاده می کرد.

اصطلاحات

از آنجا که هر بایت می تواند بین ۰ تا ۲۵۵ تغییر کند، بنابراین تا ۲۵۶ کاراکتر قابل تعریف است. از این بین کدهای بین ۰ تا ۱۲۷ بصورت استاندارد برای علائم و حروف انگلیسی تعریف شده است و کدهای بالاتر از ۱۲۷برای هر کشور خالی گذاشته شده است تا بتوانند حروف خاص زبان خود را تعریف کنند. کدهای ASCIIحروف بزرگ انگلیسی را نشان می دهد:

A	B	C	D	E	F	G	H	I	J	K	L	M
65	66	67	68	69	70	71	72	73	74	75	76	77
N	O	P	Q	R	S	T	U	V	W	X	Y	Z
78	79	80	81	82	83	84	85	86	87	88	89	90

اصطلاحات

فیلد (Fields): فیلدها از کاراکترها تشکیل شده اند. در واقع گروهی از کاراکترهایی که معنای خاصی دارند. به عنوان مثال فیلدی که شامل تنها کاراکترهای حروفی باشد می توان آن را فیلد نام یا نام خانوادگی در نظر گرفت.

رکورد (Record): به مجموعه ای از فیلدهای مرتبط به هم یک رکورد می گوئیم.

فایل (File): به مجموعه ای از رکوردهای مرتبط به هم فایل گفته می شود.

همانطور که گفتیم کوچکترین واحد حافظه بیت است. واحد حافظه بزرگتر از بیت، بایت است که از ۸ بیت تشکیل شده است، واحدهای بزرگتری برای سنجش میزان حافظه وجود دارد که در ادامه آنها را بیان می کنیم.

اصطلاحات

word

) : معموال هر ۲یا ۴بایت را یک کلمه در نظر می گیرند.

کیلوبایت (KB) : هر۱۰۲۴ بایت یک کیلو بایت را تشکیل می دهد. در سیستم دودویی هر کیلو بایت ۲ به توان ۱۰می باشد.

واحد های بزرگتر از کیلو بایت عبارتند از:

1 MegaByte or 1M = 1024 KiloByte

1 GigaByte or 1G = 1024 MegaByte

1 TeraByte or 1T = 1024 GigaByte

نرم افزار

سخت افزار به تنهایی توانایی انجام خواسته های کاربر و اجرای برنامه ها را ندارد از این رو برای بکارگیری سخت افزار از نرم افزار ها استفاده می کنیم. بطور کلی نرم افزار کامپیوتر به دو دسته اصلی تقسیم می گردد :

- **نرم افزارهای کاربردی :** نرم افزارهایی هستند که برای یک کاربرد خاص و رفع یک نیاز مشخص کاربران نوشته شده اند. مانند نرم افزار Word، ویرایش عکس و ...
- **نرم افزارهای سیستمی :** نرم افزارهایی هستند که برای ایجاد و یا اجرای برنامه های کاربردی نوشته می شوند و به سه گروه تقسیم می شوند.

نرم افزار های سیستمی

۱. سیستم عامل : سیستم عامل نرم افزاری است که ارتباط بین سخت افزار و کاربران (یا برنامه های کاربردی کاربران) را فراهم می سازد. در حقیقت سیستم عامل مدیریت منابع سخت افزاری یک کامپیوتر را بعهده دارد. چنانچه سیستم عامل نبود، کاربران مجبور بودند مستقیما و به زبان ماشین با سخت افزار صحبت نمایند که کار مشکلی بود. بهمین دلیل کلیه کاربران مجبورند با یکی از سیستم عاملهای موجود آشنا باشند. در حال حاضر دو سیستم عامل معروف برای کامپیوترهای شخصی وجود دارد : Windowsکه بیشتر در منازل و محیطهای اداری مورد استفاده قرار می گیرد و Linuxکه بیشتر در محیطهای دانشگاهی و بعنوان سرویس دهنده استفاده می شود. سیستم عامل Unixنیز بیشتر در کامپیوترهای بزرگ نصب می شود. سیستم عامل Androidنیز که امروزه بطور گسترده ای در موبایل ها و تبلت ها مورد استفاده قرار می گیرند.

۲. **برنامه های کمکی :** این برنامه ها استفاده از کامپیوتر را آسان تر می کند. از طریق این برنامه ها ارتباط کاربر با سخت افزار و برنامه های دیگر آسان تر می شود. از جمله این برنامه ها می توان به برنامه های مدیریت فضای دیسک و ویروس یاب ها اشاره کرد.

۳. **مترجم ها:** همانطور که گفتیم زبان کامپیوتر زبان ماشین است ولی برنامه نویسی به این زبان برای برنامه نویسان مشکل می باشد. برای اینکه کامپیوتر بتواند برنامه های نوشته شده به زبان سطح بالا را درک کند باید از مترجم استفاده کند. از این رو وظیفه مترجم تبدیل دستورات زبان سطح بالا به زبان قابل فهم برای کامپیوتر است.

مترجم ها

دو نوع اصلی مترجم داریم که عبارتند از:

- **کامپایلر :** ابتدا کل برنامه زبان سطح بالا را بررسی کرده و درصورت نبود خطا کل آن را به زبان ماشین تبدیل می کند. اکنون برنامه آماده اجرا است.

- **مفسر :** برنامه زبان سطح بالا را دستور به دستور به زبان ماشین تبدیل و همزمان آن را اجرا می کند

نمایش اطلاعات در کامپیوتر

اطلاعات در کامپیوتر به دو دسته اصلی تقسیم می گردند:

- اطلاعات کاراکتری (حرفی) : مانند : A B ...Z $ # @ !
- اطلاعات عددی که خود به دو دسته اعداد صحیح و اعداد اعشاری تقسیم می گردند.

برای نمایش اطلاعات در کامپیوتر از مبنای ۲استفاده می گردد .

سیستم اعداد

از کودکی یاد گرفته ایم که برای شمارش از اعداد دهدهی استفاده کنیم و با مفاهیمی مانند یکان، دهگان، صدگان و ... آشنا شده ایم. درواقع در ریاضیات متداول هر عدد N بصورت زیر تفسیر می گردد:

$$N = (a_{n-1}\, a_{n-2} \ldots\ a_2\, a_1\, a_0)_{10} = a_0 \times 10^0 + a_1 \times 10^1 + a_2 \times 10^2 + \ldots\ a_{n-1} \times 10^{n-1}$$

برای مثال عدد ۵۶۰۸۹ به صورت زیر تفسیر میشود.

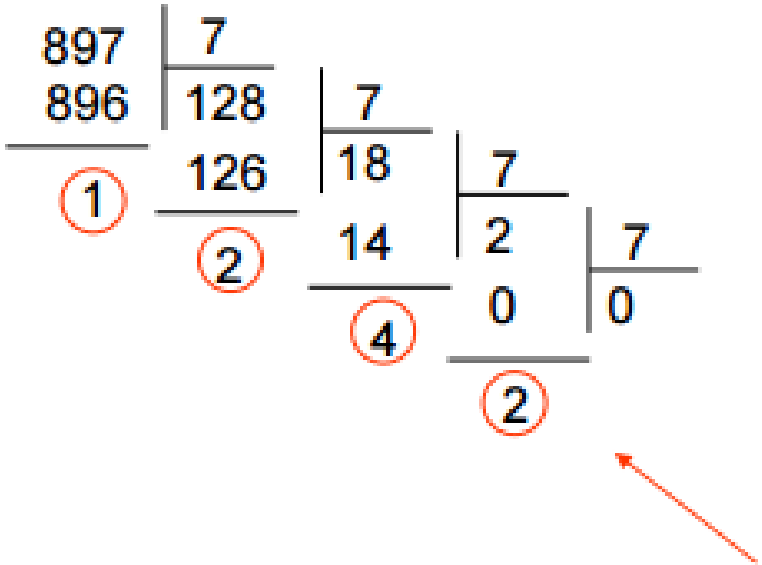
$$5 \times 10^0 + 6 \times 10^1 + 0 \times 10^2 + 8 \times 10^3 + 9 \times 10^4$$

تبدیل مبناها

برای تبدیل یک عدد از مبنای ۱۰به هر مبنای دلخواه x، از روش تقسیمات متوالی استفاده می گردد. بدین ترتیب که عدد مورد نظر بر x تقسیم می گردد و باقیمانده ذخیره می گردد. سپس همین عمل برروی خارج قسمت تقسیم انجام می شود و عملیات تا زمانیکه خارج قسمت به ۰ برسد ادامه پیدا می کند. در پایان باقیمانده های ذخیره شده به ترتیب از آخرین باقیمانده تا اولین باقیمانده به ترتیب از چپ به راست نوشته می شوند و عددحاصل از این فرایند عدد مورد نظر در مبنای x است.

تبدیل مبناها

در زیر روند تبدیل مبنای عدد $(897)_{10}$ به مبنای ۷ نشان داده شده است



$$(897)_{10} = (2421)_{7}$$

تبدیل مبناها

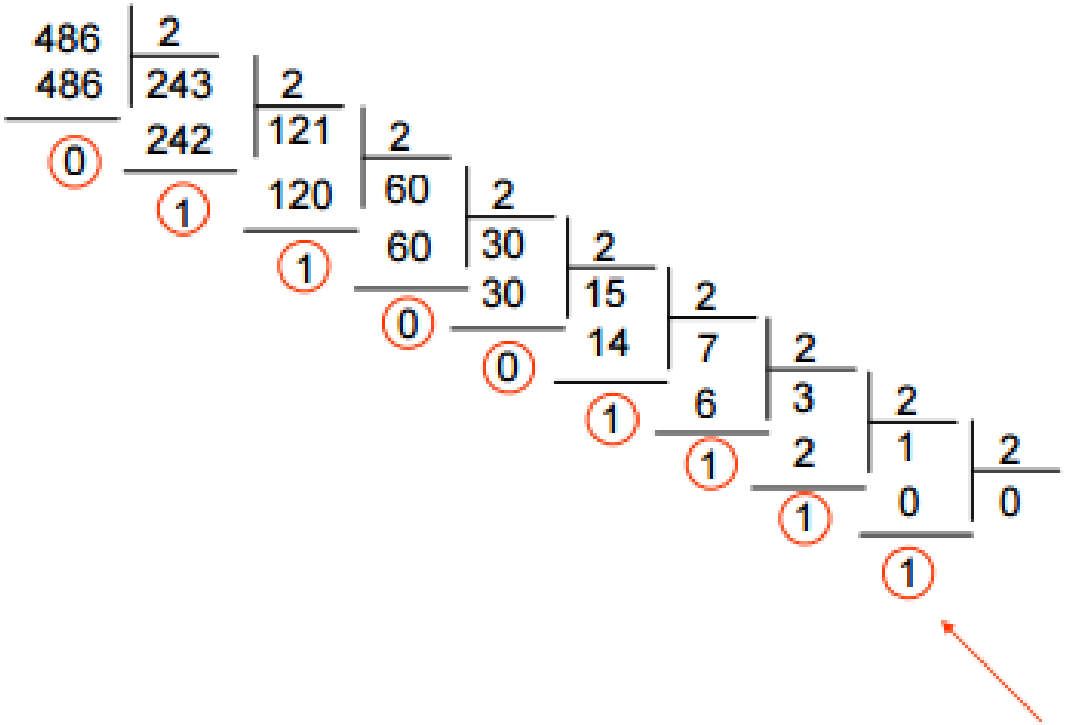
برای تبدیل از مبناهای دیگر به مبنای ۱۰باید اعداد را در ارزش مکانی خود ضرب کنیم و حاصل ضرب ها را با هم جمع کنیم. عدد حاصل آن عدد در مبنای ۱۰است. برای مثال بالا داریم:

$$(2421)_7 = 2 \times 7^3 + 4 \times 7^2 + 2 \times 7^1 + 1 \times 7^0 = 686 + 196 + 14 + 1 = 897$$

همانطور که قبلا نیز گفته شد واحد نگهداری اطلاعات در کامپیوتر بیت می باشد که هر بیت قادر به نگهداری ۰ و یا ۱ است. با کنار هم قرار دادن بیتها، بایتها تشکیل می گردند و بدینوسیله اطلاعات مورد نظر در قالب بایتها تشکیل می گردند.تبدیل اعداد از مبنای ۱۰به ۲ و بالعکس بسیار ساده و همانند سایر مبنا ها است.

تبدیل مبناها

در زیر عدد $(486)_{10}$ را به مبنای ۲ تبدیل می کنیم.



$$(486)_{10} = (111100110)_2$$

تبدیل مبناها

برای تبدیل عدد $(111100110)_2$ به مبنای ۱۰ همانند سایر مبنا که در قبل توضیح دادیم عمل می کنیم.

$$(111100110)_2 = 0\times 2^0 + 1\times 2^1 + 1\times 2^2 + 0\times 2^3 + 0\times 2^4 + 1\times 2^5 + 1\times 2^6 + 1\times 2^7 + 1\times 2^8$$

$$= 0+2+4+0+0+32+64+128+256=486$$

تبدیل مبناها

یک عدد در مبنای ۸ از مجموعه ارقام {0,1,2,3,4,5,6,7} تشکیل شده است. هر رقم در مبنای هشت از ۳ رقم دودویی تشکیل شده است .

مبنای ۸	۰	۱	۲	۳	۴	۵	۶	۷
مبنای ۲	۰۰۰	۰۰۱	۰۱۰	۰۱۱	۱۰۰	۱۰۱	۱۱۰	۱۱۱

در زیر تبدیل عدد $(1011111)_2$ به مبنای ۸ نشان داده شده است.

$$\underbrace{0}_{2}\underbrace{100}_{3}\underbrace{111}_{7} = (237)_8$$

تبدیل مبناها

مبنای شانزده یکی دیگر از مبناها است که بسیار مورد استفاده قرار می گیرد.مبنای شانزده همانند مبنای هشت است با این تفاوت که هر ۴ رقم در مبنای دو یک عدد در مبنای شانزده است. مشکلی که در نمایش عدد در مبنای شانزده وجود دارد این است که در این مبنا نیاز به ۱۶رقم داریم درحالیکه ارقام موجود فقط ۱۰تا است. بهمین دلیل از حروف A تا Fبرای ارقام ۱۰تا ۱۵استفاده می گردد.یعنی ارقام عبارتند از :

0 1 2 3 4 5 6 7 8 9 A B C D E F

مبنای ۱۶	۰	۱	۲	۳	۴	۵	۶	۷	۸	۹	A	B	C	D	E	F
مبنای ۲	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111

تبدیل مبناها

برای تبدیل اعداد از مبنای ۲به ۱۶ازهمان روش گفته شده برای مبنای ۸ استفاده می نماییم با این تفاوت که از سمت راست بصورت دسته های ۴ تایی جدا می کنیم و برای دسته آخر اگر تعدادشان کمتر از ۴بود به سمت چپ آن ۰اضافه می کنیم تا تعدادشان به ۴برسد.

در زیر تبدیل عدد $(1010011101)_2$ به مبنای 16 را نشان می دهد

$$\underset{2}{0\,0\,1\,0} \quad \underset{9}{1\,0\,0\,1} \quad \underset{D}{1\,1\,0\,1} = (29D)_{16}$$

تبدیل مبناها

برای تبدیل از مبنای ۸ به مبنای ۲ بازای هر رقم مبنای ۸، ۳ رقم در مبنای ۲ قرار می دهیم و برای تبدیل از مبنای ۱۶ به مبنای ۲ بازای هر رقم مبنای ۱۶، ۴ رقم در مبنای ۲ قرار می دهیم.

در مثال زیر عدد $(367)_8$ و عدد $(3BF)_{16}$ را به مبنای ۲ تبدیل می کنیم.

$(3\ 6\ 7)_8 = (11110111)_2$

011 110 111

$(3\ B\ F)_{16} = (1110111111)_2$

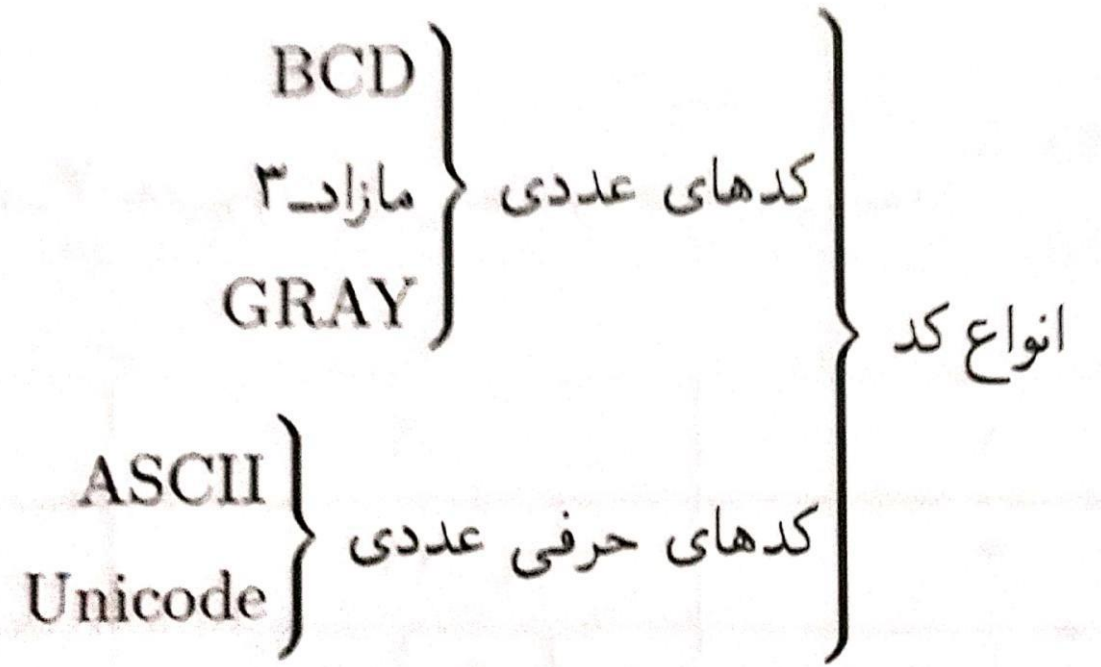
0011 1011 1111

تبدیل مبناها

دهدهی	باینری	هگزا دسیمال
۰	۰۰۰۰	۰
۱	۰۰۰۱	۱
۲	۰۰۱۰	۲
۳	۰۰۱۱	۳
۴	۰۱۰۰	۴
۵	۰۱۰۱	۵
۶	۰۱۱۰	۶
۷	۰۱۱۱	۷

دهدهی	باینری	هگزا دسیمال
۸	۱۰۰۰	۸
۹	۱۰۰۱	۹
۱۰	۱۰۱۰	A
۱۱	۱۰۱۱	B
۱۲	۱۱۰۰	C
۱۳	۱۱۰۱	D
۱۴	۱۱۱۰	E
۱۵	۱۱۱۱	F

انواع کد


انواع کد
کدهای عددی
BCD
مازاد_۳
GRAY
کدهای حرفی عددی
ASCII
Unicode

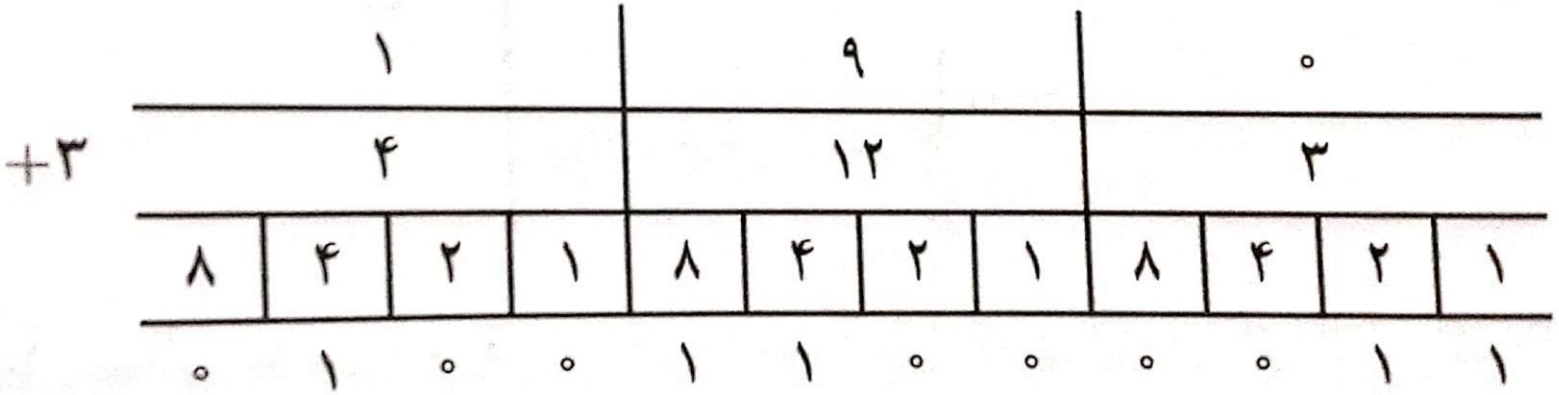
کد BCD (Binary Coded Decimal)

در کد BCD یا کد «نمایش اعداد دهدهی به‌صورت کدهای دودویی» برای هر رقم دهدهی یک عدد دودویی چهاربیتی در نظر گرفته می‌شود و هر رقم به یک چهاربیتی تبدیل می‌شود. ارزش مکانی ارقام در این نوع کد کردن می‌تواند متفاوت باشد مثلاً ۱، ۲، ۴، ۸ یا ۱، ۲، ۲، ۴ و .... اگر ارزش این چهار رقم ۱، ۲، ۴، ۸ باشد آن را NBCD یا Natural BCD می‌نامند.

۲				۳				۷			
۸	۴	۲	۱	۸	۴	۲	۱	۸	۴	۲	۱
۰	۰	۱	۰	۰	۰	۱	۱	۰	۱	۱	۱

انواع کد

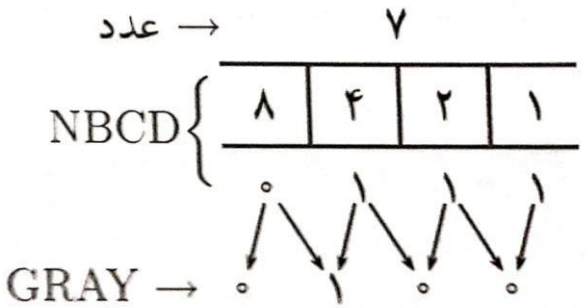
کد مازاد ۳ (EXCESS-3)



انواع کد

کد گری(GRAY)

کد گری دارای این خاصیت است که هر کدی با کد قبلی و بعدی فقط در یک بیت با هم اختلاف دارند. این نوع کدها را انعکاسی می‌نامند. به این ترتیب اگر بخواهیم از یک کد (عدد) به عدد بعدی برویم فقط کافی است یک بیت تغییر کند و اگر به اشتباه بیش از یک بیت تغییر کند و خطایی ایجاد شود به‌راحتی قابل کشف است.
این کد را می‌توان در تعداد ارقام مختلفی تعریف کرد که در اینجا ۴ بیتی آن را بررسی می‌کنیم.



فصل دوم

الگوریتم و فلوچارت

الگوریتم

تعریف عامیانه ای که می شود برای الگوریتم ارائه کرد روال خاص انجام کاری است. الگوریتم در لغت به معنای روش حل مسائل می باشد و از نام دانشمند و ریاضی دان ایرانی ، ابو جعفربن محمد خوارزمی گرفته شده است. برای اینکه کامپیوترها بتوانند یک کار را بطور کامل انجام دهند باید تمام مراحل انجام آن کار بطور دقیق برایشان مشخص شده باشد. تعریف دقیقی که می شود برای الگوریتم ارائه کرد این است که: " به مجموعه ای از دستورالعمل ها که مراحل مختلف (برای حل یک مسئله) را به زبان قطعی و با جزئیات کافی بیان کرده و در آن ترتیب مراحل و خاتمه پذیر بودن عملیات در آن کاملا مشخص باشد را الگوریتم می نامند."

الگوریتم

در هنگام نوشتن هر الگوریتم باید به نکات زیر توجه کرد :

- **ورودی** : یک الگوریتم می تواند صفر یا چند ورودی داشته باشد که از محیط خارج تامین می گردد.
- **خروجی** : الگوریتم باید یک یا چند خروجی داشته باشد.
- **زبان دقیق**: اگر از یک جمله برداشت های متفاوتی شود و یا اینکه جمله مبهم باشد ، گوییم بیان آن جمله یا عبارت دقیق نیست.

الگوریتم

- **جزئیات کافی:** عملیات ذکر شده در یک الگوریتم باید برای مجری آن شناخته شده باشد. نویسنده الگوریتم باید از چیزهایی که مجری الگوریتم(زبان برنامه نویسی مورد نظر) می داند آگاه باشد و هیچگونه فرضی را در مورد مجری الگوریتم منظور نکند.

- **ترتیب مراحل :** مراحل انجام دستورالعمل ها باید به ترتیب انجام گیرد و در غیر اینصورت دستورالعمل ها قابل اجرا نخواهد بود.

- **کارایی :** هر دستورالعمل باید قابل اجرا باشد.

- **خاتمه عملیات:** شرط یا شروط خاتمه عملیات باید در الگوریتم ذکر شود بخصوص وقتی که عملیات تکراری هستند.

الگوریتم

معمولا برای حل یک مسئله، مراحل زیر طی می گردد:

- تعریف مسئله بصورت جامع و دقیق (شامل تعریف ورودی ها و خروجی ها)
- بررسی راه حل های مختلف برای حل مسئله
- انتخاب مناسبترین راه حل و تهیه یک الگوریتم برای آن
- آزمایش الگوریتم با داده های ورودی و اشکال زدایی آن
- تبدیل الگوریتم به یک زبان برنامه نویسی کامپیوتری
- وارد کردن برنامه به کامپیوتر و تست و اشکال زدایی آن
- استفاده از برنامه

قرارداد کار با الگوریتم

در زیر قراردادهایی وجود دارند که در هنگام نوشتن الگوریتم باید به آن توجه کرد:

- در ابتدای هر الگوریتم کلمه شروع و در انتهای آن از کلمه پایان استفاده می کنیم.
- برای هر یک از دستورالعمل ها شماره ای در نظر می گیریم.
- برای دریافت اطلاعات از کاربر از دستور بخوان استفاده می گردد.
- برای نوشتن اطلاعات در خروجی از دستور چاپ کن استفاده می گردد.

الگوریتم

- برای انجام محاسبات ریاضی یا کار بر روی داده ها از داده، مکانی برای ذخیره داده ها و اضافه کردن نتایج در نظر می گیریم که در کامپیوتر این مکان ها در خانه های حافظه ذخیره می شوند. برای اینکه بتوانیم بعدا به این خانه های حافظه مراجعه کنیم، به هریک از آنها یک نام نسبت می دهیم. به هریک از این نامها یک متغیر گفته می شود. دلیل این نامگذاری آنست که مقادیر ذخیره شده در هریک از این خانه های حافظه می تواند تغییر کند. نام متغیر ها بصورت دلخواه در نظر گرفته می شود ولی بهتر است نام آن متناسب با کاربرد باشد.

- برای انتساب یک مقدار به یک متغیر از علامت ← استفاده می شود.

الگوریتم

مثال۱ : الگوریتمی بنویسید که ۳ عدد را از کاربر دریافت کند و میانگین آنها را چاپ کند.

۱. شروع
۲. A،B و C را بخوان
۳. S ← A+B+C
۴. Average ←S/3
۵. Average را چاپ کن
۶. پایان

الگوریتم

جملات شرطی گونه ای از جملاتی که در هنگام الگوریتم نویسی از آنها استفاده می کنیم جملات شرطی هستند که شرط یا شرایطی خاصی را چک می کنند که در صورت برقراری آن عملیات خاصی را انجام دهند و در صورت عدم برقراری عملیات دیگری را انجام دهند.

از این رو دو نوع جمله شرطی داریم:

- شرطی نوع ساده
- شرطی نوع دو

الگوریتم

✓شرطی نوع ساده:

شکل کلی دستورات شرطی نوع ساده بصورت زیر است:

اگر (عبارت شرطی) آنگاه دستورات

و بدین صورت تعبیر می شود که اگر عبارت شرطی درست باشد آنگاه قسمت دستورات اجرا خواهد شد. اما در غیر اینصورت به دستور بعدی خواهد رفت.

✓شرطی نوع دو:

شکل کلی این نوع شرط به صورت زیر است:

اگر (عبارت شرطی) آنگاه دستورات ۱

درغیر اینصورت دستورات ۲

اگر عبارت شرطی درست باشد دستورات ۱ اجرا می شود و اگر درست نباشد دستورات ۲ اجرا خواهند شد.

الگوریتم

الگوریتمی بنویسید که عددی را دریافت کند و اگر عدد زوج بود کلمه زوج و اگر فرد بود کلمه فرد را در خروجی چاپ کند.

۱. شروع کن

۲. عدد A را دریافت کن

۳. $B \leftarrow A/2$

۴. $C \leftarrow A-2*B$

۵. اگر C برابر با صفر بود آنگاه در خروجی زوج را چاپ کن در غیر اینصورت فرد

۶. پایان

الگوریتم

الگوریتمی بنویسید که ضرایب یک معادله درجه دوم بصورت $x^2 + b\,x + c = 0$ را دریافت و ریشه های آن را در صورت وجود محاسبه و چاپ کند.

۱. شروع

۲. a و b و c را بخوان

۳. اگر (a=0) آنگاه چاپ کن "معادله درجه دو نیست" و به مرحله پایان برو.

۴. delta ← b2 – 4ac

۵. اگر (delta<0) آنگاه چاپ کن "معادله جواب ندارد"

در غیر اینصورت $x_1 \leftarrow \frac{-b-\sqrt{delta}}{2a}$ و $x_2 \leftarrow \frac{-b+\sqrt{delta}}{2a}$ x_1 و x_2 را چاپ کن

۶. پایان

الگوریتم

الگوریتمی بنویسید که ۳ عدد را دریافت کند و مشخص کند که می توان با این ۳ عدد مثلث ساخت یا خیر همانطور که می دانید شرط اینکه سه عدد a، b وc تشکیل یک مثلث بدهند این است که روابط زیر برقرار باشد:

$a <= b + c$

$b <= a + c$

$c <= b + a$

۱. شروع

۲. a ، b و c را بخوان

۳. اگر $a <= b + c$ آنگاه به خط ۴ برو در غیراینصورت به خط ۷ برو

۴. اگر $b <= a + c$ آنگاه به خط ۴ برو در غیراینصورت به خط ۷ برو

۵. اگر $c <= b + a$ آنگاه به خط ۴ برو در غیراینصورت به خط ۷ برو

۶. چاپ کن می توان یک مثلث ساخت

۷. چاپ کن نمی توان یک مثلث ساخت

۸. توقف

الگوریتم

حلقه های تکرار

برای حل مسائل گاهی اوقات لازم است که عملیات مشخصی چندین بار تکرار شوند. به عنوان مثال برای نوشتن الگوریتمی که ۵۰ عدد را از کاربر دریافت کند تا میانگین آنها را حساب کند نباید ۵۰ بار" دستور عدد را بخوان" و" آن را با عددهای خوانده شده جمع کن " را نوشت. راه حل بهتر آنست که بگونه ای به الگوریتم بگوییم به نحوی عمل خواندن اعداد و جمع زدن آنها را ۵۰ بار تکرار کند.

حلقه تکرار به گونه ای است که مجموعه ای از دستورات را تا زمانیکه شرط خاصی برقرار باشد تکرار می کند.

الگوریتم

بطور کلی حلقه های تکرار به دو شکل مورد استفاده قرار می گیرد :

- حلقه های شرطی که شرط آنها در ابتدای حلقه چک می شوند و دارای ساختار زیر هستند:

تا زمانیکه (شرط مورد نظر) دستورات a تا bرا تکرارکن

(a …

.

.

.

(b …

در این حالت ابتدا شرط موردنظر بررسی می گردد؛ درصورتیکه شرط برقرار نباشد به اولین دستور پس از b می رود. اما در صورتیکه شرط درست ارزیابی شود، دستورات شماره a تا b انجام می شوند و سپس مجددا به ابتدای حلقه بازگشته و عملیات فوق را مجددا تکرار می کند.

الگوریتم

- حلقه های شرطی که شرط در انتهای حلقه چک می شود و دارای ساختار زیر هستند :

تکرار کن

a) ...

.

.

.

b) ...

تا زمانیکه (شرط مورد نظر)

در این روش ابتدا دستورات حلقه یکبار انجام می شوند و در پایان حلقه شرط بررسی می گردد. چنانچه شرط برقرار نبود به دستور بعدی می رود و در صورت برقرار بودن شرط، مجددا به ابتدای حلقه باز می گردد.

الگوریتم

الگوریتمی بنویسید که nعدد را به دلخواه از ورودی دریافت کند و حاصل جمع آنها را چاپ نماید.

۱. شروع

۲. عدد n را بخوان

۳. Sum ← ۰

۴. i ← ۱

۵. تا زمانیکه i<=n است دستورات ۶ تا ۸ را تکرار کن

۶. A را بخوان

۷. Sum ← Sum + A

۸. i ← 1+i

۹. Sum را چاپ کن

۱۰. توقف

الگوریتم

الگوریتمی بنویسید که یک عدد را دریافت و فاکتوریال آن را محاسبه و چاپ کند.

$N! = 1 \times 2 \times 3 \times \ldots \times (N-1) \times N$

مقدار فاکتوریل : fact　　شمارنده حلقه : i　　عدد ورودی برای محسبه فاکتوریل آن : n

۱) شروع

۲) n را بخوان

۳) ۱ ← i و ۱ ← fact

۴) تا زمانیکه ($i \leq n$) دستورات 5-6 را تکرار کن

۵) $fact \leftarrow fact \times i$

۶) $i \leftarrow i + 1$

۷) fact را چاپ کن

۸) توقف کن

الگوریتم

الگوریتمی بنویسید که یک عدد را دریافت و وارون آن را محاسبه و به همراه خود عدد چاپ کند.

مثال: وارون عدد ۱۳۹۹برابر ۹۹۳۱می باشد.

متغیر کمکی برای انجام عملیات برروی عدد مورد نظر : a عدد مورد نظر : adad

وارون عدد موردنظر : varoon مقدار باقیمانده تقسیم عدد بر ۱۰ در هر مرحله : remain

الگوریتم

۱. شروع

۲. adadرا بخوان

۳. اگر (adad < 0) آنگاه a ← -adad در غیر اینصورت a ← adad

۴. varoon ← 0

۵. تازمانیکه (a > 0) آنگاه دستورات ۶ تا ۸ را تکرار کن

۶. remain ← a mod 10

۷. a ← a / 10

۸. varoon ← varoon × 10 + remain

۹. اگر (adad < 0) آنگاه varoon ← -varoon

۱۰. adadو varoon را چاپ کن

۱۱. توقف کن

الگوریتم

الگوریتمی بنویسید که یک عدد صحیح مثبت در مبنای ۱۰ را دریافت و سپس آن را به مبنای bببرد. مبنای bنیز از کاربر دریافت می گردد.

مبنای موردنظر : b	عدد موردنظر : adad
حاصل تبدیل عدد به مبنای موردنظر : mabnaieb	
باقیمانده در هر مرحله : remain	شمارنده حلقه : i

۱. شروع

۲. adadو b را بخوان

۳. mabnaieb ← 0وi ← 0

۴. تازمانیکه (adad > 0) دستورات 5 تا 8 را تکرار کن

۵. remain ← adad mod b

۶. mabnaieb ← mabnaieb + remain × 10^{i}

۷. adad ← adad / b

۸. i ← i + 1

۹. mabnaiebرا چاپ کن

۱۰. توقف کن

الگوریتم

الگوریتمی بنویسید که برای دانشجویان ورودی ۹۹دانشکده ریاضی، نام و شماره دانشجویی و تعداد دروس را دریافت کند و سپس برای هر دانشجو نمره و تعداد واحد هر درس وی را دریافت و معدل وی را محاسبه نماید. در پایان میانگین معدل دانشکده و بیشترین و کم ترین معدل دانشکده را به همراه شماره دانشجویی آنها چاپ کند.

n : تعداد دانشجویان ورودی ۹۹ i : شمارنده دانشجویان dars: تعداد دروس هر دانشجو

j : شمارنده دروس vahed : تعداد واحد هر درس total Ave: معدل کل دانشجویان

maxAve: بیشترین معدل minAve: کمترین معدل totalVahed: تعداد کل واحدهای هر دانشجو

Sum: جمع نمره هر دانشجو moadel: معدل هر دانشجو maxStudentID: شماره دانشجویی بیشترین معدل

minStudentID: شماره دانشجویی کمترین معدل nomre: نمره درس StudentID: شماره دانشجویی

الگوریتم

۱ شروع

۲ n را دریافت کن

۳ i ← ۱

۴ total Ave ← ۰ ، maxAve ← -۱ ، minAve ← ۱ ، maxStudentID ← ۰ ، minStudentID ← ۰

۵ تا زمانیکه i<=n دستورات ۶ تا 17 را تکرار کن

۶ Sum ← ۰ ، moadel ← 0 ، totalVahed ← 0

۷ dars و StudentID را دریافت کن

۸ j ← ۱

۹ تا زمانیکه j<=dars دستورات ۱۰ تا۱۳ را تکرار کن

۱۰ Vahed و nomre را دریافت کن

۱۱ sum ← Vahed×nomre + sum

الگوریتم

۱۲ totalVahed ← totalVahed+ vahed

۱۳ j ← j+1

۱۴ moadel ← sum/totalVahed

۱۵ totalAve ← totalAve+moadel

۱۶ اگر maxAve <moadel باشد آنگاه maxAve ← moadel و StudentID ← maxStudentID در غیر

اینصورت اگر moadel<minAve باشد آنگاه minAve ← moadel و StudentID ← minStudentID

۱۷ i ← i+1

۱۸ totalAve ← totalAve/n

۱۹ totalAve رو چاپ کن.

۲۰ maxAve رو به همراه maxStudentID چاپ کن

۲۱ minAve رو به همراه minStudentID چاپ کن

۲۲ توقف

الگوریتم

تمرین:

۱. الگوریتمی بنویسید که nعدد صحیح را دریافت و حداکثر و حداقل آنها را چاپ کند.

۲. الگوریتمی بنویسید که عدد طبیعی nرا دریافت کند و مجموع عبارات زیر را محاسبه و چاپ نماید:

$$s = 1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \ldots + \frac{1}{n}$$

۳. الگوریتمی بنویسید که یک عدد را دریافت و تعیین کند که آیا اول است یا خیر؟

فلوچارت

در گذشته برای درک بهتر الگوریتم ها و سهولت دنبال کردن دستورالعمل های آن از یکسری اشکال خاص برای نشان دادن روند الکوریتم ها استفاده می کردند که به آن فلوچارت گفته می شود. در واقع فلوچارت مجموعه ای از علائم ساده است که الگوریتم را به صورت نمادهای تصویری نمایش می دهد.

فلوچارت

- **علامت شروع و پایان در فلوچارت**

از شکل بیضی برای نمایش شروع و پایان الگوریتم استفاده می کنیم

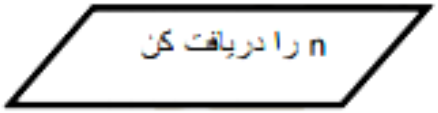
- **علامت جایگزینی و انتساب در فلوچارت**

برای انجام عمل جایگزینی و یا انتساب و یا عمل محاسباتی از مستطیل استفاده می شود

فلوچارت

- **علامت ورودی در فلوچارت**

از علامت متوازی الاضلاع برای گرفتن ورودی ها استفاده می شود



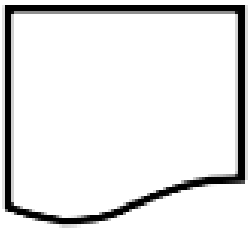
- **علامت شرط در فلوچارت**

از علامت لوزی برای شرط استفاده می شود

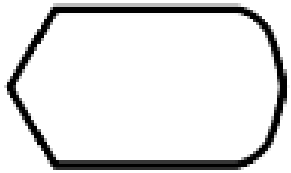
فلوچارت

- **علامت چاپ در فلوچارت**

از علائم زیر برای چاپ استفاده می شود



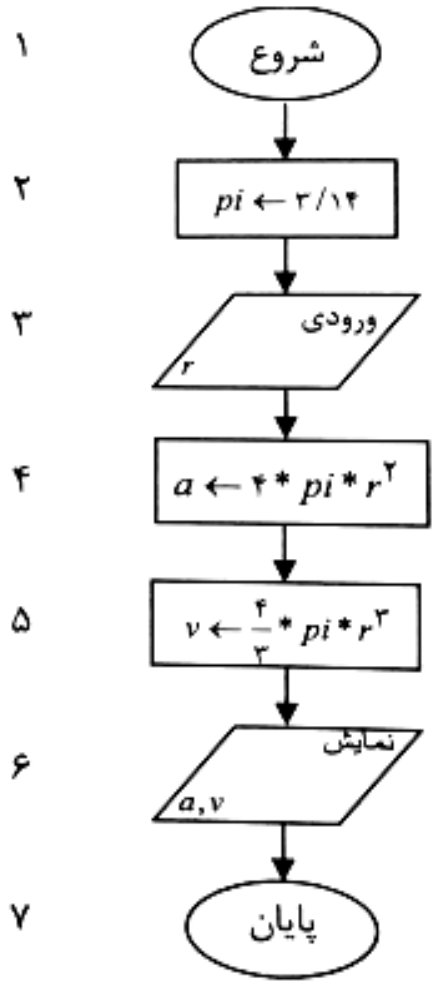
چاپ روی کاغذ



چاپ روی صفحه نمایشگر

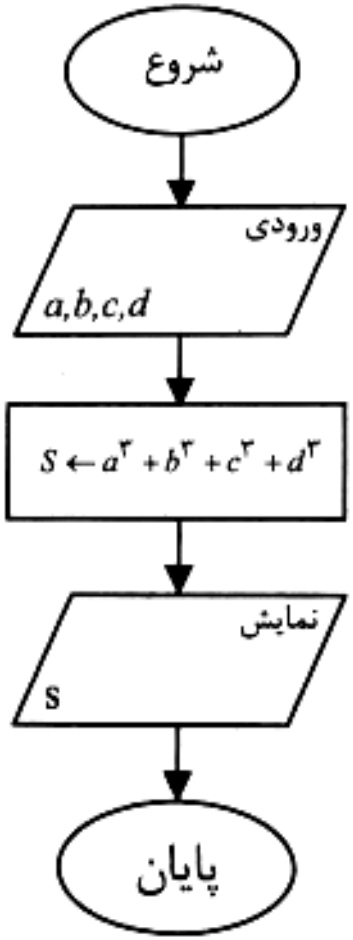
فلوچارت

فلوچارتی بنویسید که شعاع کره ای را بعنوان ورودی دریافت کند،مساخت و حجم کره را محاسبه کرده و نمایش دهد.



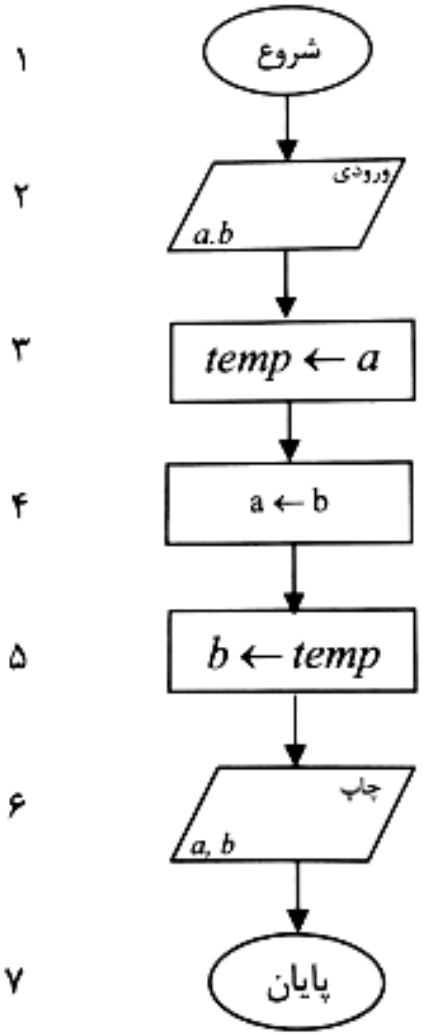
فلوچارت

فلوچارتی بنویسید که مقادیر چهار متغیر a,b,c و d را خوانده، مجموع مکعبات آنها را محاسبه کرده و نمایش دهد.



فلوچارت

فلوچارتی بنویسید که مقادیر دو متغیر را از ورودی خوانده، سپس مقادیر آنها را با یکدیگر تعویض نماید و نتیجه را چاپ کند.



آشنایی با C++

C++ یک زبان portable است. زبان های قابل حمل یا مستقل از متن زبان هایی هستند که به هیچ ماشین خاصی وابسته نیستند، برنامه های نوشته شده با این زبان ها تا **حد زیادی** قابل حمل می باشند. در مقابل زبان های قابل حمل زبان های غیر قابل حمل وجود دارند که این زبان ها وابسته به سخت افزاری که برروی آن نوشته شده اند می باشند که از جمله این زبان ها می توان به زبان ماشین و اسمبلی اشاره کرد.

همانطور که گفتیم C ++توسعه یافته زبان Cاست که علاوه بر ویژگی های زبان Cدارای خاصیت شی گرایی نیز هست.

نحوه اجرای یک برنامه در C++

همانطور که گفتیم برای اینکه الگوریتمی که برای حل یک مسئله طرح کردیم برای اینکه توسط یک سیستم کامپیوتری اجرا شود نیاز دارد آن را ابتدا با استفاده از یک زبان برنامه سازی که اکثرا از زبانهای سطح بالا استفاده می شود نوشته شود که به این برنامه در این مرحله کد اصلی گفته می شود و پس از اینکه این برنامه به زبان ماشین برگردانده شد توسط کامپیوتر قابل فهم و اجرا است.

اجرای یک برنامه به زبان C ++شامل ۶ مرحله است:

آشنایی با C++

۱. مرحله تولید برنامه:

در این مرحله برنامه توسط یکی از برنامه های ادیتوری به زبان C++ نوشته می شود و بعد از این مرحله برنامه با پسوند های .cc ،.cxx ،.cpp یا C در یکی از مکان های ذخیره سازی اطلاعات ذخیره می شود.

۲. پیش پردازش

پیش پردازنده ها شامل دستوراتی هستند که توسط کامپایلر قبل از شروع به کامپایل انجام می شوند. هر دستوری در C++ که با علامت # شروع شود جزو دستورات پیش پردازنده هستند و زمانی که بخواهیم از کتابخانه خود C++ استفاده کنیم از پیش پردازنده ها استفاده می کنیم،

آشنایی با C++

۳. کامپایل:

در این مرحله کامپایلر دستورات کد اصلی را به زبان ماشین ترجمه می کند.

۴. پیوند دهنده :

هنگام برنامه نویسی در C ++این امکان وجود دارد که از توابع کمکی نوشته شده در برنامه های دیگر استفاده کرد که این توابع باید به برنامه کد اصلی پیوند زده شود که با استفاده از پیونددهنده این امکان فراهم می شود.

۵. بارگذاری:

برای اینکه یک برنامه در کامپیوتر قابل اجرا باشد باید ابتدا در حافظه اصلی بارگذاری شود که اینکار با استفاده از بارگذار کننده صورت می گیرد.

۶. اجرا:

در مرحله آخر کامپیوتر با استفاده از CPUدستورات برنامه نوشته شده را اجرا می کند.

آشنایی با C++

کلیه مراحل گفته شده در بالا در یک IDEبطور کامل پس از اجرای برنامه انجام می گیرد. علاوه براین، محیط ها IDEمعمولا دارای امکانات اشکال زدایی برنامه (Debug) نیز می باشند که شامل مواردی همچون اجرای خط به خط برنامه و یا دیدن محتویات متغیرها در زمان اجرا است

مراحل گفته شده در بالا برای اجرای یک برنامه است ولی اکثر مواقع اجرای برنامه ها برای بار اول دارای خطاهای اجتناب ناپذیری هستند خطاها از لحاظ تاثیری که بر اجرای برنامه ها می گذارند، متفاوتند. گروهی ممکن است باعث شوند که از همان ابتدا برنامه اصلا کامپایل نشود و گروه دیگر ممکن است پس از گذشت مدتها و در اثر دادن یک ورودی خاص به برنامه، باعث یک خروجی نامناسب و یا یک رفتار دور از انتظار (مانند قفل شدن برنامه) شوند.

بطور کلی خطاها به دو دسته تقسیم می شوند:

۱. خطاهای نحوی (خطاهای زمان کامپایل):

این خطاها در اثر رعایت نکردن قواعد دستورات زبان C ++و یا تایپ اشتباه یک دستور بوجود می آیند و در همان ابتدا توسط کامپایلر به برنامه نویس اعلام می گردد. برنامه نویس باید این خطا را رفع کرده و سپس برنامه را مجددا کامپایل نماید. لذا معمولا این قبیل خطاها خطر کمتری را در بردارند.

۲. خطاهای منطقی (خطاهای زمان اجرا):

این دسته خطاها در اثر اشتباه برنامه نویس در طراحی الگوریتم درست برای برنامه و یا گاهی در اثر درنظر نگرفتن بعضی شرایط خاص در برنامه ایجاد می شوند. متاسفانه این دسته خطاها در زمان کامپایل اعلام نمی شوند و در زمان اجرای برنامه خود را نشان می دهند. بنابراین، این خود برنامه نویس است که پس از نوشتن برنامه باید آن را تست کرده و خطاهای منطقی آن را پیدا کرده و رفع نماید. متاسفانه ممکن است یک برنامه نویس خطای منطقی برنامه خود را تشخیص ندهد و این خطا پس از مدتها و تحت یک شرایط خاص توسط کاربر برنامه کشف شود. به همین دلیل این دسته از خطاها خطرناکتر هستند.

خود این خطاها به دو دسته تقسیم می گردند

a) خطاهای مهلک: در این دسته خطاها کامپیوتر بلافاصله اجرای برنامه را متوقف کرده و خطا را به کاربر گزارش می کند. مثال معروف این خطاها، خطای تقسیم بر صفر می باشد.

a) خطاهای غیرمهلک: در این دسته خطا، اجرای برنامه ادامه می یابد ولی برنامه نتایج اشتباه تولید می نماید. به عنوان مثال ممکن است دراثر وجود یک خطای منطقی در یک برنامه حقوق و دستمزد، حقوق کارمندان اشتباه محاسبه شود و تا مدتها نیز کسی متوجه این خطا نشود!

آشنایی با C++

در این قسمت برای آشنایی با زبان برنامه نویسی C ++یک نمونه برنامه ساده آن را قرار داده ایم که با استفاده از آن چندین نکته مهم در رابطه با برنامه نویسی به زبان C ++را توضیح می دهیم.

```
// This is a program in C++
#include <iostream>
/* allows program to
 output data to the screen */

using namespace std;

// function main begins program execution
 Void  main()
 {
        cout << "Welcome to C++!\n ";        // display message

  } // end function main
```

خروجی برنامه:

```
Welcome to C++!
```

آشنایی با C++

همانطور که در خروجی مثال بالا می بینید که نماد \n که در انتهای رشته قرار گرفته است در خروجی نمایش داده نشده است. وقتی (\) Backslashدر انتهای یک رشته قرار گیرد و کاراکتری که بعد آن قرار بگیرد یک کاراکتر خاص است.

```
#include <iostream> // allows program to output data to the screen
using namespace std;
// function main begins program execution
int main()
{
    cout << "Welcome\n to\n\n C++!\n";
} // end function main
```

خروجی برنامه:

```
Welcome
to

C++!
```

آشنایی با C++

بعضی کاراکتر ها هستند که به همراه \ عملیات خاصی را انجام می دهند که در جدول زیر آنها را توضیح می دهیم.

نماد	توضیحات
\n	باعث می شود کرسر به خط جدید برود
\t	باعث می شود موقعیت کرسر به اندازه یک tab جلوتر رود
\a	بوق کامپیوتر
\\	کاراکتر \ را چاپ می کند
\'	کاراکتر ' را چاپ می کند
\"	کاراکتر " را چاپ می کند

آشنایی با C++

متغیر: نامی است که به یک قسمت از برنامه مانند متغیر، تابع، ثابت و یا ... داده می شود. در زبان C ++برای انتخاب شناسه ها فقط می توان از علائم زیر استفاده کرد:

- حروف انگلیسی کوچک و بزرگ (A...Z a...z)
- ارقام (0,1,...,9)
- علامت خط پایین یا_

البته یک متغیر نمی تواند با یک رقم شروع شود. مثال شناسه های number_one، number1 ، number مجاز هستند اما شناسه های number one، number1 مجاز نیستند. البته در برنامه نویسی امروزی پیشنهاد می شود بجای متغیرهایی همانند number_one از numberOne استفاده گردد. یعنی بجای اینکه در شناسه ها از _ برای جداکننده استفاده کنیم، اولین حرف هر قسمت را بصورت بزرگ بنویسیم.

آشنایی با C++

این مسئله معمولا درهنگام برنامه نویسی باعث ایجاد بعضی خطاها می شود.

نکته: زبان C++ هیچ محدودیتی در طول شناسه ها قرار نداده است

نکته: بهتر است از شناسه های با معنی در برنامه هایمان استفاده کنیم

نکته: بهتر است از اختصار استفاده نکنیم تا درک عملکرد برنامه برای دیگران قابل فهم باشد.

نکته: در هنگام انتخاب شناسه نمی توانید از کلمات کلیدی که برای منظورهای خاص در زبان C++ رزرو شده اند استفاده کنید.

داده ها

همانطور که در بخش های قبل بیان کردیم؛ یک متغییر مکانی از حافظه است که یک مقدار می تواند در آن ذخیره شود. هر متغیر پیش از آنکه دریک برنامه استفاده گردد ابتدا باید اعلان گردد. اعلان یک متغیر بدین معنی است که نوع آن متغییر را مشخص شود.

نوع داده	توضیح	اندازه(بیت)	دامنه
char	کاراکتر	۸	-۱۲۸ تا ۱۲۷
int	عدد صحیح	۱۶	-32768 تا ۳۲۷۶۷
float	عدد اعشاری	۳۲	3.4e-38 تا 3.4e+38
double	عدد اعشاری با دقت مضاعف	۶۴	1.7e-308 تا 1.7e308

تعریف متغیر

برای تعریف متغیرها به شکل زیر عمل می کنیم:

<type> <variable-list> ;

که type یکی از نوع داده های گفته شده و variable-list لیستی از متغیرها است که با کاما از یکدیگر جدا شده اند. بعنوان مثال :

```
int   number;
float  average;
double   a, b, c ;
```

علاوه براین می توان در هنگام تعریف متغیر به آن مقدار اولیه نیز داد. مثال :

```
int  d = 0 ;
```

عملگر ها

عملگر، نمادی است که به کامپایلر می گوید تا عملیات محاسباتی یا منطقی خاصی را برروی یک یا چند عملوند، انجام دهد. به عملگرهایی که فقط یک عملوند دارند، عملگر یکانی می گوییم و همواره عملگر در سمت چپ عملوند قرار می گیرد(مانند عدد -125). اما عملگرهایی که برروی دو عملوند اثر می کنند را عملگر دودویی نامیده و عملگر را بین دو عملوند قرار می دهیم (مانند 23+86). هر ترکیب درستی از عملگرها و عملوندها را یک عبارت می نامیم.

عملگرها به چند دسته اصلی تقسیم می گردند که آنها را به ترتیب بررسی می کنیم.

عملگر محاسباتی

عمل در C++	عملگر در C++	نوع عمل	عمل جبری	مثال
منفی کردن	-	یکانی	-b	-4
جمع	+	دودویی	a + b	1+3
تفریق	-	دودویی	a - b	1-3
ضرب	*	دودویی	a * b	1*3
تقسیم	/	دودویی	a / b	1/3
باقی مانده	%	دودویی	a % b	1%3

عملگر محاسباتی

عملگرهای فوق همه اعمال متداول ریاضی هستند که برروی همه انواع داده های C عمل می کنند بجز عملگر % که فقط برروی نوع داده های صحیح عمل می کند و برروی داده های اعشاری تعریف نشده است. اما نکته مهمی که باید به آن اشاره کرد، نحوه کار عملگر تقسیم برروی نوع داده های مختلف است. درصورتیکه هردو عملوند صحیحی باشند، تقسیم بصورت صحیح بر صحیح انجام خواهد شد. اما اگر یکی یا هر دو عملوند اعشاری باشند، تقسیم بصورت اعشاری انجام خواهد پذیرفت.

$c * (a + d)$

$(a / (w - x))$ پرانتز تو در تو

نکته بسار مهم دیگری که باید در هنگام کار با عبارات محاسباتی به آن توجه کرد، اولویت عملگرها است. یعنی در عبارتی که شامل چندین عملگر است، کدامیک در ابتدا اعمال خواهد گردید. در جدول زیر اولویت عملگرهای محاسباتی از بالا به پایین نشان داده ایم.

عملگر محاسباتی

توضیحات	عمل	عملگر در C++
اولویت اول	عملگر یکانی	-
اولویت دوم	پرانتز	( )
اگر عبارت شامل پرانتزهای تو در تو باشد ابتدا داخلی ترین پرانتز محاسبه می شود		
اولویت سوم	ضرب، تقسیم و باقی مانده	*, /, %
اگر این ها باهم در یک عبارت محاسباتی قرار بگیرند از چپ به راست ارزیابی می شوند		
اولویت آخر	جمع و تفریق	+, -
اگر این ها باهم در یک عبارت محاسباتی قرار بگیرند از چپ به راست ارزیابی می شوند		

عملگر محاسباتی

چنانچه اولویت دو عملگر یکسان باشد، این عملگرها از چپ به راست محاسبه خواهند شد. چنانچه بخواهیم یک عمل با اولویت پایین زودتر انجام شود، باید از پرانتز استفاده کنیم. در مورد پرانتزهای متداخل، ابتدا پرانتز داخلی محاسبه می شود؛ اما در مورد پرانتزهای هم سطح، ابتدا پرانتز سمت چپتر محاسبه می گردد.

```
a=5;
b = c + 2 * d;
```

این عملگر باعث می شود که عبارت سمت راست در عبارت سمت چپ قرار گیرد. توجه کنید که مقدار سمت چپ باید عبارتی باشد که بتوان به آن یک مقدار را نسبت داد(مانند یک متغیر) بنابراین یک ثابت نمی تواند در سمت چپ قرار گیرد. نکته دیگر اینکه اولویت عملگر = از عملگرهای ریاضی کمتر است و درنتیجه ابتدا آن عملیات انجام شده و در پایان حاصل در عبارت سمت

عملگر محاسباتی

چپ ریخته می شود. لازم به ذکر است که در هنگام انتساب، درصورت لزوم نوع عبارت سمت راست به نوع عبارت سمت چپ تبدیل می شود. مثال:

```
int  a;
a = 2.5 * 5.0;
```

که دراینصورت عدد ۱۲ در a ذخیره خواهد شد.

شرکت پذیری این عملگر از راست به چپ می باشد، بدین معنا که چنانچه چندین عملگر نسبت دهی داشته باشیم، این عملگرها از راست به چپ محاسبه می شوند. مثلا پس از اجرای دستور زیر :

```
a = b = c = 10;
```

عملگر انتسابی

توضیحات	مثال	اولویت	نام	عملگر
ابتدا a را افزایش داده و سپس از آن در عبارت استفاده می کند.	++a	راست به چپ	پیش افزایش	++
ابتدا از مقدار فعلی a در عبارت موردنظر استفاده کن و سپس آن را افزایش بده	a++	چپ به راست	پس افزایش	++
ابتدا a را کاهش داده و سپس از آن در عبارت استفاده می کند.	--a	راست به چپ	پیش کاهش	--
ابتدا از مقدار فعلی a در عبارت موردنظر استفاده کن و سپس آن را کاهش بده	a--	چپ به راست	پس کاهش	--

عملگر انتسابی

```
#include<iostream>
using namespace std;

void main()
{
        int a=5;
        int b,c,d,e,g;
        b=a++;
        cout<<"a is: "<<a<<"\n";
        cout<<"b is: "<<b<<"\n\n";
        c=++a;
        cout<<"a is: "<<a<<"\n";
        cout<<"c is: "<<c<<"\n\n";
        d=a--;
        cout<<"a is: "<<a<<"\n";
        cout<<"d is: "<<d<<"\n\n";
        e=--a;
        cout<<"a is: "<<a<<"\n";
        cout<<"e is: "<<e<<"\n";
}
```

```
a is: 6
b is: 5

a is: 7
c is: 7


a is: 6
d is: 7


a is: 5
e is: 5
```

عملگر انتسابی

int a=12;

عملگر	مثال	توسعه یافته عملگر	مقدار متغییر پس از اعمال عملگر
		49	
+=	a+=4	a = a + 4	a=16
-=	a-=5	a = a - 5	a=7
*=	a*=3	a = a * 3	a=36
/=	a/=4	a = a / 4	a=3
%=	a%5	a = a %5	a=2

عملگر مقایسه ای

عملگر	مفهوم عملگر	مثال
>	بزرگتر (>)	a > b
<	کوچکتر (<)	a < b
>=	بزرگتر یا مساوی (≥)	a >= b
<=	کوچکتر یا مساوی (≤)	a <= b
==	مساوی (=)	a == b
!=	نامساوی (≠)	a != b

عملگر مقایسه ای

نکته مهمی که باید به آن دقت کرد عملگر مساوی (= =) است، چرا که یک اشتباه بسیار متداول برنامه نویسان C استفاده اشتباه از عملگر انتساب (=) بجای عملگر تساوی (= =) است که باعث ایجاد خطا در برنامه می شود.

عامل دیگری که باید در بکارگیری عملگرهای مقایسه ای دقت کرد این است که عملگر =< و => را نباید بصورت <= و >= بکار برد زیرا یک خطای نحوی است.

اولویت این عملگرها از بالا به پایین بشرح زیر است:

۱- عملگرهای <، > ، =< و =>

۲- عملگرهای == و =!

عملگر منطقی

عملگر	مفهوم عملگر	نحوه کار
&&	منطقی AND	اگر هر دو عملوند درست باشند, درست و در غیر اینصورت نادرست باز می گرداند.
||	منطقی OR	اگر هر دو عملوند نادرست باشند, نادرست و در غیر اینصورت درست باز می گرداند.
!	منطقی NOT	اگر عملوند درست باشد، نادرست و اگر نادرست باشد، درست برمی گرداند.

۱- عملگر ! ۲- عملگر && ۳- عملگر ||

عملگر شرطی

گاهی لازم است که ابتدا یک شرط بررسی شده و سپس برمبنای نتیجه (درست یا نادرست بودن آن) یکی از دو عبارت ممکن بازگردانده شود. گرچه معمولا برای اینکار از دستور if (که در فصل بعدی بحث خواهد شد) استفاده می شود، اما یک عملگر ۳تایی (با ۳ عملوند) نیز برای آن وجود دارد. شکل کلی این عملگر بصورت زیر است:

(شرط) ? عبارت۱ : عبارت۲

```
a = (k<10) ? 100 : 50;
```

که این عبارت معادل دستور زیر است:

```
if (k<10) a=100;
else a=50;
```

دریافت داده از کاربر و نمایش اطلاعات

```
// Comparing integers using if statements, relational operators
// and equality operators.
#include <iostream> // allows program to perform input and output

using namespace std;

// function main begins program execution
int main()
{
  int number1; // first integer to compare
  int number2; // second integer to compare

  cout << "Enter two integers to compare: "; // prompt user for data
  cin >> number1 >> number2; // read two integers from user

  if ( number1 == number2 )
    cout << number1 << " == " << number2 << "\n";

  if ( number1 != number2 )
    cout << number1 << " != " << number2 << "\n";
```

دریافت داده از کاربر و نمایش اطلاعات

```
   if ( number1 < number2 )
      cout << number1 << " < " << number2 << "\n";

   if ( number1 > number2 )
      cout << number1 << " > " << number2 << "\n";

   if ( number1 <= number2 )
      cout << number1 << " <= " << number2 << "\n";

   if ( number1 >= number2 )
      cout << number1 << " >= " << number2 << "\n";
} // end function main
```

```
Enter two integers to compare: 3 7
3 != 7
3 < 7
3 <= 7
```

```
Enter two integers to compare: 22 12
22 != 12
22 > 12
22 >= 12
```

```
Enter two integers to compare: 7 7
7 == 7
7 <= 7
7 >= 7
```

عملگر کاما

این عملگر برای به زنجیر در آوردن چندین عملیات مختلف استفاده میشود.بدین شکل که ارزش لیستی از عبارتها که بوسیله کاما از هم جدا میشوند برای آن عبارتی منظور میشود که در سمت راست بقیه قرار دارد.دستور زیر نحوه استفاده از این عملگر را نشان میدهد:

```
value=(count,99,3,100);
```

عملگر sizeof

این عملگر طول یک متغییر یا یک نوع داده را مشخص میکند:

sizeof متغییر ;

sizeof(نوع داده) ;

عملگر بیتی انتقال

عملگرهای بیتی انتقال، عبارتند از عملگر بیتی انتقال به چپ و عملگر انتقال بیتی به راست که آنها را به شکل زیر نشان میدهند:

کاربرد	نماد در c++
عملگر بیتی انتقال به چپ	<<
عملگر انتقال بیتی به راست	>>

عملگرهای بیتی انتقال به چپ >> باعث انتقال بیتهای عملوند متغییر چپ به اندازه تعداد بیت های تعیین شده بوسیله مقدار طرف راست عملگر به سمت چپ مشود.در اینصورت بیتهای سمت چپ از دست رفته وبیت های خالی شده سمت راست با صفر پر میگردند.

عملگر بیتی انتقال

X=0x7ca4;

X=0111,1100,1010,0100

Y=x<<4;

Y=1100,1010,0100,0000

در نتیجه خواهیم داشت:

Y=0xca40;

عملگر بیتی انتقال

عملگرهای بیتی انتقال به راست << باعث انتقال بیتهای عملوند متغییر راست به اندازه تعداد بیت های تعیین شده بوسیله مقدار طرف راست عملگر به سمت راست مشود.در اینصورت بیتهای سمت راست از دست رفته وبیت های خالی شده سمت چپ با صفر پر میگردند.

علاوه بر چند عملگر بالا میتوان از عملگرهای نیز برای عملیات بیتی استفاده نمود:

عمل	نماد در C++
نقیض بیتی	!
یای بیتی	|
و بیتی	&
یای انحصاری بیتی	^

عملگر بیت انتقال

مثال: اگر a=10 و b=15 باشد مقدار c=a&b به صورت زیر محاسبه میشود:

	بایت پر ارزش								بایت کم ارزش							
	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
a	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0
b	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
c	0	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0

در اینصورت خروجی برابر با c=10 خواهد بود.

عملگر توصیف کننده نوع داده

با استفاده از یک توصیف کننده نوع داده میتوان انواع داده را ایجاد نمود که با نیازهای خاص به طور دقیق انطباق داشته باشد.

توصیف کننده های نوع داده عبارتند از:

signed

unsigned

long

short

توصیف کننده long بزرگی داده ها را تقریبا دوبرابر میکند وتوصیف کننده short طول داده را تقریبا نصف میکنند.

توصیف کننده unsigned را میتوان روی char و int اعمال نمود.این توصیف کننده را میتوان همراه long وshort نیز مورد استفاده قرار داد. این توصیف کننده برای ایجاد یک عدد صحیح بدون علامت استفاده میشود.اعداد صحیح بدون علامت از تمامی بیت های جهت نگهداری مقادیر استفاده میکنند و همیشه عددی مثبت خواهندبود.

عملگر توصیف کننده نوع داده

```
#include<iostream.h>
#include<iomanip.h>
#include<conio.h>
main(){
   cout << "sizeof(char) is " << sizeof(char) << " bytes " << endl;
 cout << "sizeof(short) is " << sizeof(short) << " bytes " << endl;
 cout << "sizeof(int) is " << sizeof(int) << " bytes " << endl;
 cout << "sizeof(long) is " << sizeof(long) << " bytes " << endl;
 cout << "sizeof(long long) is " << sizeof(long long) << " bytes " << endl;
 cout << "sizeof(float) is " << sizeof(float) << " bytes " << endl;
 cout << "sizeof(double) is " << sizeof(double) << " bytes " << endl;
 cout << "sizeof(long double) is " << sizeof(long double) << " bytes " << endl;
 cout << "sizeof(bool) is " << sizeof(bool) << " bytes " << endl;
}
```

```
sizeof(char) is 1 bytes
sizeof(short) is 2 bytes
sizeof(int) is 4 bytes
sizeof(long) is 4 bytes
sizeof(long long) is 8 bytes
sizeof(float) is 4 bytes
sizeof(double) is 8 bytes
sizeof(long double) is 12 bytes
sizeof(bool) is 1 bytes
```

عملگر توصیف کننده نوع داده

نوع داده	اندازه(بایت)	مینیمم	ماکزیمم
int	4 (2)	-147483648	2147483647
unsigned int	4 (2)	0	4294967295
char	1	-128	127
unsigned char	1	0	255
short (or short int)	2	-32768	32767
unsigned short	2	0	65535
long (or long int)	4 (8)	-2147483648	2147483647
unsigned long	4 (8)	0	مانند بالا
long long (or long long int)	8	-2^{63}	$2^{63}-1$
unsigned long long	8	0	$2^{64}-1$
float	4	3.4e38	3.4e-38
double	8	1.7e308	1.7e-308
long double	12		
bool	1	false (0)	true (1 or non-zero)
wchar_t	2 (4)		

قالب بندی نوع داده

نوع داده را میتوان به طور موقت تغییر داد .شکل کلی این عمل به صورت زیر خواهد بود.

متغییر (نوع داده) ;

مثال:

```
float f;
f=100.2;
cout<<(int) f;//100چاپ
int i=10;
cout<<i/3<<'\n';
cout<<(float) i/3<<'\n';
```

قالب بندی خروجی

برای چاپ کمیت ها به شکل مورد نظر بااستفاده از cout میتوان آنها راقالب بندی نمود.برای این کار از کمیتهای به شکل زیر که در فایل سرآیندiomanip قرار دارند، میتوان استفاده نمود.

دستور	عملکرد
setw(d)	به اندازه d فضای خالی چاپ میشود
Setprecision(d)	تا دقت dرقم عددراچاپ میکند

قالب بندی خروجی

```
#include <iomanio.h>
main() {
   // Floating point numbers
   double pi = 3.14159265;
   cout << fixed << setprecision(4); //fixed format with 2 decimal
places
   cout << pi << endl;
   cout << "|" << setw(8) << pi << "|" << setw(10) << pi << "|" <<
endl;
      // setw() is not sticky, only apply to the next operation.
   cout << setfill('-');
   cout << "|" << setw(8) << pi << "|" << setw(10) << pi << "|" <<
endl;
   cout << scientific;  // in scientific format with exponent
   cout << pi << endl;

   // booleans
   bool done = false;
   cout << done << endl;  // print 0 (for false) or 1 (for true)
   cout << boolalpha;     // print true or false
   cout << done << endl;


}
```

قالب بندی خروجی

در این صورت خروجی به شکل زیر خواهد بود:

```
3.1416
|  3.1416|   3.1416|
|--3.1416|----3.1416|
3.1416e+000
0
false
```

تبدیل نوع داده

یک بخش از قواعد تبدیل در C++ را ترفیع نوع (type casting) مینامند.در C++ هر جا یک char یا short در عبارتی به کاررفته باشد،مقدار آن در طی ارزیابی آن عبارت به طور اتوماتیک به int ارتقا داده میشود.

باید توجه نمود که ترفیع نوع فقط در طی ارزیابی آن عبارت موثر میباشد و آن متغییر از نظر فیزیکی بزرگتر نمیشود.در اصل کامپایلر از کپی موقتی از مقدار آن متغییر استفاد خواهد نمود. پس از آنکه ترفیع نوع های اتوماتیک اعمال گردید کامپایلر همه عملوندها را به نوع بزرگترین عملوند تبدیل میکند.

اگر در یک دستور انتساب که نوع داده سمت راست آن با نوع داده سمت چپ آن تفاوت داشته باشد ،نوع داده سمت راست به نوع داده سمت چپ تبدیل میگردد.وقتی نوع داده سمت چپ بزرگتر از نوع داده سمت راست باشداین فرایندمشکلی ایجاد نمی کند.اما وقتی نوع داده سمت چپ از نوع داده سمت راست کوچکتر باشد ممکن است مقداری از داده ها گم شود .

تبدیل نوع داده

مثال:در عمل تبدیل نوع داده از یک مقدار اعشاری به یک مقدار صحیح ،جز اعشاری عدد ازبین میرود:

```
#include <iostream>
#include <iomanip>


main() {
   int i;
   double d;



   i = 3;
   d = i;     // انتساب یک متغییر صحیح به یک متغییر اعشتری
   cout << "d = " << d << endl;   // 3.0خروجی

   d = 5.5;
   i = d;     // انتساب یک متغییر اعشاری به یک متغییر صحیح
   cout << "i = " << i << endl;   // 5 خروجی

   i = 6.6;   // انتساب یک مقدار اعشاری به یک متغییر صحیح
   cout << "i = " << i << endl;   // 6خروجی

   }
```

تبدیل نوع داده

مثال: خروجي قطعه كد زير برابر ۲۴- ميباشد:

```
#include <iostream.h>

main(){

char ch;

int i; i=1000;

ch=i;

i=ch;

cout<<i;}
```

رفتار با ثابت ها

به طور پیش فرض ،کامپایلر یک عدد ثابت عددی را در کوچکترین نوع داده ای که میتواند آن داده را جای دهد ،نگه میدارد.بنابراین اگر فرض کنیم که اعدا صحیح ۱۶ بیتی هستند ،به طور پیش فرض ۱۰ یک int است اما ۴۶۰۰۰ یک Unsigned و 100001 یک long است.گرچه مقدار ۱۰ را در یک char هم میشود جای داد اما کامپایلر چنین کاری نمی کند چون این کار به معنای شکستن حدود نوع داده است.

تنها استثنا در قاعده کوچکترین نوع داده زمانی است که ثابت های مورد بحث اعشاری هستند، که در این صورت از نوع double خواهند بود.

رفتار با ثابت ها

مثال: با توجه به توضیحات بالا خروجی قطعه کد زیر

```
cout<<"enter a number";
cin>>i;
if(i&32768) cout<<"N";
else cout<< "P" ;
```

اگر عدد i مثبت باشد P و اگر منفی باشدN خواهد بود.

در مواردی که فرض C++ در مورد یک ثابت عددی ،همان چیزی نیست که شما میخواهید،میتوانید با بکار بردن یک پسوند نوع دقیق آن ثابت عددی را مشخص کنید.

برای نوع اعشاری ،اگر پس از عدد مورد نظر خود یک 'F' قرار دهید با آن عدد به مثابه یک float رفتار میشود.

اگر پس از آن عددیک L قرار دهید،آن عدد یک long doubleتلقی میشود. برای انواع صحیح پسوند 'U' به معنای unsigned و'L' به معنای long است.

تمرین ها

۱-با اجرای هریک از دستورهای زیر چه چیزی در خروجی چاپ می شود. فرض کنید x=2 و y=3 باشد و در صورتی که خروجی ندارد بنویسید "هیچ".

A. cout<<x;
B. cout<<x+x;
C. cout<<"x=";
D. cout<<"x= "<<x;
E. z=x+y;
F. // cout<<x;

۲-کد زیر چه چیزی را در خروجی چاپ می کند.

```
cout<<"*\n**\n***\n****\n*****\n";
```

تمرین ها

۳-در دستورهای C++ زیر ترتیب ارزیابی عملگرها را مشخص کنید و مقدار x را پس از اجرای هر دستور نشان دهید:

A. x=7+3*6/2-1;
B. x=7%8+3*5-7/8;
C. x=(3*9*(3+(9*3/(3))));

۴-خروجی را بیابید:

```
int a,b=5;
a=(b++)*(++b);
cout<<a<<b;
```

۵-خروجی را بیابید:

```
int a=6;
int b=2;
c=++a*( b++)+a-a++;
cout<<a<<b<<c;
```

نمایش و چاپ اطلاعات

با استفاده از **عملگر** <<، می‌توان اطلاعات قابل چاپ روی صفحه نمایش را به **شیء cout** فرستاد. این شیء در فایل کتابخانه‌ای iostream.h قرار دارد. فرم عمومی کاربرد آن عبارتنداز:

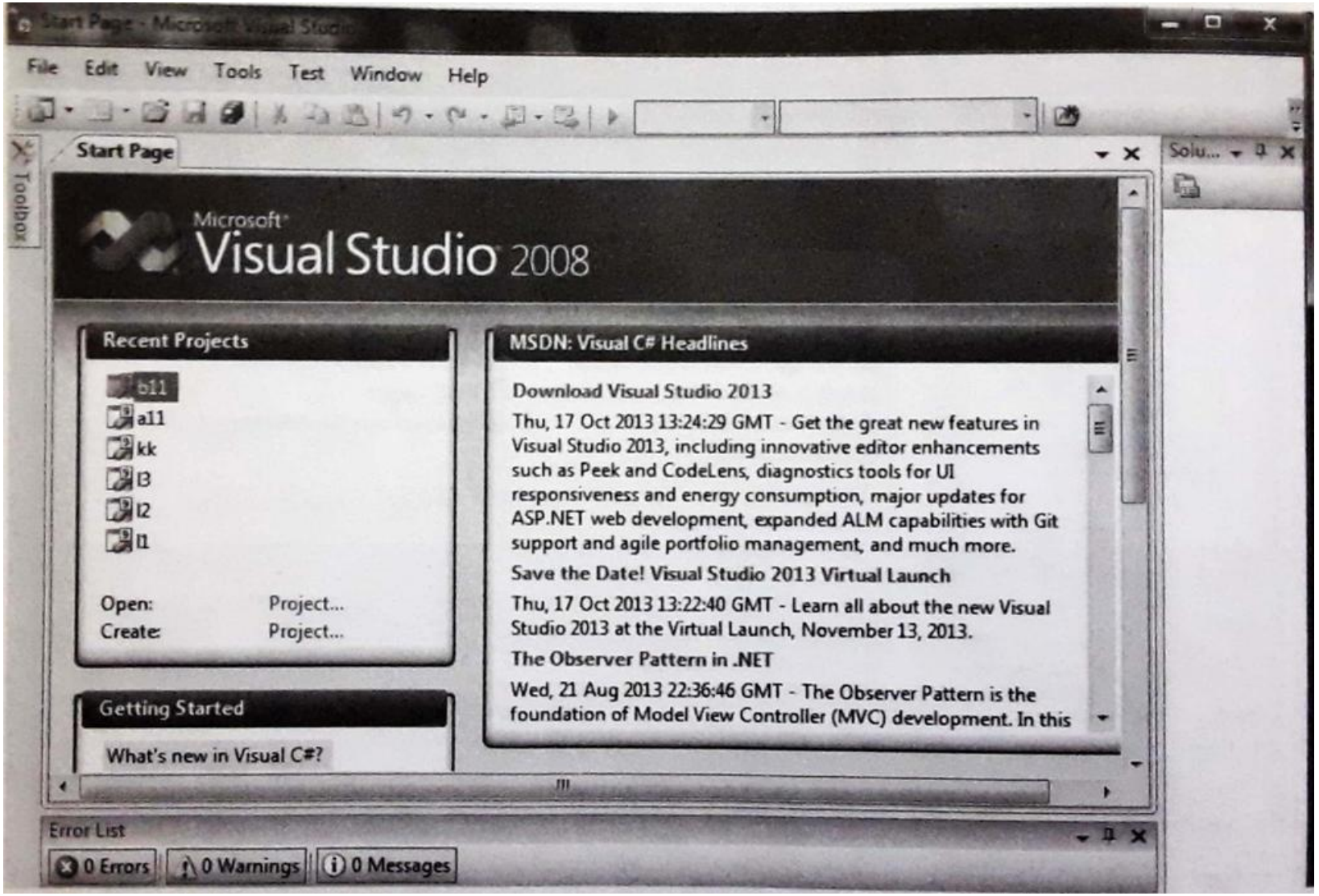
cout << عبارت ۱ << عبارت ۲ << ... ;

عبارت ۱، عبارت ۲ و ... داده‌هایی هستند که باید نمایش داده شوند. اگر عبارتی شامل **متن ثابت** باشد باید آن را داخل دابل کوتیشن یا گیومه دوتایی("...") قرار داد. برای نمایش **مقدار یک متغیر** نیاز به استفاده از دابل کوتیشن نیست. برای انتقال مکان به سطر بعدی می‌توان کارکتر کنترلی "\n" یا دستور endl را بکار برد. جدول زیر تعدادی از **کاراکترهای کنترلی** را نمایش می‌دهد.

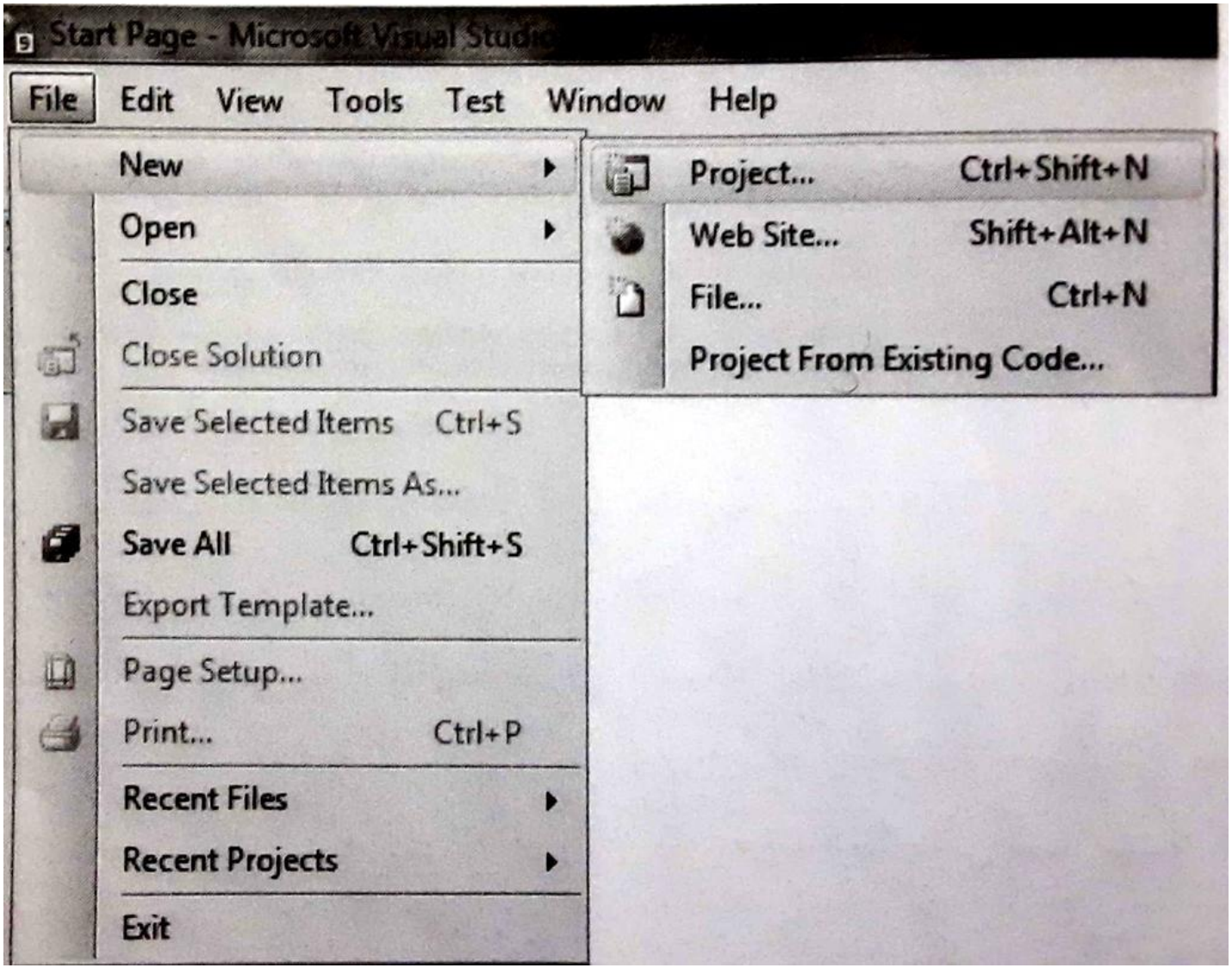
نمایش و چاپ اطلاعات

کارکتر کنترلی	توضیح
\n	مکان‌نما به ابتدای سطر بعدی انتقال می‌دهد.
\t	مکان‌نما به ابتدای کلمه بعدی به اندازه‌ی ۸ کارکتر منتقل می‌شود.
\a یا \v	بوق سیستم را فعال می‌کند.
\\	باعث نمایش کارکتر \ می‌شود
\"	کارکتر " را نمایش می‌دهد
\b	کارکتر قبل از خودش را حذف می‌کند
\r	نقش کلید Enter را ایفا می‌نماید
\?	علامت ? را نمایش می‌دهد
\:	علامت : را نمایش می‌دهد

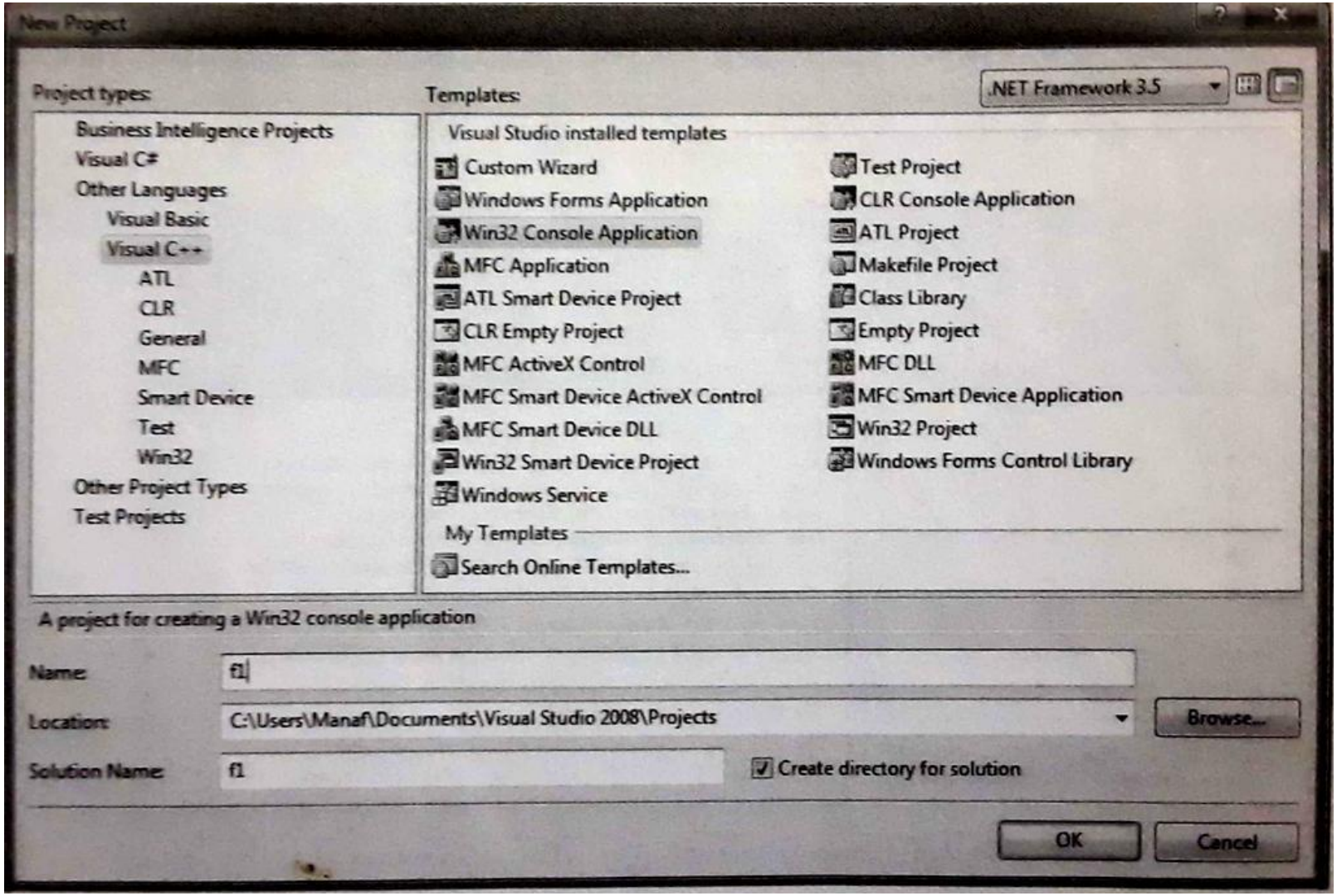
آشنایی با محیط Visual Studio



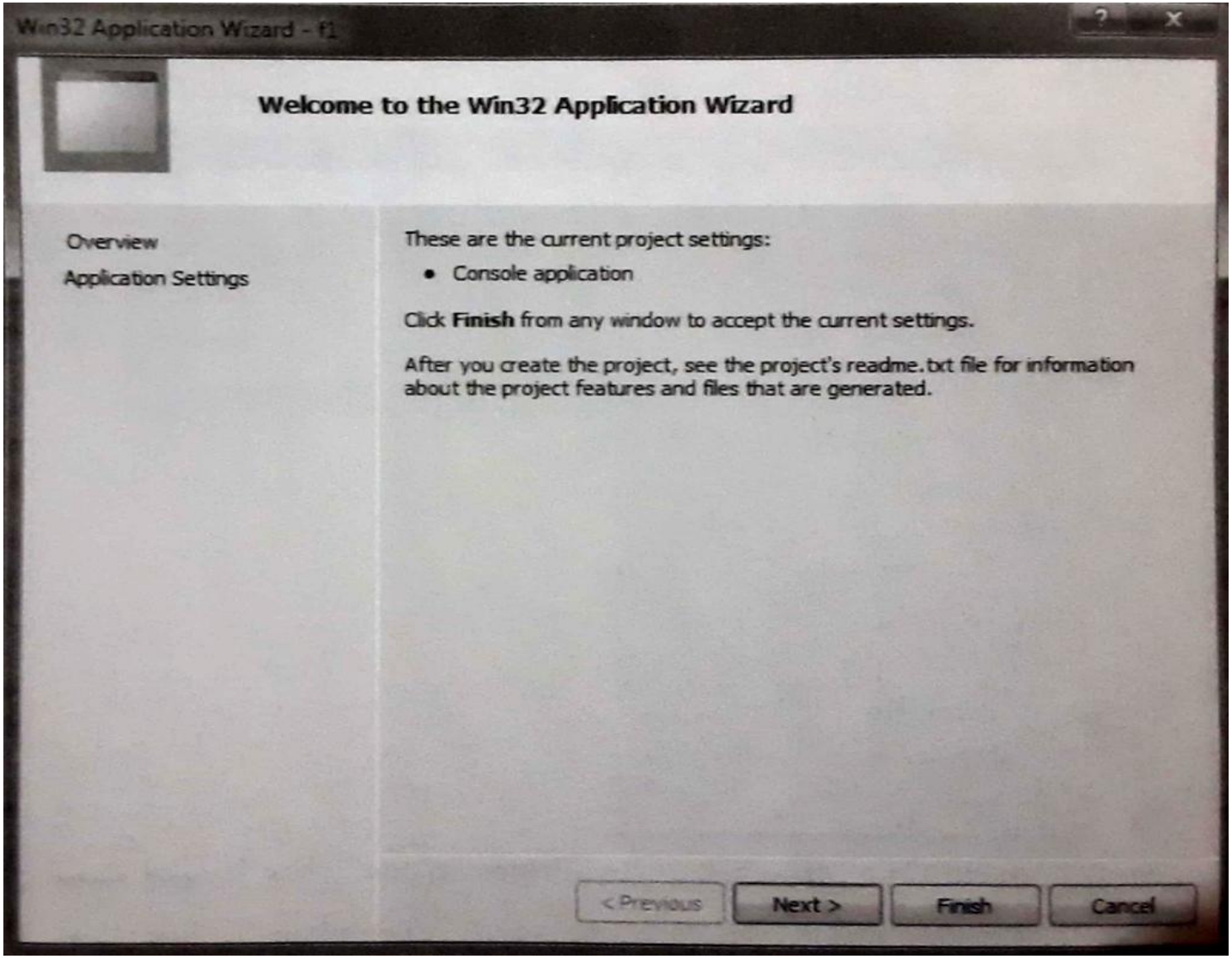
آشنایی با محیط Visual Studio



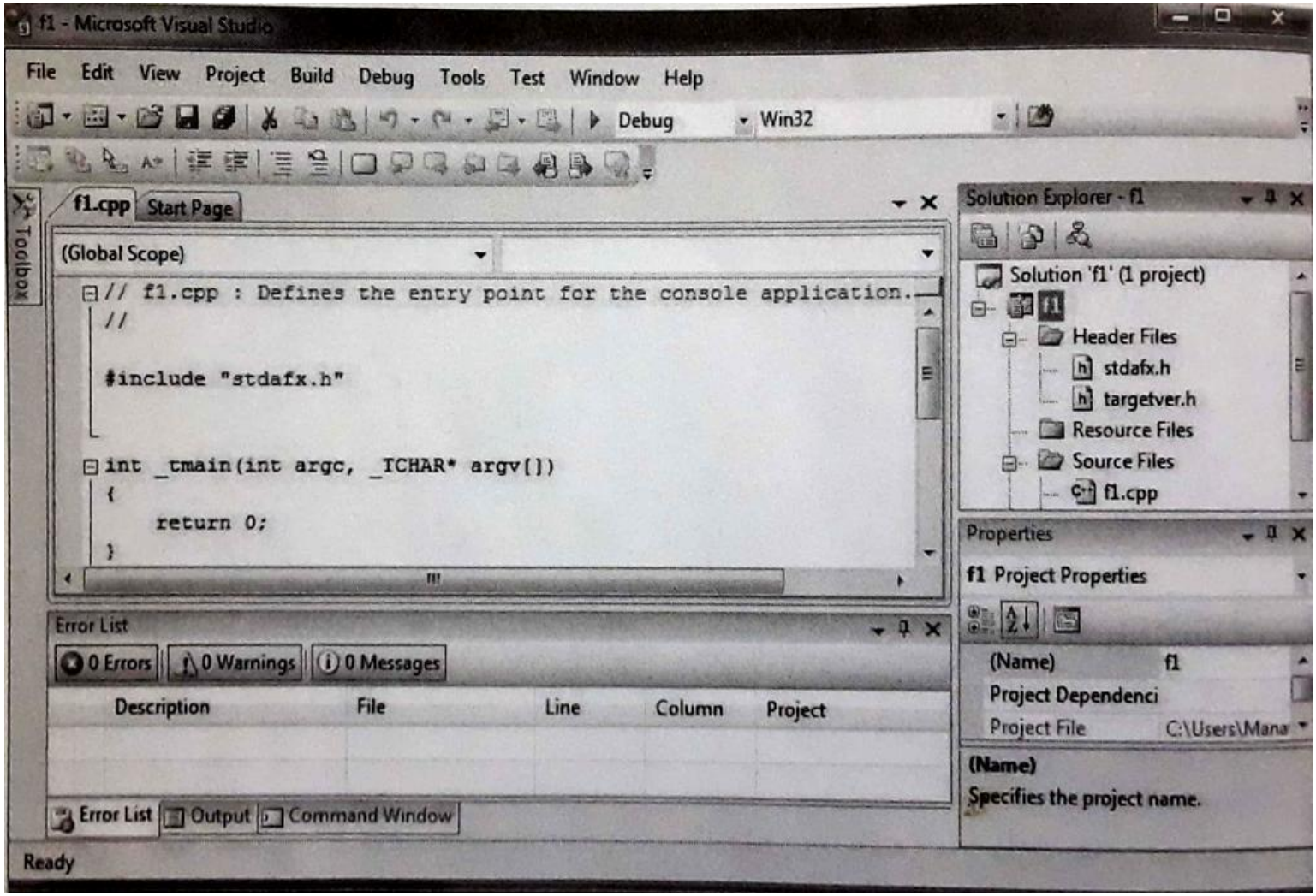
آشنایی با محیط Visual Studio



آشنایی با محیط Visual Studio



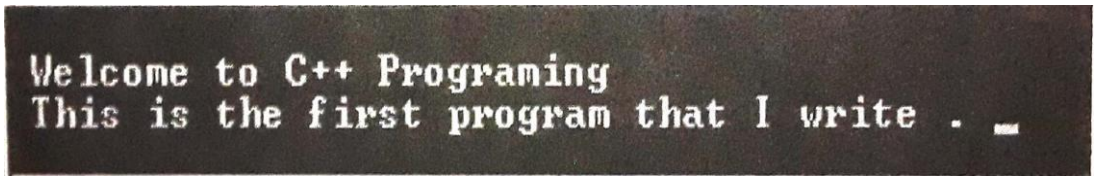
آشنایی با محیط Visual Studio



نمایش دو جمله زیر هم

برنامه‌ای به زبان Visual C++ بنویسید که دو جمله خوش‌آمدگویی و نوشتن اولین برنامه را در دو سطر زیر هم نمایش دهد.

```
#include"stdafx.h"
#include<iostream>
#include<conio.h>
using namespace std;

int main (   )
{
  cout << endl << " Welcome to C++ Programing ";
  cout << endl << " This is the first program that I write . ";
  getch (  );
  return  0 ;
}
```



مقدار دهی اولیه سه متغیر و نمایش آنها

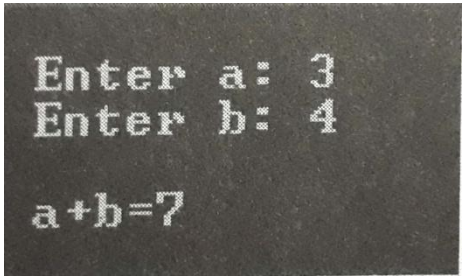
برنامه‌ای به زبان Visual C++ بنویسید که سه متغیر a، b و ch را که به‌ترتیب از نوع صحیح، اعشاری و کارکتری می‌باشند را با مقدار اولیه ۱۲۳۴ و ۷۶۵۴/۳۲۱ و 'B' اعلان کرده و سپس مقادیر آنها را نمایش می‌دهد.

```
#include"stdafx.h"
#include<iostream>
#include<conio.h>
using namespace std;

int main()
{
  char  ch='B';
  int   a=1234; // This statement declares variable as a integer and initialize it with
                   1234
  float b=7654.321;
  cout <<endl<< " ch= "<<ch;
  cout <<endl<< " a = " <<a;
  cout <<endl<< " b = "<< b;
  getch ();
  return 0;
}
```

ورود مقادیر دو متغیر و محاسبه مجموع آنها

برنامه‌ای به زبان Visual C++ بنویسید که دو متغیر a و b را از صفحه کلید خوانده، سپس مجموع آنها را نمایش دهد.

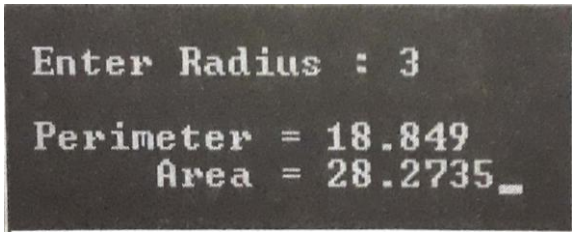
```
#include"stdafx.h"
#include<iostream>
#include<conio.h>
using namespace std;

int main ()
{
  int a , b ;
  cout <<"\n Enter a: " ;
  cin >> a;
  cout << " Enter b: " ;
  cin >> b ;
  cout << "\n a+b=" <<a+b ;
  getch () ;
  return 0 ;
}
```



محیط و مساحت دایره

برنامه‌ای بنویسید که شعاع دایره‌ای را خوانده، محیط و مساحت آن را محاسبه کرده و نمایش دهد.

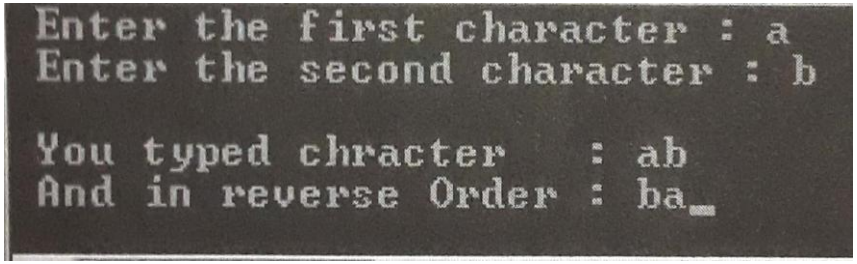
```
#include"stdafx.h"
#include<iostream>
#include<conio.h>
#define  PI 3.1415
using namespace std;

int main()
{
  float radius, perimeter, area;
  cout <<"\n Enter Radius : ";
  cin >> radius;
  perimeter = 2*PI*radius;
  area= PI * radius* radius;
  cout <<"\n Perimeter = "<<perimeter<< "\n     Area = "<<area;
  getch();
  return 0;
}
```



نمایش دو کارکتر ورودی

برنامه‌ای بنویسید که دو کاراکتر را از صفحه کلید خوانده، آن‌ها را به دو روش مختلف کنار هم قرار داده و نمایش دهد.

```
#include "stdafx.h"
#include <iostream>d;
#include <conio.h>
using namespace std;
int main(){
  char ch1,ch2;
  cout << " Enter the first character : ";
  cin>>ch1;
  cout << " Enter the second character : ";
  cin>>ch2;
  cout << "\n You typed chracter   : " << ch1<<ch2 ;
  cout << "\n And in reverse Order : " << ch2<<ch1 ;
  getch();
  return 0;
}
```



میانگین سه عدد

برنامه‌ای به زبان Visual C++ بنویسید که مقادیر سه متغیر a ، b و c را بعنوان ورودی دریافت کند. سپس **میانگین** (امید ریاضی) آنها را محاسبه کرده و نمایش دهد.

۱. شروع

۲. **مقادیر متغیرهای *a* ، *b* و *c* را از صفحه کلید بخوان** (واردکن).

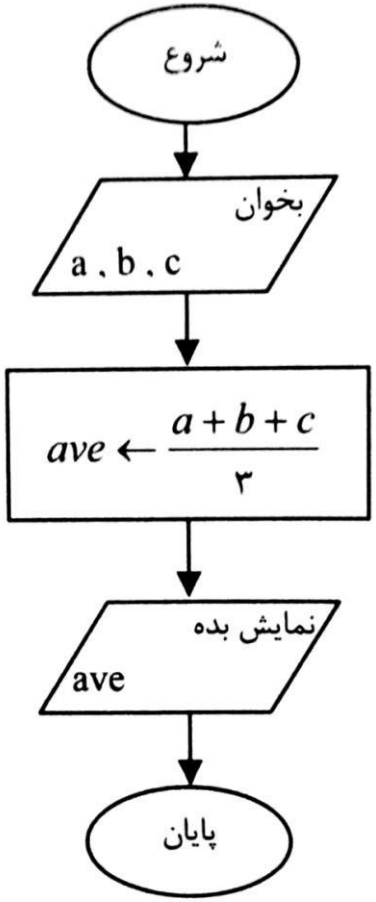
۳. **مقدار میانگین را محاسبه** کرده و در **متغیر *ave* جایگزین** کن $ave \leftarrow \frac{a+b+c}{3}$

۴. **مقدار *ave* را نمایش بده.**

۵. پایان.

میانگین سه عدد

برنامه‌ای به زبان Visual C++ بنویسید که مقادیر سه متغیر a ، b و c را بعنوان ورودی دریافت کند. سپس **میانگین** (امید ریاضی) آنها را محاسبه کرده و نمایش دهد.



میانگین سه عدد

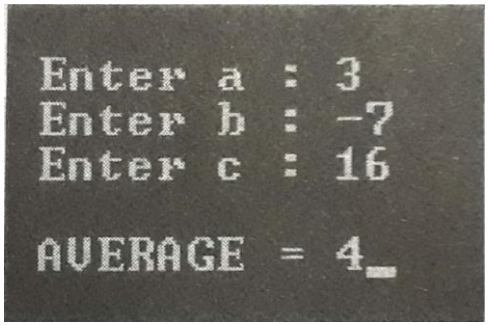
برنامه‌ای به زبان Visual C++ بنویسید که مقادیر سه متغیر a ، b و c را بعنوان ورودی دریافت کند. سپس **میانگین** (امید ریاضی) آنها را محاسبه کرده و نمایش دهد.

	a	*b*	*c*	*ave*
نمونه اول	~~۳~~	~~۷-~~	~~۱۶~~	$\frac{۳-۷+۱۶}{۳} =$ ۴
نمونه دوم	۰	۱۲	-۵	$\frac{۰+۱۲-۵}{۳} =$ ۲/۳۳

میانگین سه عدد

برنامه‌ای به زبان Visual C++ بنویسید که مقادیر سه متغیر a ، b و c را بعنوان ورودی دریافت کند. سپس **میانگین** (امید ریاضی) آنها را محاسبه کرده و نمایش دهد.

```
#include "stdafx.h"
#include <iostream>
#include <conio.h>
using namespace std;
int main()
{
  float a,b,c,ave;
  cout<<"\n Enter a : " ;
  cin>>a ;
  cout<<" Enter b : " ;
  cin>>b ;
  cout<<" Enter c : " ;
  cin>>c ;
  ave= (a+b+c) /3 ;
  cout<<"\n AVERAGE = "<<ave ;
  getch () ;
  return 0 ;
}
```



تعویض مقادیر دو متغیر

برنامه‌ای به زبان Visual C++ بنویسید که مقادیر دو متغیر را از ورودی خوانده، **سپس مقادیر آنها را با یکدیگر تعویض** نماید و نتیجه را چاپ کند.

۱. شروع.

۲. مقادیر **متغیرهای *a* و *b*** را از صفحه کلید بخوان.

۳. مقدار *a* را در یک **متغیرموقت بنام *temp*** قرار بده.

$temp \leftarrow a$

۴. مقدار *b* را در متغیر *a* کپی کن

$a \leftarrow b$

۵. مقدار *temp* را در متغیر *b* قرار بده

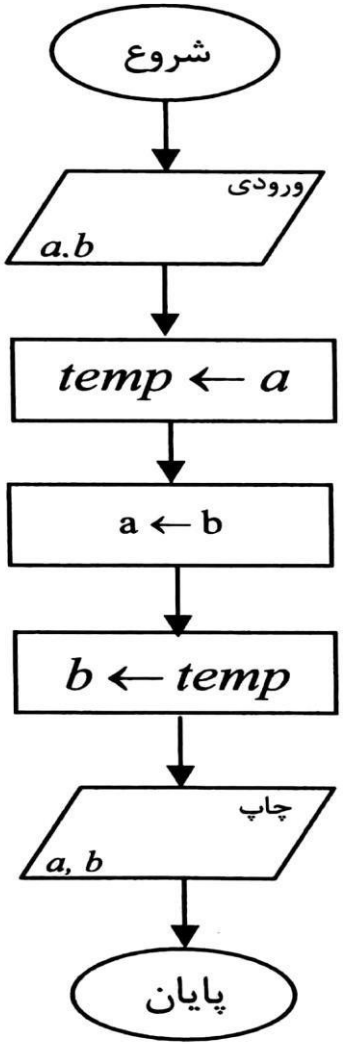
$b \leftarrow temp$

۶. مقادیر *a* و *b* را چاپ کن.

۷. پایان

تعویض مقادیر دو متغیر

برنامه‌ای به زبان Visual C++ بنویسید که مقادیر دو متغیر را از ورودی خوانده، **سپس مقادیر آنها را با یکدیگر تعویض** نماید و نتیجه را چاپ کند.

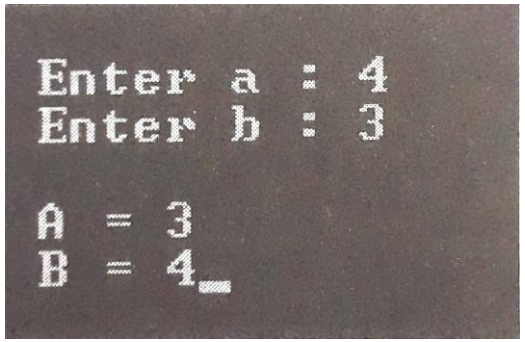


a	b	temp
۴	۳	۴
③	④	

تعویض مقادیر دو متغیر

برنامه‌ای به زبان Visual C++ بنویسید که مقادیر دو متغیر را از ورودی خوانده، **سپس مقادیر آنها را با یکدیگر تعویض** نماید و نتیجه را چاپ کند.

```
#include "stdafx.h"
#include <iostream>
#include <conio.h>
using namespace std;

int main()
{
  int a,b,temp;
  cout<<"\n Enter a : ";
  cin>>a;
  cout<<" Enter b : ";
  cin>>b;
  temp=a;
  a=b;
  b=temp;
  cout<<"\n A = "<<a<<"\n B = "<<b;
  getch();
  return 0;
}
```



تبدیل زمان

برنامه‌ای به زبان Visual C++بنویسیدکه زمان t (یک عدد اعشاری با دو رقم اعشار) را برحسب ساعت از صفحه کلید خوانده، آن‌را به **ساعت، دقیقه و ثانیه** تبدیل کرده و نتیجه را نمایش دهد.

۱. شروع

۲. **زمان t** را از صفحه کلید بخوان

۳. **جزء صحیح متغیر t را در متغیر h** قرار بده تا مقدار ساعت بدست آید. $h \leftarrow \lfloor t \rfloor$

(بجای این دستور می توان نوشت $h \leftarrow int\ (t)$)

۴. مقدار x را از تفاضل مقدار h از t محاسبه کن. $x \leftarrow t - h$

۵. جزء ساعت را به دقیقه تبدیل می‌کنیم. $x \leftarrow x * ۶۰$

۶. **جزء صحیح x را در m قرار بده** تا تعداد دقایق بدست آید $m \leftarrow [x]$

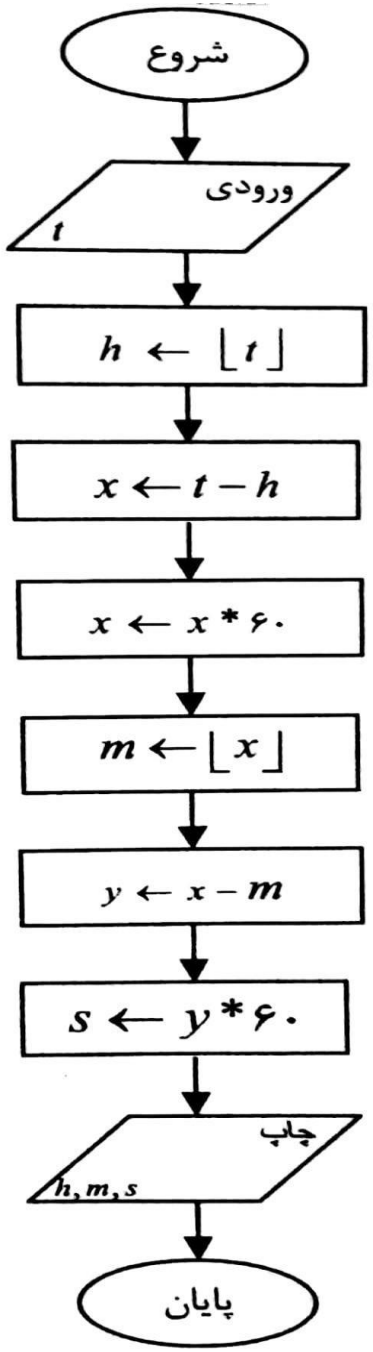
۷. مقدار y را از تفاضل مقدار m از x محاسبه کن. $y \leftarrow x - m$

۸. جهت تبدیل جزء دقیقه y به ثانیه آنرا در ۶۰ ضرب می‌کنیم. $s \leftarrow y * ۶۰$

۹. h و m و s را چاپ کن.

۱۰. پایان.

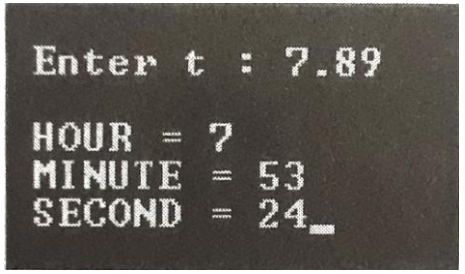
تبدیل زمان



	t	h	x	m	y	s
نمونه اول	۷/۸۹	۷	۰/۸۹ ۵۳/۴۰	۵۳	۰/۴۰ ۲۴	۲۴
نمونه دوم	۳/۱۶	۳	۰/۱۶ ۹/۶	۹	۰/۶ ۳۶	۳۶

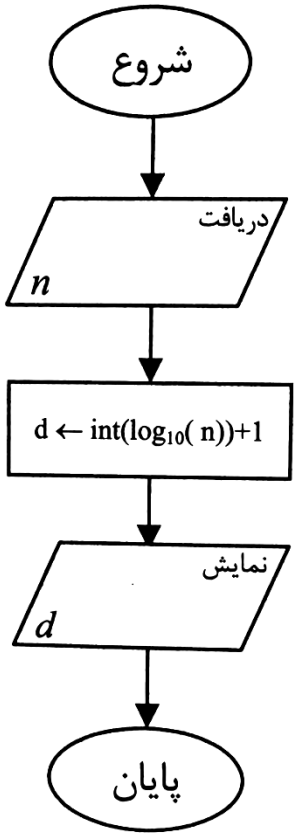
تبدیل زمان

```
#include "stdafx.h"
#include <iostream>
#include <conio.h>
#include <math.h>
using namespace std;

#define N 100
int main()
{
  float t,x,y,h,m,s;
  cout<<"\n Enter t : ";
  cin>>t;
  h=int(t);
  x=t-h;
  x=x*60;
  m=int(x);
  y=x-m;
  s=y*60;
  cout<<"\n HOUR = "<<h<<"\n MINUTE = "<<m<<"\n SECOND = "<<ceil(s);
  getch();
  return 0;
}
```



تعداد ارقام یک عدد طبیعی

برنامه‌ای به زبان Visual C++ بنویسید **که یک عدد طبیعی را خوانده، تعداد ارقام آن** را با استفاده از لگاریتم تعیین کرده و آنرا نمایش دهد.



۱. شروع

۲. **عدد n** را از ورودی بخوان

۳. پس از گـرفتن **لگاریتم عدد n در پایه‌ی ۱۰**، مقدار صـحیح آن را محاسبه کرده و یک واحـد بـه آن اضافـه می‌نماییم. سپس نتیجـه را در متغیر d جایگزین می‌نمائیم.

۴. **d، یا تعداد ارقـام عـدد n** را نمایش بده.

۵. پایان.

تعداد ارقام یک عدد طبیعی

برنامه‌ای به زبان Visual C++بنویسید **که یک عدد طبیعی را خوانده، تعداد ارقام آن** را با استفاده از لگاریتم تعیین کرده و آنرا نمایش دهد.

	n	log (n)	int(log(n))	d
نمونه اول	۷۳	۱/۸۶	۱	②
نمونه دوم	۱۲۴۹	۳/۰۹۷	۳	④

تعداد ارقام یک عدد طبیعی

برنامه‌ای به زبان Visual C++ بنویسید **که یک عدد طبیعی را خوانده، تعداد ارقام آن** را با استفاده از لگاریتم تعیین کرده و آنرا نمایش دهد.

```
#include "stdafx.h"
#include <iostream>
#include <conio.h>
#include <math.h>
using namespace std;

int main()
{
  double n,d;
  cout<<"\n Enter n : ";
  cin>>n;
  d=int(log10(n))+1;
 cout<<"\n THE NUMBER OF DIGITS = "<<d;
  getch();
  return 0;
}
```

```
Enter n : 73
THE NUMBER OF DIGITS = 2
```

نمایش کد اسکی یک کاراکتر

برنامه‌ای با Visual C++ بنویسید که یک کاراکتر را دریافت کرده، کد اسکی معـادل آن‌را نمایش دهد.

```
#include ”stdafx.h”
#include <iostream>
#include <conio.h>

int main()
{
  char ch;
  cout<<” Enter  a character : ”;
  cin>>ch;
  cout << ”\n The ASCII code of ” << ch << ” = ” << (int)ch << ”\n”;
  getch();
  return 0;
}
```

```
Enter  a character : a
The ASCII code of a = 97
```

دستورات شرطی

دستورات شرطی مورد نیاز برای تشخیص تعداد ارقام یک عدد صحیح مثبت ,
تا چهار رقم بنویسید.

```
cin>>n;
if (n > = 0   &&  n < =9)
  cout << "1- digit";
else if (n > = 10  &&  n < =99)
  cout << "2- digits";
else if (n > =100  &&  n <=999)
  cout << "3- digits";
else if (n >=1000  &&  n <=9999)
  cout << "4- digits";
else
  cout << "More than 4 digits";
```

برنامه‌ای بنویسید که یک عدد صحیح مثبت را خوانده، بخش‌پذیری آن‌را بر ۷ یا ۱۳ تعیین کرده و نمایش دهد.

```
#include <iostream.h>
#include <conio.h>

int main()
{
  int n;
  cout<<" Enter n : ";
  cin>>n;
  if(n%7==0)
    cout<<n<<" is divisible by 7 ";
  else if(n%13==0)
    cout<<n<<" is divisible by 13 ";
  else
    cout<<n<<" is not divisible by 7 or 13 ";
  getch();
  return 0;
}
```

```
 Enter n : 123
123 is not divisible by 7 or 13
```

قطعه برنامه‌ای بنویسید که با استفاده از دستور switch، برحسب مقادیر ورودی n از ۱ تا ۷، اسامی یکی از روزهای هفته را نمایش می‌دهد.

```
cin >> n;
switch (n)
{
case 1:   cout << "Saturday";
                  break;
case 2:   cout << "Sunday";
                  break;
case 3:   cout << "Monday";
                  break;
case 4:   cout << "Tuesday";
                  break;
case 5:   cout << "Wednesday";
                  break;
case 6:   cout << "Thursday";
                  break;
case 7:   cout << "Friday";
                  break;
default:  cout << "Error";
}
```

دستور goto

این دستور **بدون ارزیابی یک شرط،** کنترل جریان اجرای دستورات را به نقطه‌ای دیگر از برنامه که دارای **برچسب** است، انتقال می‌دهد. فرم عمومی این دستور عبارتند از:

goto برچسب دستور ;

برچسب دستور، قبل از یک دستور دیگری در برنامه ظاهر می‌شود و آن همان نقطه پرش یا

انتقال کنترل را مشخص می‌کند. هر چند این دستور در زبان‌های برنامه‌نویسی وجود دارد ولی عملاً آن را بکار نمی‌برند، زیرا باعث کاهش خوانایی و فهم در برنامه می‌شود. دستورات دیگری مانند حلقه‌های تکرار وجود دارند که این عمل را، بدون کاهش خوانایی و فهم در برنامه انجام می‌دهند.

دستور goto

قطعه برنامه‌ای بنویسید که مجموع اعداد فرد متوالی با شروع از ۱ را در یک حلقه بی‌نهایت بااستفاده از دستور goto محاسبه کرده و در متغیر sum قرار دهد. سپس جدول ردیابی مربوطه را تکمیل کنید.

```
i = 1;
sum =0 ;
l1:sum + = i;
i + = 2;
goto l1
```

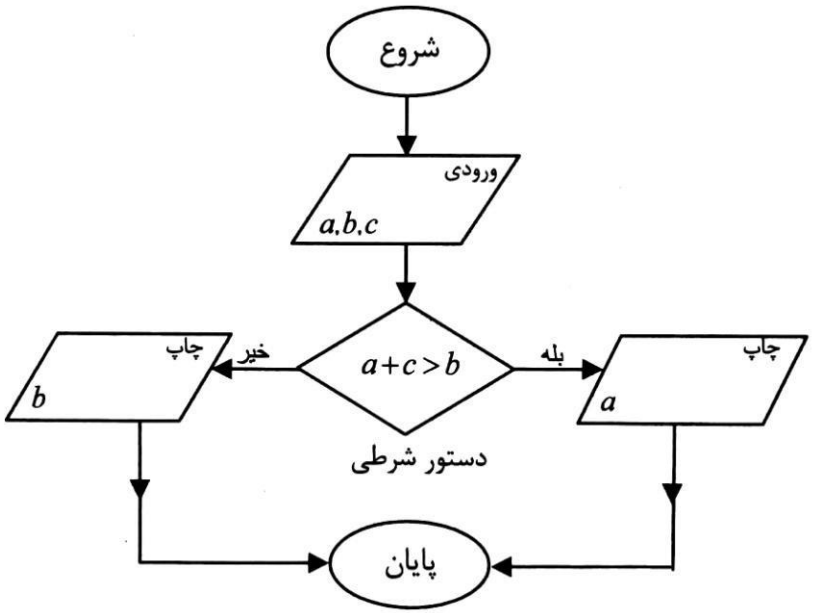
برنامه‌ای بنویسید که سه عدد a ، b و c را بعنوان ورودی دریافت کند. **اگر** $a + c > b$ باشد **آنگاه** مقدار a و **درغیراین‌صورت** مقدار b را چاپ نماید.

۱. شروع

۲. **مقادیر متغیرهای** a **و** b **و** c را بعنوان ورودی دریافت کن.

۳. **اگر** $a + c > b$ **آنگاه مقدار** a را چاپ کن وگرنه مقدار b را چاپ کن.

۴. پایان



برنامه‌ای بنویسید که سه عدد a ، b و c را بعنوان ورودی دریافت کند. **اگر** $a + c > b$ باشد **آنگاه** مقدار a و **درغیراین‌صورت** مقدار b را چاپ نماید.

ارزیابی شرط	شرط	c	b	a	
بله	$۳ < ۷ + ۹$	۹	۳	⑦	نمونه اول
خیر	$۹ < ۳ + ۴$	۴	⑨	۳	نمونه دوم

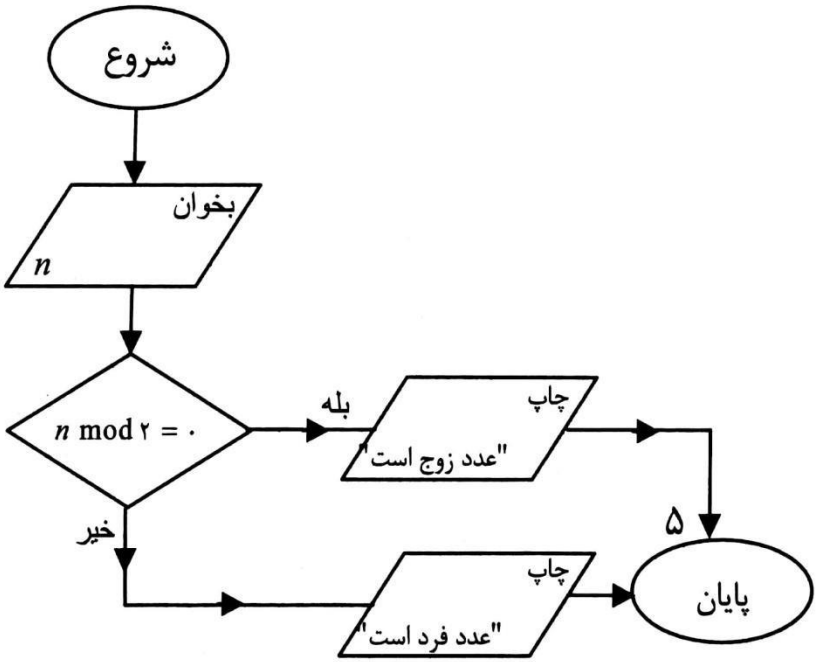
دستورات شرطی

برنامه‌ای بنویسید که سه عدد a ، b و c را بعنوان ورودی دریافت کند. **اگر** $a + c > b$ باشد **آنگاه** مقدار a و **درغیراین‌صورت** مقدار b را چاپ نماید.

```
#include "stdafx.h"
#include <iostream>
#include <conio.h>
using namespace std;

int main()
{
  int a,b,c;
 cout<<"\n Enter a : ";
 cin>>a;
 cout<<" Enter b : ";
 cin>>b;
 cout<<" Enter c : ";
 cin>>c;
 if(a+c>b)
   cout<<"\n A+B>C  ==> "<<a;
 else
   cout<<"\n A+B<=C ==> "<<b;
 getch();
 return 0;
}
```

```
Enter a : 7
Enter b : 3
Enter c : 9

A+B>C  ==> 7
```

برنامه‌ای بنویسید که یک عدد صحیح مثبت را از صفحه کلید دریافت کرده، سپس **زوج یا فرد بودن** آنرا با پیام مناسبی نمایش دهد.



۱. شروع
۲. **عدد *n*** را از صفحه کلید بخوان.
۳. اگر باقیمانده تقسیم صحیح *n* بر ۲ برابر صفر است، پیام "**عدد زوج است**" را چاپ کن و به مرحله ۵ برو وگرنه به مرحله ۴ برو.
۴. پیام "**عدد فرد است**" را چاپ کن و به مرحله ۵ برو
۵. به اجرای الگوریتم خاتمه بده.

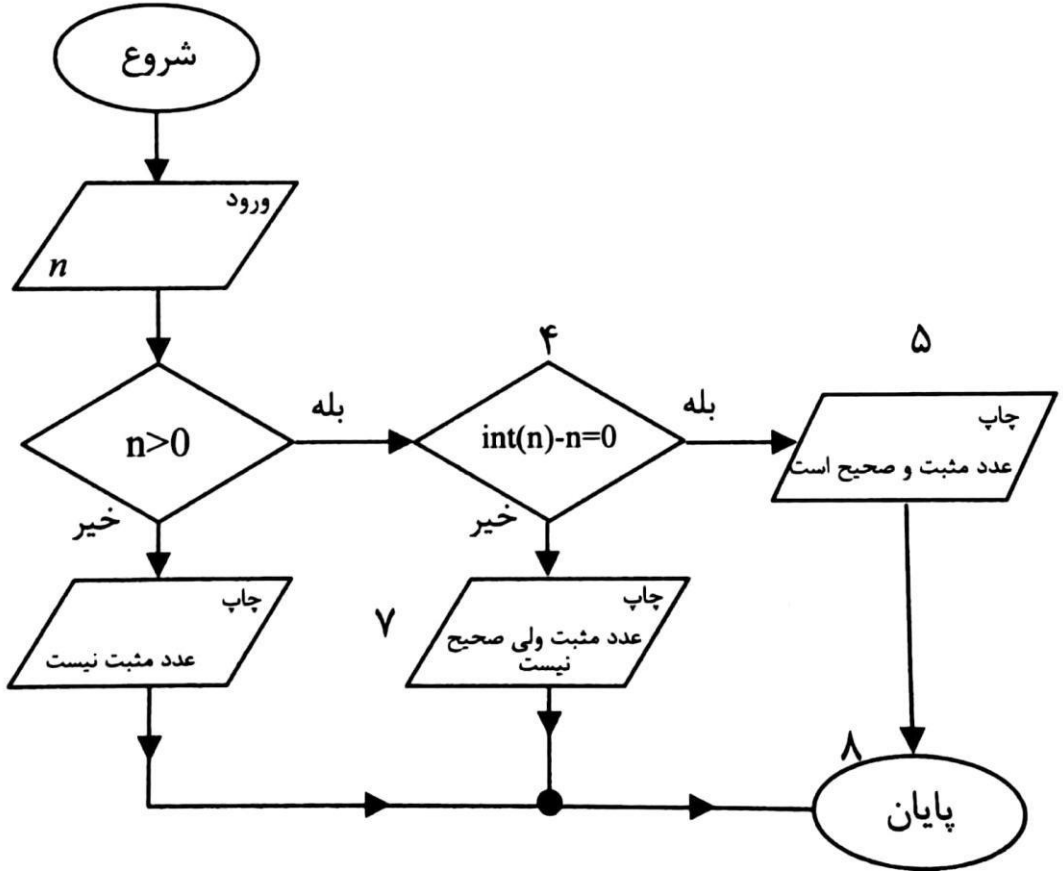
برنامه‌ای بنویسید که یک عدد صحیح مثبت را از صفحه کلید دریافت کرده، سپس **زوج یا فرد بودن** آنرا با پیام مناسبی نمایش دهد.

	n	$n \bmod ۲$	چاپ
نمونه اول	۳	۱	عدد فرد است
نمونه دوم	۶	۰	عدد زوج است

برنامه‌ای بنویسید که یک عدد صحیح مثبت را از صفحه کلید دریافت کرده، سپس **زوج یا فرد بودن** آنرا با پیام مناسبی نمایش دهد.

```
#include "stdafx.h"
#include <iostream>
#include <conio.h>
using namespace std;

int main()
{
  int n;
  cout<<"\n Enter n : ";
  cin>>n;
  if(n % 2 ==0)
    cout<<"\n Number is even";
  else
    cout<<"\n Number is odd";
  getch();
  return 0;
}
```

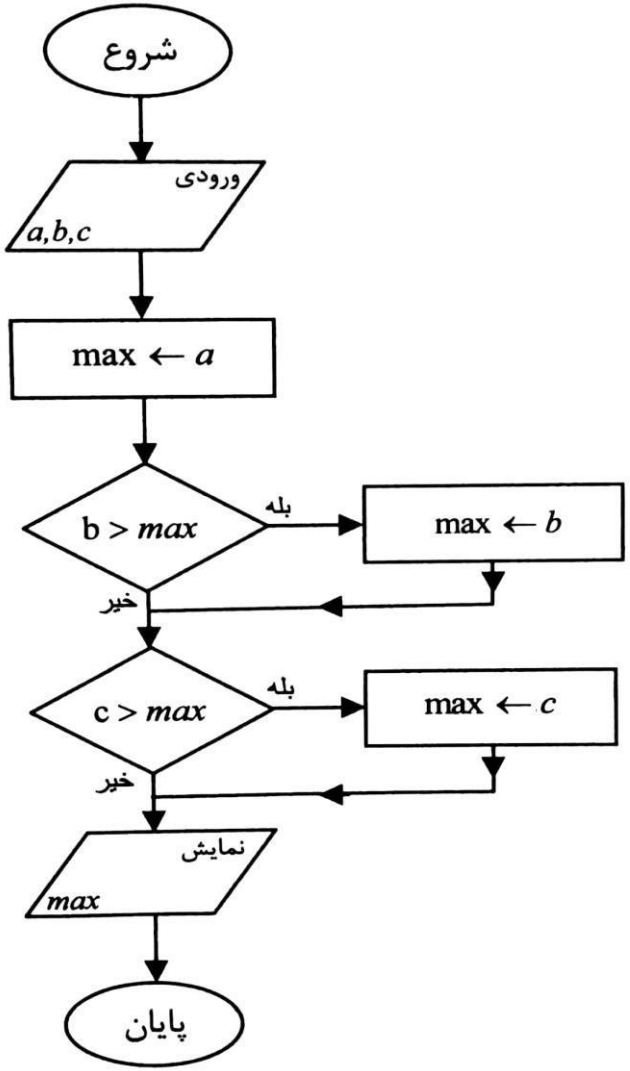
برنامه‌ای بنویسید که یک عدد را از صفحه کلید خوانده، سپس با چاپ پیام مناسبی مثبت و صحیح بودن آن را تعیین کند.



برنامه‌ای بنویسید که یک عدد را از صفحه کلید خوانده، سپس با چاپ پیام مناسبی مثبت و صحیح بودن آن را تعیین کند.

```
#include"stdafx.h"
#include<iostream>
#include<conio.h>
using namespace std;

int main()
{
  float n;
  cout<<"\n Enter n : ";
  cin>>n;
  if(n>0)
  {
    if((int(n)-n)==0)
      cout<<"\n N IS POSITIVE AND INTEGER ";
    else
      cout<<"\n N IS POSITIVE BUT IT IS NOT INTEGER ";
  }
  else
  cout<<"\n N IS NOT POSITIVE ";
  getch();
  return 0;
}
```

برنامه‌ای بنویسید که سه عدد a ، b و c را خوانده، سپس **بزرگترین عدد(ماکزیمم)** را تعیین کرده و چاپ نماید.

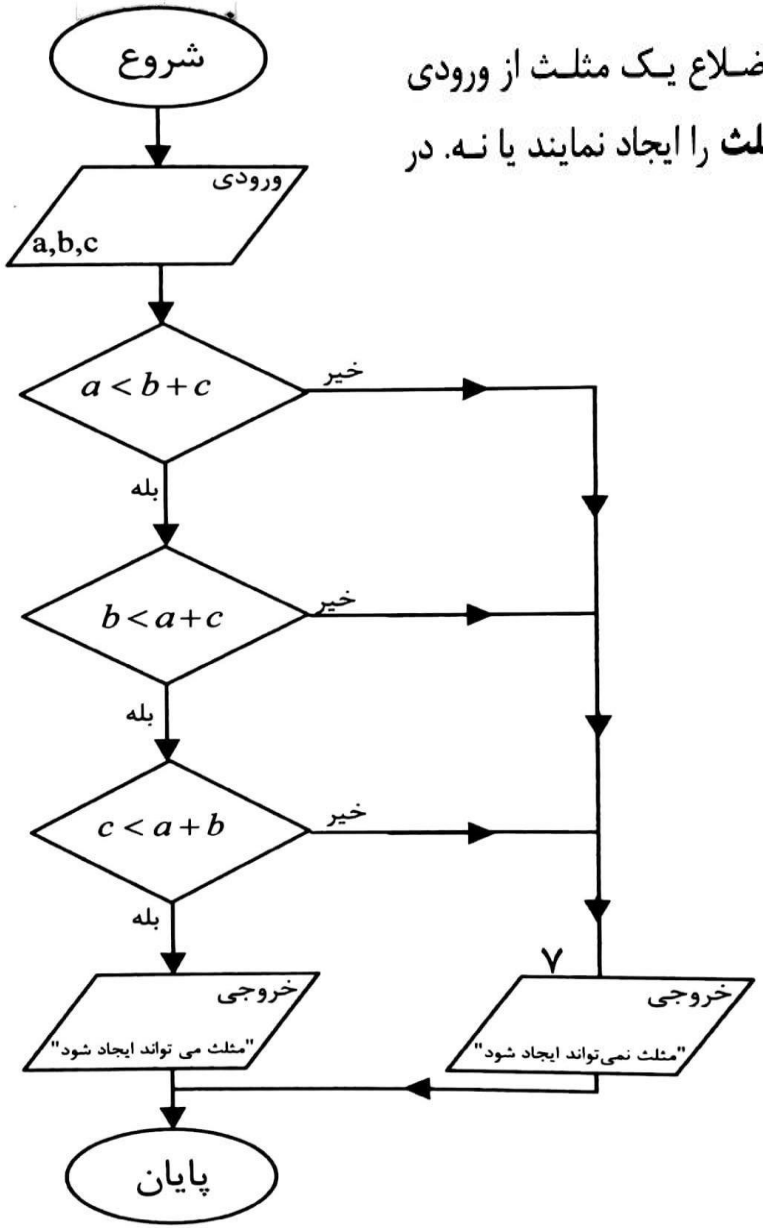


برنامه‌ای بنویسید که سه عدد a ، b و c را خوانده، سپس **بزرگترین عدد(ماکزیمم)** را تعیین کرده و چاپ نماید.

```
#include"stdafx.h"
#include<iostream>
#include<conio.h>
using namespace std;
int main()
{
  double a,b,c,max;

  cout<<"\n Enter a : ";
  cin>>a;
  cout<<" Enter b : ";
  cin>>b;
  cout<<" Enter c : ";
  cin>>c;
  max=a;
  if(b>max)
    max=b;
  if(c>max)
    max=c;
  cout<<"\n MAX = "<<max;
  getch();
  return 0;
}
```

دستورات شرطی

برنامه‌ای بنویسید که سه عـدد a ، b و c را بـه‌عنـوان طـول اضـلاع یـک مثلـث از ورودی دریافت نماید، تعیین کنید که آیا این **سه عدد می‌توانند یک مثلث** را ایجاد نمایند یا نـه. در هر مورد پیام مناسبی چاپ کند.



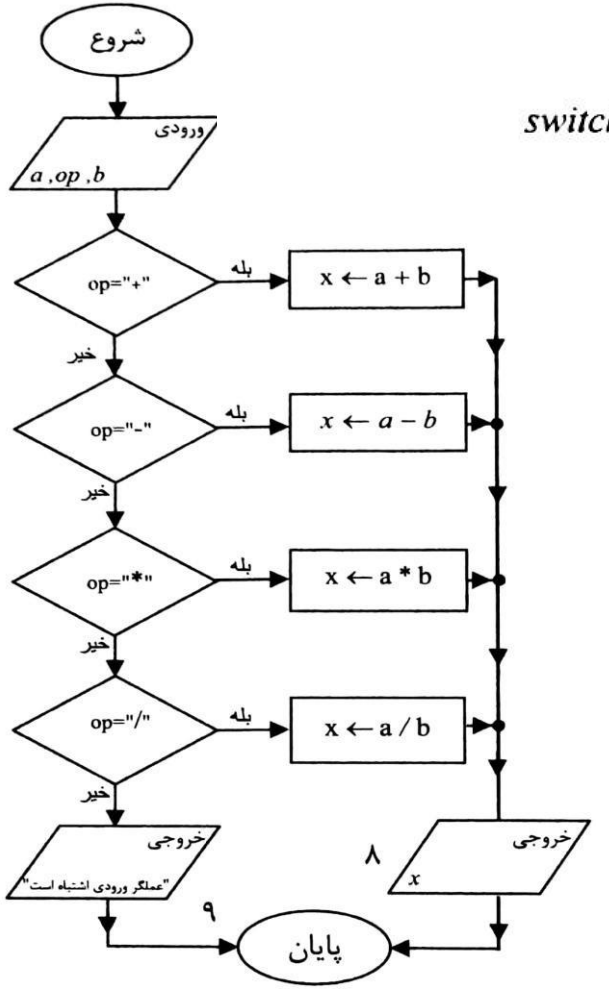
برنامه‌ای بنویسید که سه عـدد a ، b و c را بـه‌عنـوان طـول اضـلاع یـک مثلـث از ورودی دریافت نماید، تعیین کنید که آیا این **سه عدد می‌توانند یک مثلث** را ایجاد نمایند یا نـه. در هر مورد پیام مناسبی چاپ کند.

```
#include"stdafx.h"
#include<iostream>
#include<conio.h>
using namespace std;

int main()
{
  double a,b,c;

  cout<<"\n Enter a : ";
  cin>>a;
  cout<<" Enter b : ";
  cin>>b;
  cout<<" Enter c : ";
  cin>>c;
  if(a<b+c && b<a+c && c<a+b)
    cout<<"\n The triangle can be created ";
  else
    cout<<"\n The triangle can't be created ";
  getch();
  return 0;
}
```

دستورات شرطی

برنامه‌ای بنویسید که **دو عدد و یکی از چهار عملگر** اصلی را دریافت کند. سپس آن عملگر را روی دو عدد اعمال نموده و نتیجه را نمایش دهد. برنامه را با دو روش مختلف زیر پیاده‌سازی کنید:

- با استفاده از دستور *else if*
- با کاربرد دستور *switch*



برنامه‌ای بنویسید که **دو عدد و یکی از چهار عملگر** اصلی را دریافت کند. سپس آن عملگر را روی دو عدد اعمال نموده و نتیجه را نمایش دهد. برنامه را با دو روش مختلف زیر پیاده‌سازی کنید:

- با استفاده از دستور *else if*
- با کاربرد دستور *switch*

	a	*b*	*op*	*x*
نمونه اول	۶	۲	+	۸
نمونه دوم	۹	۳	-	۶
نمونه سوم	۴	۵	*	۲۰
نمونه چهارم	۲۱	۷	/	۳
نمونه پنجم	۸	۴	?	**عملگر ورودی اشتباه است**

برنامه‌ای بنویسید که **دو عدد و یکی از چهار عملگر** اصلی را دریافت کند. سپس آن عملگر را روی دو عدد اعمال نموده و نتیجه را نمایش دهد. برنامه را با دو روش مختلف زیر پیاده‌سازی کنید:

- با استفاده از دستور *else if*
- با کاربرد دستور *switch*

```
#include"stdafx.h"
#include<iostream>
#includ <conio.h>
using namespace std;
int main(){
  float a,b;
  char op;
  cout<<"\n Enter a : ";
  cin>>a;
  cout<<" Enter op: ";
  cin>>op;
  cout<<" Enter b : ";
  cin>>b;
```

برنامه‌ای بنویسید که **دو عدد و یکی از چهار عملگر** اصلی را دریافت کند. سپس آن عملگر را روی دو عدد اعمال نموده و نتیجه را نمایش دهد. برنامه را با دو روش مختلف زیر پیاده‌سازی کنید:

- با استفاده از دستور *else if*
- با کاربرد دستور *switch*

```
if(op == '+')
  cout<<"\n A+B = "<<a+b;
else if(op == '-')
  cout<<"\n A-B = "<<a-b;
else if(op == '*')
  cout<<"\n A*B = "<<a*b;
else if(op == '/')
  cout<<"\n A/B = "<<a/b;
else
  cout<<"\n The operator is wrong ";
getch();
return 0;}
```

برنامه‌ای بنویسید که **دو عدد و یکی از چهار عملگر** اصلی را دریافت کند. سپس آن عملگر را روی دو عدد اعمال نموده و نتیجه را نمایش دهد. برنامه را با دو روش مختلف زیر پیاده‌سازی کنید:

- با استفاده از دستور *else if*
- با کاربرد دستور *switch*

```
#include"stdafx.h"
#include<iostream>
#includ <conio.h>
using namespace std;
int main(){
  float a,b;
  char op;
  cout<<"\n Enter a : ";
  cin>>a;
  cout<<" Enter op : ";
  cin>>op;
  cout<<" Enter b : ";
```

برنامه‌ای بنویسید که **دو عدد و یکی از چهار عملگر** اصلی را دریافت کند. سپس آن عملگر را روی دو عدد اعمال نموده و نتیجه را نمایش دهد. برنامه را با دو روش مختلف زیر پیاده‌سازی کنید:

- با استفاده از دستور *else if*
- با کاربرد دستور *switch*

```
cin>>b;
switch(op)
{
  case  '+' :  cout<<"\n A+B = "<<a+b;
        break;
  case  '-' :  cout<<"\n A-B = "<<a-b;
        break;
  case  '*' :  cout<<"\n A*B = "<<a*b;
        break;
  case  '/' :  cout<<"\n A/B = "<<a/b;
        break;
  default   :  cout<<"\n The operator is wrong ";
}
getch();
return 0;}
```

دستورات شرطی

برنامه‌ای بنویسید که نمره‌ی یک دانشجو را که عددی بین صفر تا ۱۰۰ می‌باشد دریافت کـرده معادل حرفی آن‌را با توجه به جدول نمره‌ی زیر نمایش دهد.

- اگر نمره بین ۸۵ تا ۱۰۰ باشد، حرف 'A'
- اگر نمره بین ۷۰ تا ۸۴ باشد، حرف 'B'
- اگر نمره بین ۵۵ تا ۶۹ باشد، حرف C
- اگر نمره بین ۴۵ تا ۵۴ باشد، حرف D

درغیراین‌صورت Fail را نمایش دهد.

```
#include "stdafx.h"
#include <iostream>
#include <conio.h>
using namespace std;
int main( )
{
  float n;
  cout<<"\n Enter the g
  cin>>n;
  if(n >= 85 && n<=100)
    cout<<"\n Your alphabetic grade is : "<<'A';
  else if(n >= 70 && n<=84)
    cout<<"\n Your alphabetic grade is : "<<'B';
  else if(n >=55 && n<=69)
    cout<<"\n Your alphabetic grade is : "<<'C';
  else if(n >= 45 && n<=54)
    cout<<"\n Your alphabetic grade is : "<<'D';
    else
      cout<<"\n Your alphabetic grade is : "<<'Fail';
    getch();
    return 0;
  }
```

برنامه‌ای بنویسید که طول اضلاع یک مثلث را خوانده، تعیین کند که آیا مثلث متساوی‌الساقین است یا خیر؟

```
#include"stdafx.h"
#include<iostream>
#include<conio.h>
using namespace std;

int main(){
   float a,b,c;
   cout<<"\n Enter a: ";
   cin>>a;
   cout<<" Enter b: ";
   cin>>b;
   cout<<" Enter c: ";
   cin>>c;
   if((a==b) || (a==c) || (b==c))
     cout<<"\n The triangle is isosceles";
   else
     cout<<"\n The triangle is not isosceles";
   getch();
   return 0;
}
```

برنامه‌ای بنویسید که ***n* امین روز یک سال** را خوانده، سپس تعیین نمائید که ***n* چه روزی و از کدام ماه سال** می باشد.

```
#include "stdafx.h"
#include <iostream>
#include <conio.h>
using namespace std;

int main()
{
  unsigned int n,mon,month,day;

  do
  {
    cout<<"\n Enter n ==> 1=<n<=366 : ";
    cin>>n;
  }while (n<=1 || n>=366);
  if(n>186)
  {
    n -= 186;
    mon = int(n/30);
    day = n % 30;
    if( day == 0)
    {
      month = mon+6;
      day = 30;
    }
```

برنامه‌ای بنویسید که ***n* امین روز یک سال** را خوانده، سپس تعیین نمائید که ***n* چه روزی و از کدام ماه سال** می باشد.

```
    else
      month = mon+7;
  }
  else
  {
    mon = int(n/31);
  day = n%31;
  if( day == 0)
  {
    month = mon;
    day = 31;
  }
  else
    month = mon+1;
}
cout<<"\n MONTH = "<<month<<"\n DAY = "<<day;
getch();
return 0;
}
```

برنامه‌ای بنویسید که تعداد کارکترهای یک جمله ورودی را تا رسیدن به کارکتر '.' شمرده و آن‌را نمایش دهد.

```
#include "stdafx.h"
#include <iostream>d;
#include <conio.h>
using namespace std;
int main()
{
  int n;

  cout << "\n Enter a sentence that ends with (.) : \n ";
  for(n = 0; cin.get() != '.' ; n++);
  cout << "\n The length of sentence is = " << n ;
  getch();
  return 0;
}
```

دستورات شرطی

برنامه‌ای بنویسید که یک عدد ورودی از ۱ تا ۱۲ را به‌عنوان شماره‌ی یکی از ماه‌های سال شمسی دریافت کرده، سپس نام ماه متناظر با آن‌را نمایش دهد. برنامه باید ۴ بار اجرا شده و از ساختار switch در داخل حلقه‌ی for استفاده نماید.

```
#include "stdafx.h"
#include <conio.h>
#include <iostream>
using namespace std;

int main()
{
  int i,num;
  for(i=1; i<=4; i++)
  {
    cout<<"\n Enter the number of month : ";
     cin>>num;
    switch(num)
    {
      case 1 :  cout<<"\n Farvardin  \n"; break;
```

برنامه‌ای بنویسید که یک عدد ورودی از ۱ تا ۱۲ را به‌عنوان شماره‌ی یکی از ماه‌های سال شمسی دریافت کرده، سپس نام ماه متناظر با آن‌را نمایش دهد. برنامه باید ۴ بار اجرا شده و از ساختار switch در داخل حلقه‌ی for استفاده نماید.

```
        case 2 :  cout<<"\n Ordibehesht\n"; break;
        case 3 :  cout<<"\n Khordad    \n"; break;
        case 4 :  cout<<"\n Tier       \n"; break;
        case 5 :  cout<<"\n Mordad     \n"; break;
        case 6 :  cout<<"\n Shahrivar  \n"; break;
        case 7 :  cout<<"\n Mehr       \n"; break;
        case 8 :  cout<<"\n Aban       \n"; break;
        case 9 :  cout<<"\n Azar       \n"; break;
        case 10 : cout<<"\n Day        \n"; break;
        case 11 : cout<<"\n Bahman     \n"; break;
        case 12 : cout<<"\n Esfand     \n"; break;
        default : cout<<"\n Wrong Number \n";
      }
    }
    getch();
    return 0;
}
```

برنامه‌ای بنویسید که عدد اعشاری $n = ۱۳۸۹/۱۲۳۴۵۶۷۸۹$ را با دقت اعشار یک رقم، دو رقم، سه رقم تا ۹ رقم با استفاده از تابع کتابخانه‌ای setprecision() نمایش دهد.

```
#include "stdafx.h"
#include <iostream>
#include <conio.h>
#include <iomanip>
using namespace std;
int main(){
  unsigned i;
  long double n=1389.5123456789;
  for(i=1;i<=10;i++)
    cout<<setprecision(i+3)<< "\n "<<n;
  getch();
  return 0;
}
```

برنامه‌ای بنویسید که زمان اجرای مجموع مربعات اعداد ۱ تا ۱۰۰۰۰۰ را محاسبه کرده و نمایش دهد.

```
#include "stdafx.h"
#include <iostream>
#include <conio.h>
#include<ctime>
using namespace std;
int main()
{
  long i;
  double s=0.0;
  clock_t start,stop;
  start=clock();
  for(i=1;i<=100000;i++)
    s+=i*i;
  stop=clock();
  cout<<"\n The Execution Time = "<<(stop-start)/CLK_TCK;
  getch();
  return 0;
}
```

برنامه‌ای بنویسید که موقعیت حرکت را در چهار جهت اصلی (شمال، شرق، غرب و جنوب) تشخیص می‌دهد. کاربر هر بار یکی از کارکترهای n (North)، e (East)، w (West) و (South) s را زده و دو شمارنده x و y این جهات را در راستای محورهای x و y کنترل می‌نمایند. اگر کاربر کلید 'n' را فشار دهد مقدار متغیر y را یک واحد کاهش و اگر کلید 's' را بزند، به متغیر y یک واحد اضافه می‌نماید. درصورتی‌که کاربر کلید 'e' را فشار دهد به مقدار متغیر x یک واحد افزوده و درصورتی‌که کلید 'w' را بزند، از مقدار x یک واحد کم می‌کند. پایان برنامه با فشار دادن کلید Enter مشخص می‌شود. هر بار که کلیدی زده می‌شود، مقدار x و y را نمایش داده و از دستور Switch استفاده نمایید.

```
#include "stdafx.h"
#include <iostream>
#include <conio.h>
using namespace std;

int main()
{
  int x=20,y=20;
  char ch='a';
```

```
while(ch != '\r')

   cout<<"\n The Current Location is : x="<<x<<" "<< "Y="<<y<<"\n";
   cout<<"\n Press Enter to quit : ";
   cout<<"\n Enter Direction :s(South)،w(West) ،e(East) ،n(North) : ";
   ch=getche();
   switch(ch)
   {
     case 'n' :--y; break;
     case 's' :++y; break;
     case 'e' :++x; break;
     case 'w' :--x; break;
     case '\r':     break;
     default : cout<<"\n Type only one of these characters : n s e w";
   }
}
getch();
return 0;
}
```

برنامه‌ای بنویسید که کارکترهای تشکیل‌دهنده یک متن را یکی بعد از دیگری خوانده، تعداد کارکترها و کلمات را محاسبه کرده و نمایش دهد. در انتهای متن از کلید Enter استفاده می‌کنیم و بعد از هر کلمه می‌تواند بیش از یک کارکتر فاصله وجود داشته باشد.

```
#include "stdafx.h"
#include <iostream>
#include <conio.h>
using namespace std;

int main()
{
  int mychar=0,myword=1;
  char ch='a';
  cout<<"\n Enter a text : ";
  while(ch != '\r')
  {
    ch=getche();
    if(ch==' ')
      ++myword;
    else
      ++mychar;}
  cout<<"\n The number of characters : "<<mychar;
  cout<<"\n The number of words : "<<myword<<"\n";
  getch();
  return 0;}
```

برنامه‌ای بنویسید که کار یک ماشین را با چهار عملگر اصلی شبیه‌سازی کند. درصورتی‌که کاربر کلید 'Y' را بزند باید بتواند محاسبات را ادامه دهد، درغیراین‌صورت با زدن کلید دیگری از برنامه خارج شود.

```
#include "stdafx.h"
#include <iostream>
#include <conio.h>
using namespace std;

int main()
{
  double n1,n2,ans;
  char op,ch;
  do{
    cout<<" \n Enter first number,Operator and Second number : ";
    cin>>n1>>op>>n2;
```

برنامه‌ای بنویسید که کار یک ماشین را با چهار عملگر اصلی شبیه‌سازی کند. درصورتی‌که کاربر کلید 'Y' را بزند باید بتواند محاسبات را ادامه دهد، درغیراین‌صورت با زدن کلید دیگری از برنامه خارج شود.

```
  switch(op)
  {
    case '+' : ans=n1+n2; break;
    case '-' : ans=n1-n2; break;
    case '*' : ans=n1*n2; break;
    case '/' : ans=n1/n2; break;
    default : ans=0;
  }
  cout<<"\n Answer = "<<ans;
  cout<<"\n Do you want to continue ? (Enter 'y' or 'n') ";
  cin>>ch;
}while(ch=='y');
  getch();
  return 0;
}
```

برنامه‌ای بنویسید که دو عدد صحیح حداکثر ۹۹ رقمی را دریافت کرده، سپس مجموع آن‌ها را محاسبه کرده و نمایش دهد.

```
#include "stdafx.h"
#include <iostream>
#include <stdlib.h>
#include <conio.h>
using namespace std;

void main()
{
   int x1[99]={0}, x2[99]={0}, out[99]={0},i=-1,i2=-1,count,c;
   char temp[99];
   cout<<"\n Enter the  first number :  ";
   do //get it using getche()
   {
      i++;
      temp[i]=getche();
   }while(temp[i]!=13);
   i--;
   int ii=i;
```

برنامه‌ای بنویسید که دو عدد صحیح حداکثر ۹۹ رقمی را دریافت کرده، سپس مجموع آن‌ها را محاسبه کرده و نمایش دهد.

```
for(c=0;c<=i;c++) //convert 1st number from char to int
{
   x1[c]=temp[ii]-48;
   ii--;
}
cout<<"\n\n Enter the second number :  ";
do //get it using getche()
{
   i2++;
   temp[i2]=getche();
}while(temp[i2]!=13);
i2--;
ii=i2;
```

برنامه‌ای بنویسید که دو عدد صحیح حداکثر ۹۹ رقمی را دریافت کرده، سپس مجموع آن‌ها را محاسبه کرده و نمایش دهد.

```
for(c=0;c<=i2;c++) //convert 2nd number from char to int
{
   x2[c]=temp[ii]-48;
   ii--;
}
count=(i>i2)?i:i2; //recognize biggest array
int ex=0, temp2, temp3; //define some vars for temporary number and the extra
for(ii=0;ii<=count;ii++) //+ the arrays
{
   temp2=x1[ii]+x2[ii];
   temp2+=ex;
   if(temp2<10)
      out[ii]=temp2;
```

برنامه‌ای بنویسید که دو عدد صحیح حداکثر ۹۹ رقمی را دریافت کرده، سپس مجموع آن‌ها را محاسبه کرده و نمایش دهد.

```
        else
        {
           temp3=(temp2/10);
           ex=temp3;
           temp3*=10;
           out[ii]=(temp2-temp3);
        }
     }
     cout<<"\n\n          Result      : ";
    if(ex==1)
      cout<<1;
     for(;count>=0;count--)
     {
        cout<<out[count];
     }
     getch();
}
```