دانشکده علوم ریاضی

جزوه درس

مبانی کامپیوتر و برنامه سازی

(ویرایش اول، پاییز ۹۲)

فهرست مطالب

جزوه درس ۱
مبانی کامپیوتر و برنامه سازی ۱
بخش اول: مبانی و مفاهیم کامپیوتر ٤
۱,۱-مقدمه ٤
۱,۲-مفاهیم اولیه ٥
اطلاعات ٥
۱,۳-انواع کامپیوترها ۷
۱,۴-اجزای کامپیوتر: ۸
سخت افزار ۸
۱,۵-نرم افزار کامپیوتر ۱۲
۱,۶-نمایش اطلاعات در کامپیوتر ۱٤
سیستم اعداد ۱٤
تبدیل مبناها ۱٥
مبناهای دو، هشت، شانزده ۱٦
۱,۷-کد اسکی ۱۹
تمرینها ۱۹
بخش دوم: الگوریتم و فلوچارت ۲۰
۲,۱-الگوریتم ۲۰
۲,۲-قرارداد کار با الگوریتم ها ۲۱
۲,۳-جملات شرطی ۲۲
۲,۴-شرطی نوع ساده: ۲۲
۲,۵-شرطی نوع دو: ۲۳
۲,۵-حلقه های تکرار ۲٥
۲,۶-فلوچارت: ۳۳
۲,۷-علائم فلوچارت ۳۳
تمرینها ۳٥
بخش سوم: آشنایی با C++ ۳۷
۳,۱تاریخچه: ۳۷
۳,۲-نحوی اجرای یک برنامه در C++ ۳۸
۳,۳-یک نمونه برنامه در C++: ٤۰
۳,۴-مفاهیم اولیه زبان C++ ٤۳
شناسه ها ٤۳

۳,۵-داده ها در C++ ٤٤
۳,۶-تعریف متغییرها ٤٥
۳,۷-عملگرها ٤٥
۳,۸-عملگر محاسباتی: ٤٦
۳,۹-عملگرهای مقایسه ای ٥٠
10.۳.عملگرهای منطقی ٥١
۳,۱۱-عملگر شرطی ٥١
۴,۱۱-دریافت داده از کاربر و نمایش اطلاعات به کاربر ٥٢
تمرینها ٦٥
بخش چهارم: ساختارهای کنترلی ٦٦
۴,۱-مقدمه ٦٦
۴,۲-ساختارهای انتخاب ٦٦
۴,۳-ساختار دستور انتخاب if ٦٧
۴,۴-ساختار دستور انتخاب if/else ٦٨
۴,۵-ساختار چند انتخابی switch ٧١
۴,۶-ساختارهای تکرار ٧٣
۴,۷-ساختار تکرار While ٧٣
۴,۸-حلقه های تکرار for ٧٥
۴,۹-حلقه های for تو در تو ٧٧
۴,۱۰-حلقه های do/while ٧٨
۴,۱۱-دستورات break و continue ٧٩
تمرینها: ٨٢
مراجع: ٨٥

بخش اول: مبانی و مفاهیم کامپیوتر

۱,۱-مقدمه

امروزه کامپیوتر به عنوان ابزار قدرتمندی در زمینه های مختلف مورد استفاده قرار می گیرد و به جرات می توان بیان کرد که انجام بسیاری از فعالیت های پژوهشی بدون حضور کامپیوتر سخت و گاهی غیر ممکن می باشد. استفاده از کامپیوتر بخصوص در چند دهه ی اخیر منجر به فعالیت ها ی بسیاری شده است که از جمله آنها می توان به اینترنت ، شبکه های محاسباتی گرید و پروژه ژنوم انسان و... اشاره کرد. اینترنت امکان دسترسی از راه دور به کامپیوتر دیگر و انبارههای اطلاعات در هرجای دنیا که باشندرا به کاربران کامپیوتر می‌دهد. شبکه های محاسباتی گرید مجموعه ای از چندیدن سیستم با قدرت محاسباتی متفاوت می باشد که با متصل شدن این قدرت محاسباتی حاصل یک ابر رایانه ی مجازی شکل میگرید که با استفاده از آن می توان بسیاری از محاسبات پیچیده ی ریاضی،نجوم،زیست و... در زمان بسیار کمی انجام داد.شعار شبکه های محاسباتی نادیده گرفتن نگرانی های ناشی از محدودیت سخت افزاری سیستم های کامپیوتری می باشد.در پروژه ژنوم انسان با استفاده از برنامه های کامپیوتری توانستند توالی ژن های انسان را بدست آورند و آنها را در پایگاه داده ای نگهداری کنند و برای تحقیقات بیشتری از آن استفاده کنند.کامپیوترها از زمان پیدایش خود تا کنون نسل های مختلفی را سپری کرده اند که خالی از لطف نیست که نگاه کوتاهی بر آنها داشته باشیم.

کامپیوترهای نسل اول : این کامپیوترها که در اوایل دهه ۱۹۵۰ ساخته شدند از لامپ خلاء بعنوان جزو اصلی خود استفاده می کردند که در نتیجه حجم بسیار بالایی داشته و انرژی بالایی را نیز مصرف می نمودند. انیاک(Eniac) یکی از کامپیوترهای معروف این دوره بود.

کامپیوترهای نسل دوم : که در اوایل دهه ۱۹۶۰ ابداع گردیدند. ویژگی مهم آنها استفاده از ترانزیستور بجای لامپ خلاء بود که باعث کاهش اندازه کامپیوترها گردید.

کامپیوترهای نسل سوم : این کامپیوترها در سال ۱۹۶۴ با ابداع مدارات مجتمع IC که صدها ترانزیستور را در یک فضای کوچک جای می داد، ایجاد شدند. ابداع مدارات مجتمع باعث بالا رفتن سرعت و کاهش بیشتر حجم کامپیوترها گردید.

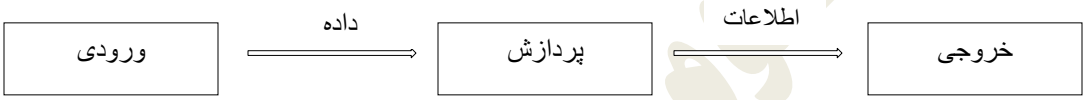
نسل چهارم کامپیوترها : در اواسط دهه ۱۹۷۰ با ابداع مدارات مجتمع با فشردگی بالا، حجم کامپیوترها بازهم کاهش یافت و پای آنها را به کاربردهای خانگی و اداری باز کرد.

نسل پنجم کامپیوترها: یا نسل کامپیوترهای هوشمند که قادر به انجام اعمالی همانند استنتاج و استدلال مانند انسانها باشند. این نسل هنوز تا رسیدن به وضعیت ایده آل راه درازی دارد.

۱,۲-مفاهیم اولیه

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. شکل ۱ نمایی از یک سیستم کامپیوتری را نشان می دهد.



شکل ۱: نمایی از یک سیستم کامپیوتری

داده[۱]: مجموعه مطالبی که وارد کامپیوتر می شود داده گفته می شود. داده ها می توانند به صورت عدد، حرف، صدا، تصویر و ... باشد.

پردازش[۲]: به کلیه فعالیت های صورت گرفته برروی داده ها که منجر به پیدایش نتایج می شود پردازش داده ها گفته می شود.

اطلاعات[۳]: به خروجی که بعد از پردازش داده ها تولید می شود اطلاعات اطلاق می گردد.

الگوریتم [۴]: دستورالعملهایی که برای کامپیوتر نوشته می شود را الگوریتم گوییم .

برنامه کامپیوتری[۵] : به تشریح الگوریتم ها برای کامپیوتر با استفاده از یک زبان برنامه سازی گفته می شود.

زبان برنامه سازی[۶] :

[1] Data
[2] Processing
[3] Information
[4] Algorithm
[5] Computer Programing
[6] Programing Languages

زبانی است که برای کامپیوتر قابل فهم بوده و الگوریتمها با استفاده از آن به کامپیوتر داده می شوند. مهندسین نرم افزار تاکنون برای زبانها پنج نسل را در نظر گرفته اند:

زبان نسل اول[7]: 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[8] مخصوص به خود را دارد که وابسته به سخت افزار خود آن کامپیوتر است. به عنوان مثال قطعه کد زیر می تواند اینگونه تعبیر شود که basepay و overpay را باهم جمع کن و حاصل آن را در grosspay ذخیره کن:

+1300042774
+1400593419
+1200274027

زبان نسل دوم[9]: زبان اسمبلی[10] است که حالت نمادین زبان ماشین است و در آن دستورات با استفاده از یک نماد بجای بجای کد صفر و یک نوشته می شوند. کد اسمبلی مثال بالا بصورت زیر است:

load basepay
add overpay
store grosspay

نکته: زبان ماشین و زبان اسمبلی جزء زبانهای سطح پایین هستند.

زبان نسل سوم[11]: این نسل شامل زبان های سطح بالا است که از جمله زبان های این نسل می توان به زبان های C، C++، C#، PASCAL، Basic، FORTRAN، JAVA و... اشاره کرد.برنامه نویسی به این زبانها بسیار نزدیک به زبان انسان هستند و از دستوراتی مشابه زبان طبیعی (اغلبا زبان انگلیسی) تشکیل شده اند. برای مثال بالا داریم:

grosspay = basepay + overpay

[7] first-generation language (1GL)
[8] machine language
[9] second-generation language (2GL)
[10] assembly language
[11] Third-generation languages (3GLs)

زبان نسل چهارم[12]: این نسل شامل زبانهای بسیار سطح بالا هست که از جمله زبان های این نسل می توان به SQL [13]اشاره کرد که زبانی است خاص منظوره و همچون زبان طبیعی که برای دریافت اطلاعات از پایگاه داده[14] بکار می رود. در زیر نمونه ای از دستورات این زبان ذکر شده است:

```
SELECT *
FROM Customers
WHERE Balance > 50
```

با اجرای این دستورات کلیه اطلاعات مشتریان پایگاه داده که موجودی حساب آنها بیشتر از ۵۰ دلار است نمایش داده می شود.

زبان نسل پنجم[15]: 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.

۱,۳-انواع کامپیوترها

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

ابر رایانه ها[16] :این نوع رایانه ها، قوی ترین و گرانترین نوع رایانه ها است که از قرت 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[17] : 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[18]: از آنجا که قیمت کامپیوترهای بزرگ بسیار بالا بود، در اواخر دهه ۱۹۵۰ کامپیوترهای کوچک وارد بازار شدند که توان محاسباتی کمتری داشتند و توسط سازمانهای کوچکتر مورد استفاده قرار می گرفتند.

---

[12] Fourth-generation languages (4GLs)
[13] Sequential Query Languages
[14] Database
[15] Fifth-generation languages (5GLs)
[16] Super Computers
[17] mainframe
[18] minicomputer

ریزکامپیوتر[19] : در آغاز دهه ۱۹۸۰ ریزکامپیوترها یا کامپیوترهای شخصی با قیمت پایین و حجم بسیار کوچک وارد بازار شدند و مورد استقبال مردم و افراد عادی قرار گرفتند.

۱,۴-اجزای کامپیوتر:

هر کامپیوتر از دو قسمت اصلی تشکیل شده است :

- **سخت افزار**[20] : کلیه دستگاههای الکتریکی، الکترونیکی و مکانیکی تشکیل دهنده یک کامپیوتر را سخت افزار آن می گوییم.
- **نرم افزار**[21] : مجموعه برنامه هایی هستند که برای یک کاربرد خاص نوشته شده اند و بدون آنها سخت افزار قادر به کاری نیست.

که هریک رابطور دقیقتر مورد بررسی قرار می دهیم.

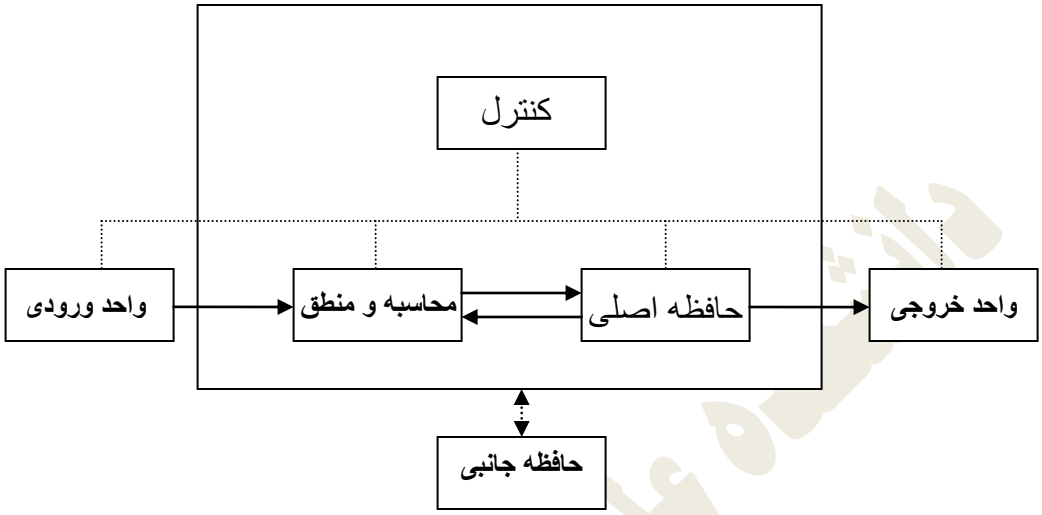
سخت افزار

اجزای تشکیل دهنده سخت افزار کامپیوتر عبارتند از : واحد ورودی، واحد خروجی، واحد حافظه، واحد محاسبه و منطق ،واحد کنترل و حافظه جانبی. نحوی ارتباط این واحد ها در شکل 2 نمایش داده شده است.

واحد ورودی[22]: 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[23]، ماوس[24]، صفحه لمسی[25]، قلم نوری، اسکنر[26] و ...

واحد خروجی[27] : این بخش وظیفه انتقال اطلاعات از کامپیوتر به محیط خارج را بعهده دارد و مهمترین دستگاههای خروجی عبارتند از :صفحه نمایش[28]، چاپگر[29]، رسام[30]، بلندگو [31]و ...

[19] microcomputer
[20] Hardware
[21] Software
[22] Input Unit
[23] Keyboard
[24] Mouse
[25] touch screen
[26] Scanner
[27] Output unit
[28] Monitor
[29] printer



شکل۲: اجزای تشکیل دهنده یک سیستم کامپیوتری

واحد محاسبه و منطق[۳۲] : 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[۳۳]: 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 CPU[۳۴] هستند. اجزای دیگر تشکیل دهنده CPU عبارتند از ثبات ها[۳۵] و حافظه نهان یا کش[۳۶].

---

[30] plotter
[31] speaker
[32] Arithmetic/logic unit
[33] Control unit
[34] Central Processing Unit
[35] Register
[36] Cache

ثباتها حافظه هایی با حجم بسیار کمی هستند که داده ها برای پردازش و بعد از پردازش در هنگام انتقال به حافظه در ثبات ها قرار می گیرند.

حافظه کش یک نوع حافظه با ظرفیت کم ولی بسیار سریع که بین CPU و حافظه اصلی قرار می گیرد سرعت دستیابی اطلاعات از حافظه کش نسبت به حافظه اصلی بیشتر و نسبت به حافظه های جانبی خیلی بیشتر است. بدین گونه که داده هایی را که CPU از حافظه اصلی فراخوانی می کند پیش از ارسال در کش قرار می گیرد که در فراخوانی های بعدی آن داده ها، با سرعت بیشتری به CPU ارسال شوند.

واحد حافظه[۳۷] : 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:

۱. **حافظه اصلی**[۳۸]: 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**: [۳۹] این حافظه قابل خواندن و نوشتن می باشد و برای ذخیره اطلاعات کاربران بکار می رود.
- **حافظه فقط خواندنی**: [۴۰] 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.

۲. **حافظه جانبی**[۴۱]: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[۴۲] ، Flash Memory ها و... را نام برد.

[37] Memory
[38] Main Memory
[39] RAM Random Access Memory
[40] ROM Read Only Memory
[41] Secondary Memory
[42] Hard disk

در هنگام کار با داده ها در حافظه با اصلاحاتی روبرو می شویم که در زیر به معرفی آنها می پردازیم

بیت[۴۳]: حافظه از واحدهای کوچکی بنام بیت تشکیل شده است که هر بیت قابلیت نگاهداری یک 0 یا 1 را در خود دارد.

بایت[۴۴]: به هر ۸ بیت یک بایت گفته می شود که واحد اندازه گیری حافظه است.

کاراکتر[۴۵]: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 ASCII بود که برای نمایش هر کاراکتر از یک بایت استفاده می کرد. از آنجا که هر بایت می تواند بین 0 تا 255 تغییر کند، بنابراین تا ۲۵۶ کاراکتر قابل تعریف است. از این بین کدهای بین 0 تا 127 بصورت استاندارد برای علائم و حروف انگلیسی تعریف شده است و کدهای بالاتر از ۱۲۷ برای هر کشور خالی گذاشته شده است تا بتوانند حروف خاص زبان خود را تعریف کنند. شکل۳ کدهای ASCII حروف بزرگ انگلیسی را نشان می دهد:

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

شکل۳

فیلد[۴۶]: 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[۴۷] :به مجموعه ای از فیلدهای مرتبط به هم یک رکورد می گوئیم. به عنوان مثال یک رکوردی تحت عنوان کارمند شامل فیلدهای زیر است:

[43] bit
[44] Byte
[45] character
[46] Fields
[47] Record

- فیلد شماره کارمندی
- فیلد نام کوچک
- فیلد نام خانوادگی
- فیلد سال تولد
- فیلد آدرس
- و ....

فایل[۴۸]:به مجموعه ای از رکوردهای مرتبط به هم فایل گفته می شود. به عنوان مثال فایل کارمندان شامل چندین رکورد کارمند است.

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 زیر آ نها را بیان می کنیم:

کلمه[۴۹]:معمولا هر ۲ یا ۴ بایت را یک کلمه در نظر می گیرند

کیلوبایت(KB)[۵۰]: هر ۱۰۲۴ بایت یک کیلو بایت را تشکیل می دهد. در سیستم دودویی هر کیلو بایت معادل $۲^{۱۰}$ می 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

۱,۵-نرم افزار کامپیوتر

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 :

[48] File
[49] word
[50] Kilo Byte

- نرم افزارهای کاربردی[۵۱] : 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

۱. **سیستم عامل** [۵۲]: 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 : Windows که بیشتر در منازل و محیطهای اداری مورد استفاده قرار می گیرد و Linux که بیشتر در محیطهای دانشگاهی و بعنوان سرویس دهنده استفاده می شود. سیستم عامل Unix نیز بیشتر در کامپیوترهای بزرگ نصب می شود. سیستم عامل Android نیز که امروزه بطور گسترده ای در موبایل ها و تبلت ها مورد استفاده قرار می گیرند.

۲. **برنامه های کمکی**[۵۳]: 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. دو نوع اصلی مترجم داریم که عبارتند از:

- **کامپایلر** [۵۴]**:** 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.

[51] Application Software
[52] Operation System
[53] Utilitues

- مفسر [55]: برنامه زبان سطح بالا را دستور به دستور به زبان ماشین تبدیل و همزمان آن را اجرا می کند.

۱,۶-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 ۲ استفاده 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 :

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

بعنوان مثال عدد۶۵۰۹۸ بصورت زیر تفسیر می گردد :

$$(98065)_{10} = 5 \times 10^0 + 6 \times 10^1 + 0 \times 10^2 + 8 \times 10^3 + 9 \times 10^4$$

در سیستم دهدهی می توان از ۱۰ رقم که از مجموعه ارقام {0,1,2,3,4,5,6,7,8,9} تشکیل شده اند، استفاده کرد.

اما می توان اعداد را در هر مبنای دلخواه دیگری مانند b نیز نشان داد در اینصورت هر عدد مانند N در مبنای b بصورت زیر تفسیر می گردد :

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

در سیستم عددی بر مبنای b می توان از b رقم که از مجموعه ارقام {0,1,2,3,…,b-2,b-1} تشکیل شده اند، استفاده کرد

[54] Compiler
[55] Interpreter

برای مثال نمایش اعداد در مبنای ۷ را در نظر بگیرید. ارقامی که برای نمایش اعداد در این مبنا بکار می رود از مجموعه ارقام $\{0,1,2,3,4,5,6\}$ تشکیل شده اند. با در نظر گرفتن این مبنا نمایش های اعدادی که دراین مبنا در ستون سمت چپ جدول آمده اند درست و نمایش اعداد در ستون سمت راست جدول ۱ نادرست است

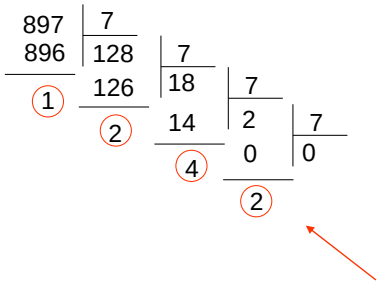
$(123456)_7$	✓	$(7654321)_7$	✗
$(654632)_7$	✓	$(5432816)_7$	✗

جدول ۱

تبدیل مبناها

برای تبدیل یک عدد از مبنای ۱۰ به هر مبنای دلخواه b، از روش تقسیمات متوالی استفاده می گردد. بدین ترتیب که عدد مورد نظر بر b 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 0 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 عدد حاصل از این فرایند عدد مورد نظر در مبنای b را است.

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



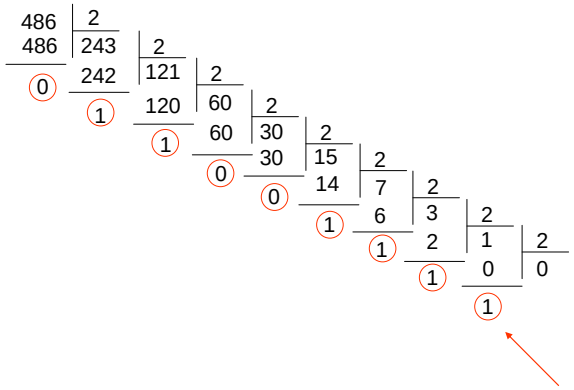
$(897)_{10} = (2421)_7$

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

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

مبناهای دو، هشت، شانزده

همانطور که قبلا نیز گفته شد واحد نگهداری اطلاعات در کامپیوتر بیت می باشد که هر بیت قادر به نگهداری 0 و یا 1 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 ۱۰ به ۲ و بالعکس بسیار ساده و همانند سایر مبنا ها است. در زیر عدد $(486)_{10}$ را به مبنای ۲ تبدیل می کنیم.



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

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

$(111100110)_2 = 0 \times 2^0 + 1\times2^1 + 1\times2^2 + 0\times2^3 + 0\times2^4 + 1\times2^5 + 1\times2^6 + 1\times2^7 + 1\times2^8 =$
$0+2+4+0+0+32+64+128+256=486$

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

از این رو تبدیل از مبنای ۲ به مبنای ۸ به سادگی انجام می گیرد، ابتدا ارقام را از راست به چپ بصورت دسته های ۳تایی تقسیم کنید و سپس برای هر دسته معادل آن را در مبنای ۸ قرار دهید. و برای دسته آخر اگر تعدادشان کمتر از ۳ بود به سمت چپ آن ۰ اضافه می کنیم تا تعدادشان به ۳ برسد.

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

$$\underbrace{0\,1\,0}_{2}\,\underbrace{0\,1\,1}_{3}\,\underbrace{1\,1\,1}_{7} = (237)_8$$

همانطور که مشاهده می شود نمایش یک عدد در مبنای هشت نسبت به نمایش همان عدد در مبنای دو ارقام کمتری دارد.

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 ۱۶ رقم داریم درحالیکه ارقام موجود فقط ۱۰ تا است. بهمین دلیل از حروف A تا F برای ارقام ۱۰ تا ۱۵ استفاده 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

برای تبدیل اعداد از مبنای ۲ به ۱۶ ازهمان 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 ۴ بود به سمت چپ آن ۰ اضافه می کنیم تا تعدادشان به ۴ برسد. در زیر تبدیل عدد $(1010011101)_2$ به مبنای 16 را نشان می دهد

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

همانطور که مشاهده می کنید یک عدد در مبنای شانزده نیز نسبت به نمایش همان عدد در مبنای دو ارقام کمتری دارد.

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

برای تبدیل یک عدد اعشاری از مبنای ۱۰ به مبنای ۲ نیز کافی است عددرابه صورت مجموع توانهای منفی عدد ۲ نوشت به عنوان مثال داریم:

$(0.75)=(0.11)2=1\times 2^{-1}+1\times 2^{-2}=0.5+0.25$

از قاعده زیر استفاده میکنیم:Nبرای تبدیل عدد

مراحل زیر را انجام بده۱N#0-تا زمانی که

۱,۱-قسمت صحیح int N=N×2

۱,۲-قسمت اعشاری float N=N×2

۱,۳-چاپ گن int N

۱,۴-float N=N

برای تبدیل یک عدد اعشاری از مبنای ۲ به مبنای ۱۰ کافی است ،هر رقم را درتوانی از ۲ ضرب نمود،که اولین توان ۲ بعداز ممیز 1- شروع و برای یک عدد اعشاری n رقمی به 2^{-n} ختم میشود.

مثلا:

$(0.101)_2=1\times2^{-1}+0\times2^{-2}+1\times2^{-3}=0.5+0.25+0.125=0.875$

۱,۷-کد اسکی

در سال ۱۹۶۸ موسسه ملی استاندارد آمریکا یک کد ۷بیتی برای تمام حروف الفبا ،ارقام ۰تا ۹،نمادهای نقطه گذاری که در اغلب ماشین های تحریر به کار میرود و چند کد کنترل خاص پایه گذاری کرد.آنها این کد را کد اسکی(ASCII) نامیدند.

گرچه کد اسکی یک کد ۷بیتی است ولی غالبا به صورت یک بایت (۸بیت) نوشته شده از بیت هشتم آن یا صرفنظر میشود یا برای توازن استفاده میشود.فرضا در این کدگذاری کد حرف بزرگ L برابر با 0100,1100یا 4C در مبنای ۱۶ است.امروزه تقریبا متنها با فرمت کد اسکی کد میشوند، به این ترتیب انتقال اطلاعات بین دو سیستم کامپیوتری مختلف ممکن میشود.هربار که یکی از کلیدهای صفحه کامپوتر را فشار دهید یک کد اسکی برای پردازش به کامپیوتر فرستفاده میشود.

لازم به ذکر است که علاوه بر کداسکی کدهای دیگری نیز وجوداردمانند کد گری وکد .BCD

تمرینها

۱-اعداد زیر را به مبناهای خواسته شده تبدیل نمایید.

$(1234)_{10}=(?)_6$

$(1030)_{10}=(?)_2$

$(1010101010101)_2=(?)_8=(?)_{16}$

$(ABCF)_{16}=(?)_{10}=(?)_8$

$(654)_8=(?)_{10}=(?)_2=(?)_{16}$

۲-نمایش دهدهی ۲۵۰/۵در مبنای ۳ چیست؟

بخش دوم: الگوریتم[۵۶] و فلوچارت[۵۷]

۲,۱-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.

تعریق 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."

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 کمیت خروجی داشته باشد.
- زبان دقیق: اگر از یک جمله برداشتهای متفاوتی شود و یا اینکه جمله مبهم باشد ، گوییم بیان آن جمله یا عبارت دقیق نیست.
- 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.

[56] Algorithm
[57] Flowchart

- ترتیب مراحل : مراحل انجام دستورالعملها باید به ترتیب انجام گیرد و در غیر اینصورت دستورالعملها 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

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

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

برای بیان الگوریتم ها از زبان طبیعی استفاده می کنیم. گاهی در قسمت هایی از الگوریتم از دستورات زبان برنامه نویسی خاصی استفاده می شود که به آنها شبه کد گفته می شود.

۲,۲-قرارداد کار با الگوریتم ها

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.

- 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 آنستکه مقادیر ذخیره شده در هریک از این خانه های حافظه می تواند تغییر کند. نام متغییر 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.

مثال2.1: 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

۲,۳-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

۲,۴-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.

۲,۵-شرطی نوع دو:

شکل کلی این نوع شرط به صورت زیر است:

اگر (عبارت شرطی) آنگاه دستورات۱

درغیر اینصورت دستورات ۲

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

مثال۲,۱: 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

مثال۲,۳: 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 $x_2 \leftarrow \frac{-b+\sqrt{delta}}{2a}$ و x1 و x2 را چاپ کن

۶. پایان

مثال۲,۴: 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 و c 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

۲,۵-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 " را نوشت. راه حل بهتر آنست که بگونه ای به الگوریتم بگوییم بنحوی عمل خواندن اعداد و جمع زدن آنها را ۵۰ بار تکرار کند.

حلقه تکرار بگونه ای است که مجموعه ای از دستوزات را تا زمانیکه شرط خاصی برقرار باشد تکرار می کند.

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 b را 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.

- 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.

مثال۵: الگوریتمی بنویسید که n عدد را به دلخواه از ورودی دریافت کند و حاصلجمع آنها را چاپ نماید.

برای حل این مسئله از متغیر i برای شمارش تعداد ورودی های دریافت شده استفاده می کنیم. از متغییر Sum برای ذخیره حاصلجمع اعداد خوانده شده استفاده می کنیم. از متغییر A برای دریافت اعداد استفاده می کنیم.

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

مثال۲,۶: 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 فاکتوریال : fact　　شمارنده حلقه : i　　عدد ورودی برای محسبه فاکتوریال آن : n

۱) شروع

۲) n را بخوان

۳) $i \leftarrow ۱$ و $fact \leftarrow ۱$

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

مثال۲,۷: الگوریتمی بنویسید که n عدد صحیح را دریافت و حداکثر و حداقل آنها را چاپ کند.

برای حل مسائل حداکثر یا حداقل ابتدا باید یک متغیر برای نگهداری آنها درنظر بگیریم. نکته مهم مقدار اولیه این متغیر است. دو روش کلی وجود دارد:

- اولین مقدار را دریافت و آن را بعنوان مقدار اولیه درنظر بگیریم.
- مقدار اولیه متغیر حداکثر را برابر کمترین عدد ممکن و متغیر حداقل را برابر بیشترین عدد ممکن درنظر بگیریم.

n : تعداد اعداد i : شمارنده حلقه A : عدد دریافت شده در هر مرحله

max : کوچکترین عدد صحیح min : بزرگترین عدد صحیح

۱) شروع

۲) n را بخوان

۳) A را بخوان

۴) $min \leftarrow A$ و $max \leftarrow A$

۵) $i \leftarrow 2$

۶) تا زمانیکه ($i \leq n$) دستورات ۷ تا ۱۰ را تکرار کن

۷) Aرا بخوان

۸) اگر $(A > max)$ آنگاه $max \leftarrow A$

۹) در غیر اینصورت اگر (A < min) آنگاه min ← A

۱۰) i ← i + 1

۱۱) max و min را چاپ کن

۱۲) توقف کن

مثال۲,۸: الگوریتمی بنویسید که عدد طبیعی n را دریافت کند و مجموع عبارات زیر را محاسبه و چاپ نماید:

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

۱. شروع

۲. N را بخوان

۳. S ←۱

۴. i ←۲

۵. تا زمانیکه (i ≤ n) دستورات 6 تا 7 را تکرار کن

۶. $s \leftarrow s + \frac{1}{i}$

۷. i ← 1+i

۸. s را چاپ کن

۹. توقف

$$s = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \ldots$$

۱. شروع

۲. N را بخوان

۳. s ←۰

۴. P ← ۱ ، k ← ۱ ، i ←۱

۵. تا زمانیکه (i ≤ n) دستورات 6 تا 9 را تکرار کن

۶. $s \leftarrow s + \frac{k}{P}$

۷. $k \leftarrow -k$

۸. $i \leftarrow 1+i$

۹. $P \leftarrow P+2$

۱۰. s را چاپ کن

۱۱. توقف

مثال۲,۹: الگوریتمی بنویسید که یک عدد را دریافت و وارون آن را محاسبه و بهمراه خود عدد چاپ کند. مثلا وارون عدد ۱۳۹۲ برابر ۲۹۳۱ می باشد.

برای وارون کردن هر عدد ابتدا باید ارقام آن را از سمت راست جدا کنیم. ساده ترین راه برای اینکار، بدست آوردن باقیمانده تقسیم عدد بر ۱۰ است. مثلا 1392 mod 10:2فرض می کنیم عملگر mod بمعنای باقیمانده قبلا تعریف شده است، درغیر اینصورت برای بدست آوردن باقیمانده عددی مانند a به شکل روبرو عمل می کنیم : a mod 10 = a – ((a/10) * 10)

تنها نکته آنستکه این فرمول برای اعداد منفی کار نمی کند. لذا برای اعداد منفی ابتدا باید آنها را مثبت کرده و در پایان مجددا تبدیل به عدد منفی می کنیم.

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

۱. شروع

۲. adadرا بخوان

۳. اگر (adad < 0) آنگاه a ← -adad در غیر اینصورتa ← 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) آنگاهvaroon ← -varoon

۱۰. adadو varoon را چاپ کن

۱۱. توقف کن

مثال۲,۱۰: الگوریتمی بنویسید که یک عدد صحیح مثبت در مبنای ۱۰ را دریافت و سپس آن را به مبنای b ببرد. مبنای b نیز از کاربر دریافت می گردد.

adad : عدد موردنظر b : مبنای موردنظر

mabnaieb : حاصل تبدیل عدد به مبنای موردنظر

remain : باقیمانده در هر مرحله i : شمارنده حلقه

۱. شروع
۲. adadو b را بخوان
۳. mabnaieb ← 0و i ← 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

مثال۲,۱۱: الگوریتمی بنویسید که برای دانشجویان ورودی ۹۱ دانشکده 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.

برای حل این مسئله باید ۲ حلقه متداخل یا تودرتو در نظر بگیریم. یک حلقه بیرونی که تعداد دانشجویان ورودی ۹۰ را می شمارد و یک حلقه داخلی که برای هر دانشجو شماره دانشجویی و تعداد دروس و تعداد واحد درس ها و معدل آن را حساب کند.

نکته مهم آنست که هنگامیکه دو حلقه متداخل داریم، باید از دو متغیر جداگانه برای شمارش هریک استفاده کنیم تا مشکلی پیش نیاید.

تعداد دروس هر دانشجو :dars شمارنده دانشجویان : i تعداد دانشجویان ورودی ۹۰ : n

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

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

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

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

۱ شروع

۲ n را دریافت کن

۳ i ← ۱

۴ total Ave ← ۰ ، maxAve ← -1 ، 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

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 minAve>moadel 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

مثال ۲,۱۲:الگوریتمی بنویسید که یک عدد را دریافت و تعیین کند که آیا اول است یا خیر؟

می دانیم که عددی اول است که بر هیچ یک از اعداد کوچکتر از خود بجز ۱ بخشپذیر نباشد. یک نگاه دقیقتر نشان می دهد که اگر یک عدد بر کلیه اعداد از ۲ تا جذر خود بخشپذیر نباشد، اول است. الگوریتم زیر برهمین اساس نوشته شده است. دقت کنید که الگوریتم بگونه ای است که به محض اینکه عدد مورد نظر بر یک عدد بخشپذیر باشد، از حلقه خارج شده و اعلام می کند که عدد موردنظر اول نیست.

نکته : همانگونه که می بینید در شرط حلقه از and استفاده شده است. بعبارت دیگر از یک شرط ترکیبی استفاده شده است. در شرطهای ترکیبی می توان از عملگرهای and و یا or استفاده کرد و معنای آنها نیز به شرح زیر است :

C_1 and C_2 : عبارت درصورتی درست است که هر دو شرط درست باشند

C_1 or C_2 : عبارت درصورتی درست است که حداقل یکی از دو شرط درست باشد

i : شمارنده حلقه root : ریشه عدد مورد نظر adad : عدد مورد نظر

primeSw : یک سویچ که نشاندهنده اول بودن(۱) یا اول نبودن (۰) عدد است

۱- شروع

۲- رابخوان adad

۳- $root \leftarrow \lfloor \sqrt{adad} \rfloor$

۴- $i \leftarrow 2$ و $primeSw \leftarrow 1$

۵- تازمانیکه ($i \leq root$ and $primeSw=1$) دستورات ۶ تا ۷ را تکرار کن

۶- اگر ($adad \bmod i = 0$) آنگاه $primeSw \leftarrow 0$

۷- $i \leftarrow i + 1$

۸- اگر ($primeSw = 1$) آنگاه چاپ کن "عدد اول است"

درغیراینصورت چاپ کن "عدد اول نیست"

۹- توقف کن

نکته مهم : پس از نوشتن الگوریتم باید الگوریتم را با داده هایی که کلیه حالات ممکن را پوشش می دهند چک کرد چراکه ممکن است الگوریتم بازای یک ورودی خاص دچار مشکل شود!

۲,۶-فلوچارت:

در گذشته برای درک بهتر الگوریتم ها و سهولت دنبال کردن دستورالعملهای 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.

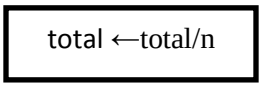
۲,۷-علائم فلوچارت

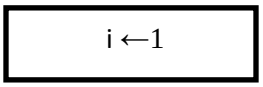
علامت شروع و پایان

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

علامت جایگزینی و انتساب

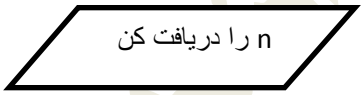
در فلوچارت برای انجام عمل جایگزینی و یا انتساب و یا عمل محاسباتی از مستطیل استفاده می شود.





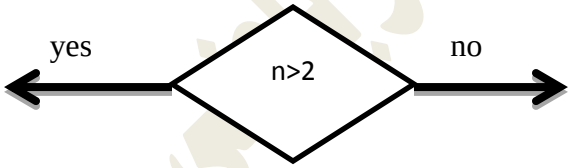
علامت ورودی

در فلوچارت از علامت متوازی الاضلاع برای گرفتن ورودی ها استفاده می شود.



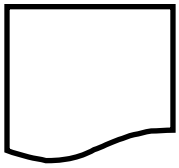
علامت شرط

در فلوچارت از علامت لوزی برای شرط استفاده می شود



علامت چاپ

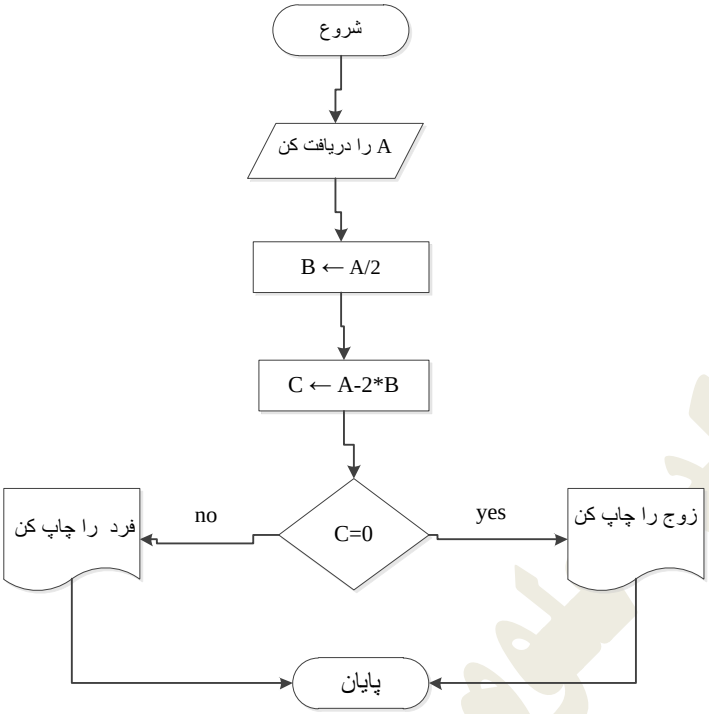
در فلوچارت از علائم زیر برای چاپ استفاده می شود



چاپ روی کاغذ

چاپ روی صفحه نمایشگر

از فلش های جهت دار به منظور ارتباط بین علائم استفاده می شود. استفاده از فلوچارت تنها برای الگوریتمهای کوچک مناسب است. از این رو ما تنها با استفده از مثال ۲ این بخش نحوی استفاده از آن را نمایش می دهیم.



تمرینها

۱-الگوریتمی بنویسید که زمان را بر حسب ثانیه دریافت کند و مشخص کند که از چند ساعت و چند دقیقه و چند ثانیه تشکیل شده است.

۲-الگوریتمی بنویسید که سه عدد a، b و c را که اضلاع یک مثلث هستند را دریافت و مشخص کند که می توان با آن یک مثلث قائم الزاویه تشکیل داد.

۳-الگوریتمی برای بنویسید که عددی را دریافت کن و مشخص کند که آن عدد کامل هست یا نه؟ (عدد کامل، یک عدد صحیح مثبت است که برابر با مجموع مقسوم‌علیه‌های سره مثبت خود (همه مقسوم‌علیه‌های مثبتش غیر از خود عدد) باشد.)

۴-الگوریتمی بنویسید که یک عدد در مبنای b را دریافت و آن را به مبنای ۱۰ ببرد.

۵-الگوریتمی بنویسید که یک عدد طبیعی را از ورودی دریافت و توان آن را به روش زیر محاسبه کند

توان دوم عدد n = مجموع n عدد فرد از ۱

$$6^2 = 1 + 3 + 5 + 7 + 9 + 11 = 36$$

۶-الگوریتمی بنویسید که ۵۰ جمله از دنباله زیر موسوم به دنباله فیبوناچی را چاپ کند.

1,1,2,3,5,8,…

۷-الگوریتمی بنویسید که عدد i را دریافت نموده و مجموع اعداد سطر iام مثلث زیر را بیابد

```
                1
             1     1
          1     2     1
       1     3     3     1
    1     4     6     4     1
 1     5     10    10    5     1
..    ..    ..    ..    ..    ..    ..
```

۸- الگوریتمی بنویسید که عدد n را دریافت نموده و مجموع n جمله از جملات سری زیر را بیابد

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

بخش سوم: آشنایی با C++

۳,۱تاریخچه:

C++ توسعه یافته زبان قابل حمل[58] C که توسط بیارنه استراوستروپ در اوایل ۱۹۸۰ در آزمایشگاه بل بوجود آمد. زبان C در سال ۱۹۷۲ توسط دنیس ریچی از روی زبان B و BCPL در آزمایشگاه بل ساخته شد ریچی از این زبان برای ایجاد سیستم عامل Unix استفاده کرد اما بعدها اکثر سیستم عاملهای دیگر نیز با همین زبان نوشته شدند. این زبان با سرعت بسیاری گسترش یافت. متاسفانه استفاده گسترده این زبان در انواع کامپیوترها و سخت افزارهای مختلف باعث شد که نسخه های مختلفی از این زبان بوجود آید که با یکدیگر ناسازگار بودند. در سال ۱۹۸۹انستیتوی ملی استاندارد آمریکا ANSI برای حل این مشکل استانداردی فا قد ابهام و مستقل از متن برای زبان C تعریف کردند. این استاندارد تحت عنوان ANSI C به تصویب رسید و سپس در سال ۱۹۹۰، سازمان استانداردهای بین المللی (ISO) نیز این استاندارد را پذیرفت و مستندات مشترک آنها تحت عنوان ANSI/ISO C منتشر گردید.

زبان های قابل حمل یا مستقل از متن زبانهایی هستند که به هیچ ماشین خاصی وابسته نیستند، برنامه های نوشته شده با این زبانها (تا حد زیادی) قابل حمل می باشند. در مقابل زبانهای قابل حمل زبانهای غیر قابل حمل وجود دارند که این زبانها وابسته به سخت افزاری که برروی آن نوشته شده اند می باشند که از جمله این زبانها می توان به زبان ماشین و اسمبلی اشاره کرد.

همانطور که گفتیم C++ توسعه یافته زبان C است که علاوه بر ویژگی های زبان C دارای خاصیت شی گرایی[59] نیز هست. با خاصیت شی گرایی در واحد های درسی آینده بیشتر آشنا خواهید شد.

برنامه نویسی به زبان C++ شامل کار با کلاس[60] ها و توابع[61] است. که اینها توسط خود برنامه نویس نوشته می شوند. البته خود زبان C++ دارای کتابخانه غنی از کلاس ها و توابع است که آشنایی با این کتابخانه ها بسیار مفید خواهد بود. که در طول این ترم و ترم های آینده با آنها آشنا خواهید شد.

[58] Portable
[59] Object Oriented
[60] Class
[61] Function

۳,۲-نحوی 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:

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 cpp.، cxx.، cc. یا C. در یکی از مکان های ذخیره سازی اطلاعات ذخیره می شود. یکی از ادیتور های محبوب برای نوشتن برنامه های C++ در سیستم عامل ویندوز Microsoft Visual C++ که شامل محیط مجتمع نرم افزار[۶۲](IDE) است . این محیطها دارای یک ویرایشگر متن می باشند که معمولا دارای خواص جالبی همچون استفاده از رنگهای مختلف برای نشان دادن اجزای مختلف برنامه مانند کلمات کلیدی، و یا قابلیت تکمیل اتوماتیک قسمتهای مختلف برنامه می باشد. از جمله ویرایشگرهای دیگری که می توان در محیط ویندوز از آن استفاده کرد برنامه ساده ویرایش متن Notepad است.

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،در قسمت های بعدی بیشتر درباره پیش پردازنده ها صحبت خواهیم کرد.

۳. **کامپایل**

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

[62] Integrated Development Environments

۴. **پیوند دهنده**[۶۳]

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**[۶۴]

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**[۶۵]

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

کلیه مراحل گفته شده در بالا در یک IDE بطور کامل پس از اجرای برنامه انجام می گیرد. علاوه براین، محیطها IDE معمولا دارای امکانات اشکالزدایی 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.

[63] Linker
[64] Loading
[65] Execution

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. بهمین 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.

b. 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!

در اصطلاح برنامه نویسی به هر گونه خطا، bug و به رفع خطا debug گفته می شود.

۳,۳-یک نمونه برنامه در 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++!
```

توضیحات برنامه:

خط اول یک توضیح درمورد برنامه است. در زبان C++ برای توضیحات یک خطی از علامت // استفاده می گردد. اما چنانچه توضیحات بیش از یک خط بود، آن را با علامت */ شروع کرده و با /* پایان دهید. کامپایلر از این توضیحات صرفنظر خواهد کرد. این توضیحات باعث می شوند که برنامه شما خواناتر شده و دیگران بهتر آن را درک کنند.

هر دستوری که با علامت # شروع شود، یک دستورC++ نیست، بلکه جزو دستورات پیش پردازنده محسوب می گردد. دستورات پیش پردازنده، دستوراتی هستند که توسط کامپایلر قبل از شروع به کامپایل انجام می شوند. بعنوان مثال دستور #include باعث می شود که تعاریف اولیه مربوط به توابعی (زیربرنامه هایی) که قصد استفاده از آنها را داریم به برنامه اضافه شود. در تمامی برنامه های C++ زمانی که قرار است اطلاعاتی در خروجی چاپ بشوند و یا زمانی که داده ای را از کاربر از طریق صفحه کلید دریافت کنیم از پیش پردازنده ی#include <iostream> استفاده می کنیم.

using namespace std; در Visual c++ این امکان را فراهم می کند که اطلاعات در صفحه نمایش نشان داده شود و یا از کاربر ورودی دریافت کند. در قسمت های بعدی در اینباره بیشتر توضیح می دهیم

هر برنامه C++ باید دارای تابعی به نام main باشد که اجرای برنامه از آن شروع می شود و در حقیقت همان برنامه اصلی است. البته می توان هر تعداد دیگری تابع (زیربرنامه) نیز تعریف کرد، اما وجود تابع main الزامی است. دقت کنید که گرچه این تابع پارامتر ورودی ندارد، اما از پرانتز باز و بسته تنها استفاده شده است.

در زبان C هر بلوک برنامه با علامت } آغاز شده و با { پایان می یابد.

cout << "Welcome to C++!"; به کامپیوتر دستور می دهد تا عملی را انجام بدهد و آن دستور این است که یک رشته را در خروجی چاپ کند. یک رشته از چندین کاراکتر تشکیل شده است که در C++ برای نمایش یک رشته آن را در داخل (“ ”) قرار می دهیم از این رو “Welcome to C++!” یک رشته است. cout << به کامپیوتر دستور می دهد تا

"Welcome to C++!" را در خروجی چاپ کند. دقت کنید که در پایان هر دستور در C++ از علامت ; استفاده شده می شود. در مجموع C++ یک زبان قالب آزاد است و شما می توانید دستورات را به هر نحوی که دوست دارید قرار دهید (مثلا چند دستور در یک خط از برنامه). تنها چیزی که نشاندهنده پایان یک دستور است، علامت ; است(و نه انتهای خط).

از آنجا که C++ یک زبان قالب آزاد است، می توان با استفاده از مکان نوشتن دستورات شکل بهتری به برنامه داد. بعنوان مثال دقت کنید که پس از شروع تابع main، دستورات حدود ۳ کاراکتر جلوتر نوشته شده اند. به این نحوه نوشتن دستورات دندانه گذاری می گویند. بطور کلی هربار که بلوک جدیدی آغاز می شود، باید آن را کمی جلوتر برد. این مسئله باعث جدا شدن بلوکها از یکدیگر و خوانایی بهتر برنامه می شود.

همانطور که در خروجی مثال بالا می بینید که نماد \n که در آنهای رشته قرار گرفته است در خروجی نمایش داده نشده است. وقتی (\) Backslash در انتهای یک رشته قرار گیرد و کاراکتری که بعد آن قرار بگیرد یک کاراکتر خاص است. به عنوان مثال \n باعث می شود که کرسر[۶۶](موقعیت جدید نمایشگر) به خط جدید برود. به خروجی برنامه زیر دقت کنید

```
#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++!
```

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

[66] cursor (i.e., the current screen-position indicator)

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

۳,۴-مفاهیم اولیه زبان C++

شناسه **ها**[67]

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

نکته مهم دیگری که باید به آن اشاره کرد آنست که زبان C++ برخلاف بسیاری از زبانهای دیگر به کوچک و بزرگی حروف حساس است .(case sensitive) در نتیجه شناسه های زیر با یکدیگر متفاوتند:

Number1 ≠ number1

[67] Identifier

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.

زبان C++ دارای کلمات کلیدی زیادی است که در زیر برخی از آنها را نشان می دهیم :

auto	break	case	char	const
continue	default	do	double	else
enum	extern	float	for	goto
if	int	long	register	return
short	signed	sizeof	static	struct
switch	typedef	union	unsigned	void
volatile	while	and	and_eq asm	bitand bitor
bool	catch	class	compl	const_cast
delete	dynamic_cast	explicit	export	false
friend	inline	mutable	namespace	new
not	not_eq	operator	or	or_eq
private	protected	public	reinterpret_cast	static_cast
template	this	throw	true	try
typeid	typename	using	virtual	wchar_t
xor	xor_eq			

۳,۵C++-داده ها در

همانطور که در بخش های قبل بیان کردیم؛ یک متغییر[۶۸] 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[۶۹] گردد. اعلان یک متغیر بدین معنی است که نوع آن متغییر را مشخص شود. در زیر برخی نوع های مورد استفاده در C++ را نشان می دهیم و در انتهای این بخش لیست کاملی از متغییر های C++ ذکر شده است.

[68] Variable
[69] declarations

نوع داده	توضیح	اندازه(بیت)	دامنه
char	کاراکتر	۸	۱۲۷تا ۱۲۸-
int	عدد صحیح	۱۶	32768-تا ۳۲۷۶۷
float	عدد اعشاری	۳۲	3.4e-38تا 3.4e+38
double	عدد اعشاری با دقت مضاعف	۶۴	1.7e-308تا1.7e308

۳,۶-تعریف متغییرها

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 ;
```

نکته مهم آنکه زبان C++ به متغیرها مقدار اولیه نمی دهد (حتی 0) و برنامه نویس خود باید اینکار را صریحا انجام دهد، درغیر اینصورت مقدار اولیه متغیر، نامعین خواهد بود. تعریف متغیرها طبق اصول زبان C++ میتواند درهرجایی از برنامه صورت پذیرد.

۳,۷-عملگرها

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.

۳,۸-عملگر محاسباتی:

در جدول زیر عملگرهای محاسباتی نشان داده شده است.

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

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.

بطورکلی هرگاه ثابتها و متغیرهایی از انواع مختلف در یک عبارت محاسباتی داشته باشیم، کامپایلر C++ همه آنها را به یک نوع یکسان که همان بزرگترین عملوند موجود است تبدیل خواهد کرد. در نوشتن عبارات محاسباتی در C++ همانند عبارات محاسباتی ریاضی می توان از پرانتز استفاده کرد.

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.

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		

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.

در زبان C++ برای انتساب چندین عملگر وجود دارد. ساده ترین عملگر انتساب، همان عملگر = است که در بسیاری از زبانها استفاده می شود.به عنوان مثال

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

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;
```

مقدار هر ۳ متغیر برابر ۱۰ خواهد شد.

در مثال زیر ترتیب عملگرها نشان داده شده است :

Z = p * r % (q + w /(x – y))

6 4 5 3 2 1

Result = a + b * (f - (g + b) / d) – c * (a – d) / e

10 8 4 3 1 2 9 7 5 6

زبان C++ دارای عملگر های انتساب دیگری هستند که به دو دسته یکانی و دودویی تقسیم می شوند. این عملگرها باعث می شوند در بعضی موارد بتوانیم عبارات کوتاهتری را بنویسیم.

در جدول زیرعملگرهای انتساب یکانی و نحوی عملکرد آنها را نشان داده شده است.

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

کن و سپس آن را افزایش بده

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	--

برای در بهتر عملگرهای انتساب یکانی به خروجی حاصل از برنامه زیر دقت کنید:

```
#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
```

در جدول زیر عملگرهای انتساب دودویی و نحوی عملکرد آنها نشان داده شده است. اولویت تمامی این عملگرها راست به چپ است.

int a=12;

عملگر	مثال	توسعه یافته عملگر	مقدار متغییر پس از اعمال عملگر

+=	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

۳,۹-عملگرهای مقایسه ای

این عملگرها دو عبارت را بایکدیگر مقایسه کرده و نتیجه را باز می گردانند. نتیجه می تواند درست (true) یا غلط (false) باشد.نوع داده دیگری که در C++ موجود است نوع داده بولین (bool) است که مقدار آن یا درست است یا نادرست . بنابراین نتیجه عملگرهای مقایسه ای یک مقدار bool است. در جدول زیر نمونه هایی از عملگرهای مقایسه ای نشان داده شده است.

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

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 =!

لازم بذکر است که در هنگام مساوی بودن اولویت چند عملگر، این عملگرها از چپ به راست محاسبه می گردند.

۳.10. عملگرهای منطقی

این عملگرها این اجازه را می دهند که شرطهای ساده ای را که با استفاده از عملگرهای مقایسه ای ایجاد شده اند را با یکدیگر ترکیب نموده و شرطهای پیچیده تری را ساخت. این عملگرها عبارتند از :

عملگر	مفهوم عملگر	نحوه کار
&&	منطقیAND	اگر هر دو عملوند درست باشند, درست و در غیر اینصورت نادرست باز می گرداند.
||	منطقیOR	اگر هر دو عملوند نادرست باشند, نادرست و در غیر اینصورت درست باز می گرداند.
!	منطقیNOT	اگر عملوند درست باشد، نادرست و اگر نادرست باشد، درست برمی گرداند.

همانطور که قبلا نیز گفته شد مقدار بازگشتی این عملگرها، درست (1) یا نادرست (0) می باشد. اولویت این عملگرها بترتیب عبارتست از :

۱- عملگر ! ۲- عملگر && ۳- عملگر ||

۳,۱۱-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

نحوه کار بدینصورت است که درصورت درست بودن شرط عبارت۱ و در غیراینصورت عبارت ۲ بازگردانده می شود. مثال :

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

که این عبارت معادل دستور زیر است:

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

۴,۱۱-دریافت داده از کاربر و نمایش اطلاعات به کاربر

همانطور که در اولین برنامه این فصل دیدیم با استفاده از cout<< می توان اطلاعات را به کاربر نشان داد. برای دریافت داده از cin>> استفاده می شود. برای اینکه در Visual c++ بتوان از این دو استفاده کرد باید using namespace std; را در ابتدای برنامه قرار داد. در زیر برنامه ای نمایش داده شده است که از کاربر دو عدد می گیرد و آنها را باهم مقایسه می کند.

```
// 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";

  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
```

خروجی حاصل از اجرای برنامه در زیر نمایش داده شده است

```
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
```

۳,۱۲-عملگرهای بیشتر

-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);
```

دستور فوق مقدار ۱۰۰ رابه متغییر value نسبت میدهد.

مثال:پس از اجرای دستور

```
a=(b=3,b+2);
```

مقدار a=5 خواهد بود.

-عملگر sizeof

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

```
sizeof متغییر ;
```

sizeof(نوع داده) ;

۳,۱۳عدد نویسی در C++

C++ اجازه میدهذ به جای سیستم عدد نویسی در مبنای ۱۰ از سیستم های عددی در مبنای ۸(octal) یا ۱۶(hexa) استفا ده نماییم.برای مشخص کردن عددی در مبنای هگزا دسیمال باید عددرا با 0x و در مبنای اکتال با 0 شروع نمود.

به عنوان مثال

X=0356; //عددرمبنای ۸

Y=0xAB; //عدد درمبنای ۱۶

همچنین با استفاده از یک هدایت کننده میتوان یک عدد را در مبنای ۸ یا 16 را از ورودی خوانده و یا در خروجی چاپ نمود.این هدایت کننده در فایل سرآیند ی به نام iomanip قراردارد.

مثال:

```
#include<iostream.h>
#include<iomanip.h>
#include<conio.h>
main(){
  cin>>hex>>a;
   cout<<"a="<<a<<'\n';
    cout<<hex<<100<<'\n';
     cout<<oct<<20<<'\n';

    }
```

ABC3

64

24

۳,۱۴عملگرهای 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.

X=0x7ca4;

X=0111,1100,1010,0100

Y=x<<4;

Y=1100,1010,0100,0000

در نتیجه خواهیم داشت:

Y=0xca40;

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.

نکته:شیفت به چپ معادل ضرب در دو ،و شیفت به راست معادل تقسیم بر ۲ میباشد.

۳,۱۵-سایر عملگرهای بیتی

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	^

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.

یک کاربرد از عملگرهای بیتی میتواند به شکل زیر باشد:

فرض کنید بخواهیم اطلاعاتی رااز پورت موازی کامپیوتر بخوانیم با عملگرهای بیتی میتوان اطلاعات خوانده شده راتفسیر نمود.مثلا اگر بیت سوم عدد خوانده شده نشاندهنده وجودنداشتن کاغذ در چاپگر باشد، وبخواهیم تنها بیت سوم را چک کنیم میتوانیم به صورت زیر عمل کنیم:

```
if(n&4!=0) cout<<"paper out\n";
```

۳,۱۶-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.

میتوان برای توصیف کننده ها از کلمه کلیدی int نیز استفاده نمود.

```
unsigned int m;
```

مثال:

```
#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;

}
```

خروجی به شکل زیر خواهد بود:

```
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
```

جدول زیر طول نوع داده های مختلف به همراه توصیف کننده های آنها را نشان میدهد

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)		

۳,۱۷.قالب بندی نوع داده

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

(نوع داده) متغییر ;

مثال:

```
float f;
f=100.2;
cout<<(int) f;//چاپ100
int i=10;
cout<<i/3<<'\n';
cout<<(float) i/3<<'\n';
```

۳,۱۸-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

مثال

```
#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;


}
```

در این صورت خروجی به شکل زیر خواهد بود:

```
3.1416

| 3.1416|   3.1416|

|--3.1416|----3.1416|

3.1416e+000

0

false
```

۳,۱۹-تبدیل نوع داده در عبارتها

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.

مثال:

۳,۲۰-تبدیل نوع داده در دستورات انتساب

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 .

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

   }
```

نکته:برای تبدیل نوع داده از عدد صحیح به کاراکتری ،یا عدد صحیح بزرگتر به عدد صحیح کوچکتر قاعده اصلی این است که به تعداد لازم بیت های با مرتبه بالاتر کاسته خواهدشد.

مثال: خروجی قطعه کد زیر برابر ۲۴- میباشد:

```
#include <iostream.h>

main(){
```

char ch;

i**nt** i; i=1000;

ch=i;

i=ch;

cout<<i;}

دلیل اینکه مقدار ۲۴- نمایش داده میشود این است که فقط هشت بیت با مرتبه پایین i به ch کپی میشود سپس این هشت بیت مجددا به i نسبت داده شده وهشت بیت با مرتبه بالاتر آن را صفر میکند.

۳,۲۱-سریز وپاریز اعداد صحیح

اعداد صحیح موجود در کامپیوتر محدود میباشند.بدین معنی که دارای بیشترین و کمترین مقدار میباشند.در صورت رعایت نکردن این کرانها خطای سرریز(برای کران بالا)و پاریز(برای کران پایین) اتفاق می افتد.

در مواقع بروز خطای سرریز یا پاریز اجرای برنامه متوقف نمیشود (خطای گرامری) نداریم اما اعدادی که تولید میشوند بر اساس قاعده خاصی بررسی میشوند:

شیوه رفتار با سرریز بدین گونه است که مقادیر به صورت چرخشی در نظر گرفته میشوند ،بنابراین عددی که پس از

بیشترین مقدار می آید به عنوان کمترین عدد مورد استفاده قرار میگیرد.

نحوه رفتار با پارریز نیز به همین صورت (چرخشی) است فقط جای بیشترین وکمترین با هم عوض میشود.

مثال:

```
#include <iostream>

main() {
   // Range of int is [-2147483648, 2147483647]
   int i1 = 2147483647;       // max int
   cout << i1 + 1 << endl;    // -2147483648 (سرریز)
   cout << i1 + 2 << endl;    // -2147483647
```

```
    cout << i1 * i1 << endl;   // 1

    int i2 = -2147483648;      // min int
    cout << i2 - 1 << endl;    // 2147483647 (پاریز)
    cout << i2 - 2 << endl;    // 2147483646
    cout << i2 * i2 << endl;   // 0

}
```

۳,۲۲-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.

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.

max=103U;

تمرینها

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";
```

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;
```

۶-خروجی را بیابید:

```
int a=2;
int b=4;
int c=5;
```

```
d=a++;
d=a+b;
d=++c;
d=b+c;
cout<<d;
```

۷-برنامه ای بنویسید که درجه هوا برحسب فارنهایت را بگیرد ومقدار دما برحسب سانتی گراد را ازرابطه زیر محاسبه وچاپ نماید:

۸- خروجی عبارت زیر را به ازای a=5 و a=3 بدست آورید:

```
B = (a<5) ? (a*5) : (a*4);
```

بخش چهارم: ساختارهای کنترلی

۴,۱-مقدمه

در حالت عادی، اجرای یک برنامه به همان ترتیبی که نوشته شده اند یکی پس از دیگری اجرا می شوند که به این کار اجرای ترتیبی[۷۰] گفته می شود. در C++ این امکان به برنامه نویس داده می شود که ترتیب اجرای برنامه را با استفاده از ساختارهای کنترلی در دست بگیرند. در برنامه نویسی ساختیافته، هر برنامه از ۳ ساختار کنترلی بنام: ساختار ترتیب، ساختار انتخاب و ساختار تکرار تشکیل می گردد. از آنجا که این ۳ ساختار، نحوه و ترتیب اجرای برنامه را کنترل می کنند، به آنها ساختارهای کنترلی گفته می شود. ساختار ترتیب همان اجرای ترتیبی برنامه پشت سر هم است که در مثال های بخش قبل با نحوی عملکرد آن آشنا شدیم. زبان C++ دارای ۳ نوع ساختار انتخاب می باشد که عبا رتند از : ساختار if یا ساختار تک انتخابی، ساختار if / else یا ساختار دو انتخابی و ساختار switch یا ساختار چند انتخابی. علاوه براین، این زبان دارای ۳ نوع ساختار تکرار بنامهای while، for و do / while نیز می باشد.

۴,۲-ساختارهای انتخاب

همانطور که در مقدمه بیان کردیم C++ از سه نوع ساختار انتخاب استفاده می کند:

۱. ساختار انتخاب if که عملی را در صورت درست بودن یک شرط انجام می دهد و در صورت نادرست بودن شرط از روی آن می پرد.

[70] Sequential execution

۲. ساختار انتخاب if/else عملی را در صورت درست بودن شرط انجام می دهد و در صورت نادرست بودن شرط عمل دیگری را انجام می دهد.
۳. ساختار انتخاب switch که با توجه به مقدار یک عبارت، یکی از چند عمل مختلف را انجام می دهد.

در زیر به بررسی دقیق هر یک از ساختارهای انتخاب می پردازیم:

۴,۳if-ساختار دستور انتخاب

شکل کلی ساختار دستور if به صورت زیر است:

```
if( شرط مورد نظر )

        دستور مورد نظر ;
```

اگر شرط مورد نظر جلوی دستور if که داخل پرانتز نوشته شده است درست باشد دستور مورد نظر اجرا میشود و در غیر اینصورت اجرا نمی شود. شرط مورد استفاده در دستور if هر عبارت منطقی می تواند باشد.

دقت کنید که پرانتز استفاده شده پس از دستور if اجباری است.

شبه کد زیر نشان می دهد که اگر نمره درس دانشجویی بزرگتر از ۶۰ باشد در صفحه نمایش عبارت "Passsed"، چاپ شود.

```
if( grade >= 60)

        cout << "Passsed";
```

در حالت عادی دستور if منتظر یک دستور در بدنه خود می باشد، اما چنانچه می خواهیم چندین دستور را در بدنه یک دستور if دهیم، باید آنها را در داخل آکولاد باز وبسته { } قرار دهید. این مجموع دستورات را یک دستور مرکب می گویند. بطور کلی در زبان C++ هرجا که می توان یک دستور قرار داد، می توان از یک دستور مرکب نیز استفاده کرد. به یک دستور مرکب، بلوک نیز گفته می شود.

مثال قبل را اینگونه در نظر بگیرید که اگر نمره دانشجویی بزرگتر از ۶۰ باشد در صفحه نمایش عبارت "Passsed"،را چاپ کند و در خط بعد در صفحه نمایش چاپ کند که "You can chose next course!"

```
if( grade >= 60)
{
        cout << "Passsed\n";
        cout<<" you can chose next course!";

}
```

نکته: قرار ندادن یک یا هر دو آکلاد دستور مرکب می تواند منجر به بروز خطای نحوی و منطقی شود.

به نحوه دندانه گذاری در شبه کد دقت کنید، هرجا که بلوک جدیدی ایجاد شده است، دستورات آن حدود ۳ کاراکتر جلوتر (به اندازه ی کلید Tab)نوشته شده اند. اینکار باعث می شود که خوانایی شبه کد افزایش یابد. این کار را برای افزایش خوانایی برنامه ها نیز بهتر است بکار برد.

۴,۴if/else-ساختار دستور انتخاب

اگر بخواهیم در صورت درست نبودن شرط if دستور دیگری اجرا شود از دستور if-else که شکل کلی آن به صورت زیر میباشد استفاده میکنیم:

```
if(عبارت منطقی)
        دستور۱;
else
        دستور۲;
```

مثال قبل را اینگونه در نظر بگیرید که اگر نمره دانشجویی بزرگتر از ۶۰ باشد در صفحه نمایش عبارت قبول شدن،را چاپ کند و در خط بعد در صفحه نمایش چاپ کند که "You can chose next course!" و اگر نمره دانشجو از مقدار ۶۰ کمتر بود در صفحه نمایش عبارت "Failed" را چاپ کند و در سطر بعد عبارت "You must take this course again" را چاپ کند.

```
if( grade >= 60)
{
        cout << "Passsed\n";
        cout<<" you can chose next course!";

}
else
{
        cout<< "Failed\n";
        cout<< "You must take this course again";

}
```

یک روش متداول استفاده از دستور if، استفاده از if های تودرتو می باشد که با استفاده از یک مثال عملکرد آن را توضیح می دهیم:

در شبه کد زیر برای نمره بزرگتر از ۹۰ حرف A را چاپ کند برای نمره بین محدوده ۸۰ تا ۸۹ حرف B را چاپ کند برای نمره بین محدوده ۷۰ تا ۷۹ حرف C را چاپ کند برای نمره بین محدوده ۶۰ تا ۶۹ حرف D را چاپ کند و برای نمرات کمتر از ۶۰ حرف E را چاپ کند.

```
if( grade >= 90)          cout<< " A";
else      if( grade >= 80)        cout<<" B";
          else      if(grade >= 70)         cout<<"C";
                    else      if(grade>=60)           cout<<"D";
                              else      cout<<"E";
```

در چنین دستوری، کلیه شرطها بترتیب از بالا به پایین بررسی شده و به محض اینکه یکی از آنها درست باشد، دستور مربوط به آن اجرا شده و از بقیه دستورات صرفنظر می گردد. درصورتیکه هیچ یک از شرطها درست نباشد، دستور مربوط به آخرین else اجرا می گردد. درچنین حالتی توصیه می گردد که شرطهای نادر را که امکان وقوع آنها کم است، در انتهای کار بررسی نمایید، تا تعداد مقایسه کمتری صورت پذیرد.

مثال ۴,۱: برنامه ای به زبان C++ بنویسید که ضرایب معادله درجه ۲ را دریافت کند و ابتدا چک کند که معادله از نوع درجه ۲ است و پس از آن ریشه های معادله را در صورت وجود چاپ کند در غیر اینصورت چاپ کن معادله ریشه ندارد.

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

void main()
{
     int a,b,c;
     double delta,x1,x2;
     cout<<"please Enter a,b,c :";
     cin>>a>>b>>c;

      if(a==0)
             cout<<"It isn't a correct equation!\n";
     else
     {
         delta=b*b-4*a*c;
         if(delta<0)
                    cout<<" no answer";
         else
```

```
        {
            if(delta==0)
            {
                x1=-b/(2*a);
                cout<<"equation has one answer, then x1 is :";
                cout<<x1;

            }
            else
            {
                delta=sqrt(delta);

                x1=(-b+delta)/4*a;
                x2=(-b-delta)/4*a;
                cout<<"x1 is :"<<x1<<"\n";
                cout<<"x2 is :"<<x2<<"\n";

            }

        }


    }

}
```

اولین نکته در رابطه با این برنامه استفاده از فایل math.h است. این فایل حاوی توابع کتابخانه ای ریاضی هست و از آنجا که در این برنامه از تابع جذر sqrt که یک تابع ریاضی تک آرگومانی است باید این فایل را در برنامه گنجاند. در جدول های زیر برخی از توابع ریاضی و مثلثاتی واقع در math.h را نشان داده شده است.

شکل تابع	عملکرد
ceil(x)	سقف عدد
exp(x)	عدد eبه توان x
fabs(x)	قدرمطلق x
pow(x,y)	Xبه توان y

شکل تابع	عملکرد
acos(x)	کسینوس وارون
asin(x)	سینوس وارون
atan(x)	تانژانت وارون
cos(x)	کسینوس
sin(x)	سینوس
tan(x)	تانژانت

شکل تابع	عملکرد
log(x)	لگاریتم x
sinh(x)	سینوس هذلولوی
sqrt(x)	جذر عددx
floor(x)	قسمت اعشارx
cosh(x)	کسینوس هذلولوی

۴,۵switch-ساختار چند انتخابی

گاهی اوقات الگوریتمی ممکن است دارای تعدادی تصمیم گیری باشد که در آنها متغییر یا عبارتی با مقادیر صحیح ثابتی که این متغییر یا عبارت می تواند به خود بگیرد به طور جدا مقایسه شود و برای هر مقدار اعمال مختلفی انجام شود. هر چند این کار را می توان با if/else های تودرتو نیز انجام داد اما ساختار مناسبتری نیز برای اینکار وجود دارد، که به آن ساختار چندانتخابی می گوییم.

که دارای ساختار کلی زیر هستند:

```
switch  (<expression>) {
  case <exp1> :
                <statement 1> ;
                 <statement 2> ;
                  …
                 <statement n>;
  case <exp2> :
                <statement 1> ;
                 <statement 2> ;
                  …
                 <statement n>;

  …
  default :
                  <statement 1> ;
                 <statement 2> ;
                  …
                 <statement n>;
```

}

مثال ۴,۲: برنامه ای به زبان C++ بنویسید که دو عدد به همراه یک عملگر از کاربر دریافت کند و حاصل عبارت را در خروجی چاپ کند.

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

void main()
{
      int a;
      int a1,a2;
      double result;
      char operand;
      char ch;
      cout<<"Enter the numbers: ";
      cin>>a1>>a2;
      cout<<"Enter operand: ";
      operand=getch();
      putch(operand);
      cout<<"\n";

      switch(operand)
      {
      case'+':
                  result=a1+a2;
                  cout<<"a1+a2= "<<result;
                  break;
      case'-':
                  result=a1-a2;
                  cout<<"a1-a2= "<<result;
                  break;
      case'*':
                  result=a1*a2;
                  cout<<"a1*a2= "<<result;
      case'/':
            if(a2==0)
            {
                  cout<<" it's not possible";
                  break;

            }
            else
            {
                  result=a1/a2;
                  cout<<"a1/a2= "<<result;
                  break;
```

```
        }
    default:
        cout<<"invalid operator!!!!!!!!!";

    }
    getch();
}
```

همانطور که در برنامه بالا مشاهده می کنید برای دریافت کاراکتر از کاربر از دستور getch() و برای نمایش کاراکتر به کاربر از دستور putch() استفاده کردیم که برای استفاده از این دستورات باید `#include<conio.h>` را در سرایند قرار داد.

اگر دستور break در یک ساختار switch بکار رود، باعث می شود که بلافاصله کنترل اجرای برنامه از ساختار خارج شده و به اولین دستور پس از ساختار برود.

۴,۶-ساختارهای تکرار

ساختارهای تکرار باعث می شود تا زمانیکه شرط خاصی برقرار است، عملیات مشخصی تکرار گردد.

۴,۷While-ساختار تکرار

دستور while نیز باعث ایجاد یک حلقه تکرار به شکل زیر می گردد:

while (شرط) دستور;

این دستور باعث می شود تا زمانیکه شرط موجود درست است، دستور تکرار شود، و به محض اینکه شرط نادرست گردد، کنترل اجرا به دستور بعد از حلقه می رود.

ممکن است ساختار کنترلی while شامل دستورات مرکب باشد که آنها را طبق قانونی که در پیش اشاره شد داخل آکولادهای باز و بسته قرار می گیرد و بصورت زیر تعریف می شود.

```
While (شرط)
{
    دستور۱;
    دستور ۲;
    ...

    دستور آخر;
}
```

مثال۴,۳: برنامه ای به زبان C++ بنویسید که عدد صحیح مثبت n را از کاربر دریافت کند و اعداد فیبوناچی آن را در خروجی چاپ کند:

1,1,2,3,5,8,13,21,...

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

void main()
{
	int n;
	cout<<"enter n :";
	cin>>n;

	cout<<"\n";

	int f1=1;
	int f2=1;
	int f3;
	cout<<f1<<" , "<<f2;
	int counter=2;

	while(counter <= n)
	{
		f3 = f2 + f1 ;
		cout<<" , "<<f3;
		f1 = f2;
		f2 = f3;
		counter++;
	}

}
```

مثال ۴,۴: برنامه ای به زبان C++ بنویسید که تا نمرات تعدادی دانشجو را از کاربر دریافت کند و میانگین نمرات را در خروجی نمایش دهد. بطوریکه کاربر با وارد کردن عدد -1 وارد کردن نمرات را به اتمام می رساند.

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

void main()
{
	float grade;
	float sum=0;
	float average;
	char finish='0';
	int counter=0;
	cout << " enter number : ";
	cin >> grade;

	while(grade != -1)
```

```
    {
        sum = sum + grade ;
        counter++;
        cout << " enter number : ";
        cin >> grade;

    }
    average = sum / counter;

    cout<<"average is : "<<average;

}
```

۴,۸for-حلقه های تکرار

گاهی نیاز به حلقه تکراری داریم که به تعداد دفعات مشخصی تکرار گردد. در چنین مواقعی با استفاده از یک متغیر شمارنده، تعداد تکرارها را تا رسیدن به مقدار مورد نظر می شماریم و سپس به حلقه پایان می دهیم. به چنین حلقه هایی، تکرار تحت کنترل شمارنده یا تکرار معین می گوییم، چرا که تعداد تکرار ها از قبل مشخص است. چنین حلقه ای دارای ۳ جزء اصلی می باشد:

۱. مقداردهی اولیه به متغیر شمارنده حلقه
۲. شرط پایان حلقه (پایان شمارش)
۳. نحوه تغییر شمارنده

ساختار کلی یک حلقه ی تکرار for بصورت زیر می باشد:

```
for(مقدار اولیه شمارنده ; شرط پایان حلقه ; گام تغییر شمارنده)
        دستور;
```

همچون سایر ساختارهایی که تاکنون بررسی کردیم اگر حلقه تکرار for قرار باشد دستوراتی را دنبال کند دستورات را در { و } قرار می دهیم. بصورت :

```
for(مقدار اولیه شمارنده ; شرط پایان حلقه ; گام تغییر شمارنده)
{
        دستور1;
        دستور۲;
        ...
        دستور آخر;
}
```

مثال۴,۵: برنامه ای بنویسید که عدد n را دریافت کند و فاکتوریل آن را حساب کند.

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

```
using namespace std;

void main()
{
	int n;
	long int factorial=1;
	cout<<"enter n: ";
	cin>>n;
	for(int i=1; i<=n ; i++)
	{
		factorial=factorial*i;

	}
	cout<<n<<"!= "<<factorial;
	getch();
}
```

مثال۴,۶: برنامه ای بنویسید که اعداد زوج بین ۱ تا ۱۰۰ را به صورت نزولی نمایش دهد.

```
#include<iostream>
#include <conio.h>
using namespace std;
void main()
{

	for(int i=100; i>0 ; i=i-2)
	{
		cout<<i<<" ";

	}

	getch();
}
```

قسمت شرط می تواند یک شرط مرکب نیز باشد، بعنوان مثال:

```
for  (count = 0; count < 100 && sw==1; count++)
```

که در اینصورت در هربار اجرای حلقه، علاوه بر مقدار شمارنده، مقدار متغیر sw نیز بررسی می گردد.

نکته آخر اینکه قسمت مقدار دهی اولیه و افزایش متغیر نیز می توانند شامل چند عبارت باشند که در اینصورت با کاما از یکدیگر جدا می شوند. بعنوان مثال:

```
for  (a = 0, b = 100; b - a > 50;  a++, b--)
```

۴,۹for-حلقه های تو در تو

در حلقه های for تو در تو به ازای هر بار اجرای for بیرونی حلقه یfor بیرونی یکبار بطور کامل اجرا می شود. و ساختار حلقه ی for تو در تو به ازای ۲ حلقه بصورت زیر است:

```
for(  1مقدار اولیه شمارنده ; 1شرط پایان حلقه ; 1گام تغییر شمارنده)
        for(2مقدار اولیه شمارنده ;2 شرط پایان حلقه ; 2گام تغییر شمارنده)
{
        ;دستور۱
        ...
        ;دستور آخر
}
```

شبه کد زیر را در نظر بگیرید

```
for(int i=1; i<5; i++)
        for(int j=1;j<=i ; j++)
                cout<<"* ";
        cout<< "\n";
```

جدول زیر خروجی را به ازای هر بار تکرار حلقه ی بیرونی نمایش می دهد:

مقدار i	مقادیر j	خروجی
۱	1	*
۲	1,2	* *
3	1,2,3	* * *
4	1,2,3	* * * *

برنامه ۷: برنامه ای بنویسید که جدول ضرب تولید کند:

```
#include<iostream>
#include <conio.h>
using namespace std;
void main()
{

      for(int i=1; i<=10; i++)
      {
```

```
            for(int j=1; j<=10; j++)
                  cout<<i<<"*"<<j<<"=" << i*j << " ";
            cout<<endl;
      }

      getch();
}
```

۴,۱۰-حلقه های do/while

در بعضی مواقع لازم است که شرط ، در انتهای حلقه بررسی گردد. دستور do / while از نوع حلقه هایی است که ابتدا دستورات را اجرا کرده و سپس شرط ادامه حلقه را بررسی می نماید. شکل کلی این دستور بصورت زیر است :

```
do

    دستور;

While (شرط);
```

نحوه کار این حلقه به این صورت است که ابتدا دستور اجرا می گردد، سپس شرط حلقه بررسی شده و درصورتیکه درست بود، به ابتدای حلقه باز گشته و آن را مجددا اجرا می کند. البته دستور می تواند یک دستور مرکب باشد.

برنامه ۸: برنامه ای بنویسید که از کاربر بخواهد سن افراد یک خانواده را وارد کند و در صورتی که کاربر سن را عددی منفی وارد کرد از او بخواهد سن را درست وارد کند و در انتها میانگین سنی افراد آن خانواده را چاپ نماید.

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

void main()
{
      int age;
      int n;
      cout<<"enter the number of family: ";
      cin>>n;
      int sum=0;
      float ave;
      char ch;
      for(int i=1;i<=n;i++)
      {
            do{
            cout<<"please enter age:";
            cin>>age;
            }while(age<0);
```

```
        sum=sum+age;
    }
    ave=sum/n;
    cout<<"\n the average is:"<<ave;


    getch();
}
```

continue و break-دستورات۴,۱۱

این دستورات قادرند مسیر اجرای برنامه را در یک حلقه تکرار تغییر دهند. چنانچه دستور break در یک ساختار while، for، do/while ویا switch بکار رود، باعث می شود که بلافاصله کنترل اجرای برنامه از ساختار خارج شده و به اولین دستور پس از ساختار برود.

اما دستور continue فقط در حلقه های while، for و do/while بکار می رود. نحوه عمل آن بدین صورت است که به محض آنکه کنترل اجرا به این دستور برسد، بلافاصله از باقیمانده حلقه صرفنظر کرده و مجددا به ابتدای حلقه باز می گردد و اجرای آن را از سر می گیرد. در مورد حلقه for، پس از بازگشت به ابتدای حلقه، عمل افزایش مقدار متغیر حلقه نیز صورت می پذیرد.

۴-۱۲.اعداد تصادفی

یکی از مهمترین موارد استفاده از کامپیوتر،شبیه سازی سیستمهای واقعی است.شبیه سازی نیاز به تولید کامپیوتری اعداد تصادفی دارد .

فایل سرآیند <stdlib.h> تابع ()rand را تعریف میکند. این تابع ،مولد اعدا د صحیح شبه تصادفی در بازه 0 تا RAND_MAX میباشد.RAND_MAX ثابتی است که در آن نیز در<stdlib.h> تعریف شده است.RAND_MAX ثابتی است که آن نیز در

تعریف شده است.هربار که تابع ()rand فراخوانده میشودعدد صحیح unsigned دیگری در این بازه تولید می کند.

مثال:

قطعه کد زیر را در نظر بگیرید:

```
#include<iostream.h>

#include<conio.h>

#include <stdlib.h>
```

```
main(){
int i=1,x;
while(i<5){
x=rand();
cout<<x<<'\n';
i++;
}
getch();
}
```

```
41
18467
6334
26500
```

مشکلی که در اینجا با آن مواجه ایم این است که اگر این برنامه را چندین بار تکرار کنیم ،همین اعدا د حاصل میشوند که این با تصادفی بودن اعداد در تعارض است. برای رفع این مشکل از تابع srand استفاده میکنیم:

```
#include<iostream.h>
#include<conio.h>
#include <stdlib.h>
#include<time.h>
main(){
int seed;
seed=time(NULL);
```

```
srand(seed);
int i=1,x;
while(i<5){
x=rand();
cout<<x<<'\n';
i++;
}
}
```

مثال:

```
#include<iostream.h>
#include<conio.h>
#include <stdlib.h>
#include<time.h>
main(){
    int seed=time(0);
    srand(seed);
cout<<(rand()%6+1)<<'\n';//تولید یک عدد تصادفی بین ۱و ۶
cout<<(float)rand()/RAND_MAX<<'\n';//تولید یک عدد تصادفی اعشاری بین صفر ویک


}
```

خروجی به شکل زیر خواهد بود:

```
5
0.5091
```

تمرینها:

۱- برنامه ای با استفاده از حلقه های for تو در تو بنویسید که خروجی آن شکل زیر باشد.

```
# # # # # # # # # #
& & & & & & & & & &
# # # # # # # # # #
& & & & & & & & & &
# # # # # # # # # #
& & & & & & & & & &
```

۲-خروجی قطعه کد زیر چیست؟

```
int x=5;

if(++x<4) cout<<x++;

else cout< <--x;
```

۳-برنامه ای بنویسید که عددطبیعی را از ورودی گرفته در صورت زوج بودن دو برابر ودر غیر اینصورت سه برابر آن را چاپ کند

۴-کدام گزینه در مورد قطعه برنامه زیر اتفاق می افتد:

```
int c=5;
if(c=6) cout<<c;
```

۱)خطای کامپایلری ۲)به دلیل عدم برقراری شرط چیزی در خروجی چاپ نمیشود ۳)چاپ ۵

۴)چاپ ۶

۵-کد if((i==2)||flag)&&!flag) معادل کدام گزینه است؟

۱) if(i==2&&!flag)

2) if(i!=2||flag)

۳) if(i!=2||!flag)

۴) if(i==2||flag)

۶-خروجی را بیابید:

```
int x=2;
```

```
int y=1;
cout<<(x<y?-1:(x==y?0:1));
```

۷-خروجی را بیابید:

```
int a=10;
int b=6;
if(a-b)    b--;
if(a-2*b)  a++;
cout<<a<<b;
```

۸-خروجی قطعه کد زیر رابیابید

```
int j=0,i;
for(i=1;i<=23;i++)
if(i%3==0) j=j+i;
j=j+i;
cout<<j;
```

۹-خروجی را بیابید:

```
int i;
for(i=0; ;++i)
cout<<"c++ programming";
```

۱۰- خروجی را بیابید:

```
for(i=0,ch='a';i<4;i++,ch+=2*i)
cout<<ch;
```

۱۱- خروجی را بیابید:

```
for(m=10;m;m--);
cout<<"x";
```

۱۲- خروجی را بیابید:

```
for(i=0;++i<5;i++)
cout<<i++;
```

۱۳-خروجی را بیابید:

```
esp=1.0;
while(eps>0.0){
cout<<eps:
eps/=2;}
```

۱۴-خروجی را بیابید:

```
int u=0,t=0;
  while(u++<=5)
            t+=u++;
```

```
cout<<"t=\n"<<t<<"u=\n"<<u;
```

۱۵-برنامه ای بنویسید که یک عدد طبیعی را بگیرد و مقلوب آنرا چاپ کند

۱۶- برنامه ای بنویسید که مقدار زیر را محاسبه کند.

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

۱۷- خروجی را بیابید:

```
int number;
 while(--number>0)
     cout<< number++;
```

۱۸- خروجی را بیابید:

```
int a=5;int b=2;
while(a<80){
            b*=++a;
            cout<<b<<'\n';
            }
```

۱۹- برنامه ای بنویسید که یک عدد طبیعی رابگیرد وانرا به شکل حرف به حرف ترجمه نماید.مثلا برای n=124خروجی one two four خواهد بود.

۲۰- برنامه ای بنویسید که با حلقه های تکرار خروجی های زیر را تولید کند:

```
 # * # * # * # *      # # # # # # # # #      # # # # # # # # #      1
# * # * # * # *       # # # # # # # #          # # # # # # # #      2 1
 # * # * # * # *      # # # # # # #              # # # # # # #      3 2 1
# * # * # * # *       # # # # # #                  # # # # # #      4 3 2 1
 # * # * # * # *      # # # # #                      # # # # #      5 4 3 2 1
# * # * # * # *       # # # #                          # # # #      6 5 4 3 2 1
 # * # * # * # *      # # #                              # # #      7 6 5 4 3 2 1
# * # * # * # *       # #                                  # #      8 7 6 5 4 3 2 1
                      #                                      #
```

خروجی۴ خروجی ۳ خروجی ۲ خروجی ۱

۲۱- برنامه ای بنویسیدکه یک سری عدد را تا وقتی صعودی است از ورودی گرفته وحاصلجمع این اعداد را محاسبه نماید.(وارد شدن یک عددکمتر از عددقبل منجر به خاتمه برنامه میشود.)

مراجع:

How to program C++, 8th edition, Paul Deitel, Harvey Deitel

مرجع علمی - کاربردی برنامه‌نویسی به زبانC ، سعید ابریشمی - جهاد دانشگاهی مشهد

الگوریتم و فلوچارت، بهرام غلامی، علیرضا جباریه، موسسه فرهنگی هنری دیباگران تهران