برنامه سازی پیشرفته

فهرست مطالب

فصل اول

مقدمات C++

فهرست مطالب فصل اول

تاریخچه مختصر C++

این زبان در اوائل دهه ۱۹۸۰ توسط Bjarne stroustrup در آزمایشگاه بل طراحی شده. این زبان عملاً توسعه یافته زبان برنامه نویسی C می باشد که امکان نوشتن برنامه‌های ساخت یافته شئ گرا را می‌دهد.

قانون نامگذاری شناسه‌ها

۱) حروف کوچک و بزرگ در نامگذاری شناسه‌ها متفاوت می‌باشند.

بنابراین Xy ، XY ، xY ، xy چهار شناسه متفاوت از نظر C++ می‌باشد.

قانون نامگذاری شناسه‌ها

۲) در نامگذاری شناسه‌ها از حروف الفباء، ارقام وزیر خط (**underscore**) استفاده می‌شود و حداکثر طول شناسه ۳۱ می‌باشد و شناسه بایستی با یک رقم شروع نگردد.

قانون نامگذاری شناسه‌ها

۳) برای نامگذاری شناسه‌ها از کلمات کلیدی نبایستی استفاده نمود. در زیر بعضی از کلمات کلیدی داده شده است.

And	Sizeof	then	xor	Template
Float	False	Friend	While	continue
extern	Private	Switch	Default	Const
delete	typedef	if	this	Virtual

لیست کامل کلمات کلیدی

متغیرها

متغیر، مکانی در حافظه اصلی کامپیوتر می‌باشد که در آنجا یک مقدار را می‌توان ذخیره و در برنامه از آن استفاده نمود. قانون نامگذاری متغیرها همان قانون نامگذاری شناسه‌ها می‌باشد.

در اسلاید بعد به انواع داده ها اشاره می شود.

انواع داده ها

نوع داده	مقادیر	حافظه لازم
int	۳۲۷۶۷ تا ۳۲۷۶۸-	۲ بایت
unsigned int	۰ تا ۶۵۵۳۵	۲بایت
long int	۲۱۴۷۴۸۳۶۴۷ تا ۲۱۴۷۴۸۳۶۴۸-	۴ بایت
unsigned long int	۰ تا ۴۲۹۴۹۶۷۲۹۵	۴ بایت
char	یک کارکتر	۱بایت
unsigned char	۱۲۷ تا ۱۲۸-	۱ بایت
float	۳/۴e۳۸ تا ۱/۲e-۳۸	۴ بایت
double	۱/۸e۳۰۸ تا ۲/۲e-۳۰۸	۸ بایت

اعلان متغیرها

قبل از آنکه در برنامه به متغیرها مقـداری تخصـیص داده شود و از آنها استفاده گردد بایستی آنها را در برنامه اعلان نمود.

در اسلاید بعد مثال هایی از اعلان متغیر ذکر شده است.

چند مثال از اعلان متغیر ها :

- ✓ برای اعلان متغیرx از نوع int :

```
int   x;
```

- ✓ برای اعلان متغیرهای p و q را از نوع float که هر کدام چهار بایت از حافظه را اشغال می‌کنند :

```
float  p , q  ;
```

- ✓ برای اعلان متغیر next از نوع کرکتر که می‌توان یکی از ۲۵۶ کرکتر را به آن تخصیص داد و یک بایت را اشغال می‌کند.

```
char next ;
```

تخصیص مقادیر به متغیرها

با استفاده از عملگر = می‌توان به متغیرها مقدار اولیه تخصیص نمود.

در اسلاید بعد مثال هایی از اعلان متغیر ذکر شده است.

مثال :

- در دستورالعمل `int x=۲۶;`
  X را از نوع **int** با مقدار اولیه ۲۶ اعلان نموده .

- در دستورالعمل `long a=۶۷۰۰۰ , b=۲۶۰;`
  متغیرهای **b** و **a** را از نوع **long int** تعریف نموده با مقادیر بترتیب ۲۶۰ و ۶۷۰۰۰.

داده‌های از نوع کرکتر

برای نمایش داده‌های از نوع **char** در حافظه کامپیوتر از جدول **ASCII** استفاده می‌شود. جدول اسکی به هر یک از ۲۵۶ کرکتر یک عدد منحصر بفرد بین ۰ تا ۲۵۵ تخصیص می‌دهد.

کرکترهای مخصوص

کامپیلر C++ بعضی از کرکترهای مخصوص که در برنامه می‌توان از آنها برای فرمت بندی استفاده کرد را تشخیص می‌دهد. تعدادی از این کرکترهای مخصوص به همراه کاربرد آنها در اسلاید بعد آورده شده است .

کرکترهای مخصوص

\n	Newline
\t	Tab
\b	Backspace
\a	Beep sound
\”	Double quote
\’	Single quote
\۰	Null character
\?	Question mark
\\	Back slash

بعنوان مثال از کرکتر a\ می‌توان برای ایجاد صدای beep استفاده نمود.

```
char      x = '\a' ;
```

رشته‌ها

رشته یا string عبارتست از دنباله‌ای از کرکترها که بین " " قرار داده می‌شود. در حافظه کامپیوتر انتهای رشته‌ها بوسیله ۰\ ختم می‌گردد.

در اسلاید بعد به دو مثال دقت نمایید.

مثال ۱ :

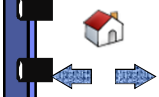
"BOOK STORE" یک رشته ده کرکتری می‌باشد که با توجه به کرکتر \۰ که به انتهای آن در حافظه اضافه می شود جمعاً یازده بایت را اشغال می‌کند.

مثال ۲ :

دقت نمایید که "w" یک رشته می‌باشد که دو بایت از حافظه را اشغال می‌کند در حالیکه 'w' یک کرکتر می‌باشد که یک بایت از حافظه را اشغال می‌نماید.

نمایش مقادیر داده‌ها

برای نمایش داده‌ها بر روی صفحه مانتور از cout که بـدنبال آن عملگـر درج یعنی << قید شده باشد استفاده می‌گردد. بایستی توجه داشت کـه دوکرکتر < پشت سر هم توسط C++ بصورت یک کرکتر تلقی می‌گردد.



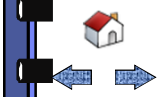
مثال :

- ✓ برای نمایش پیغام good morning بر روی صفحه نمایش :

```
cout << "good morning";
```

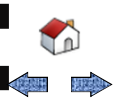
- ✓ برای نمایش مقدار متغیر X بر روی صفحه نمایش :

```
cout << x ;
```



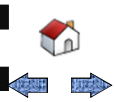
دریافت مقادیر متغیرها

به منظور دریافت مقادیر برای متغیرها در ضمن اجرای برنامه از صفحه کلید، از cin که بدنبال آن عملگر استخراج یعنی >> قید شده باشد می‌توان استفاده نمود.



مثال :

```
int x;
cout << "Enter a number:" ;
cin >> x;
```



عملگر انتساب

عملگر انتساب = می‌باشد که باعث می‌گردد مقدار عبارت در طرف راست این عملگر ارزیابی شده و در متغیر طرف چپ آن قرار گیرد.

مثال :

```
x=a+b;
x=۳۵ ;
x=y=z=۲۶ ;
```

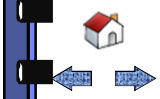
از عملگرهای انتساب چندگانه نیز می‌توان استفاده نمود. که مقدار سه متغیر z و y و x برابر با ۲۶ میشود.

عملگرهای محاسباتی

در **C++** پنج عملگر محاسباتی وجود دارد که عبارتند از :

جمع	+
تفریق	–
ضرب	*
تقسیم	/
باقیمانده	٪

این عملگرها دو تائی می‌باشند زیرا روی دو عملوند عمل می‌نمایند. از طرف دیگر عملگرهای + و – رامی‌توان بعنوان عملگرهای یکتائی نیز در نظر گرفت.



مثال ۱ :

در حالتی که هر دو عملوند عملگرهای ٪ ، / ، * ، + ، – از نوع صحیح باشد نتیجـه عمل از نوع صحیح می‌باشد.

عبارت	نتیجه
۵ + ۲	۷
۵ * ۲	۱۰
۵ – ۲	۳
۵ % ۲	۱
۵ / ۲	۲

مثال ۲ :

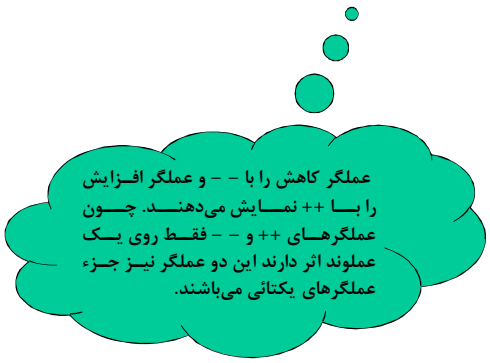
در صورتیکه حداقل یکی از عملوندهای عملگرهای / ، * ، – ، + از نوع اعشاری باشد نتیجه عمل از نوع اعشاری می‌باشد.

نتیجه	عبارت
۷. ۰	۵. ۰ +۲
۱۰/۰	۵* ۲. ۰
۲/۵	۵. ۰ / ۲
۳. ۰	۵. ۰ -۲
۲/۵	۵. ۰ / ۲. ۰

عملگرهای افزایش و کاهش

در C++ ، افزایش یک واحد به مقدار یک متغیر از نوع صحیح را افزایش و بطور مشابه کاهش یک واحد از مقدار یک متغیر از نوع صحیح را کاهش می‌نامند..

عملگرهای افزایش و کاهش



مثال :

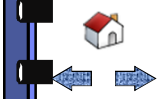
سه دستور العمل :

```
++x;
x++;
x=x+۱;
```

معادل می‌باشند و بطریق مشابه سه دستورالعمل زیر نیز معادل می باشند.

```
- - y ;
y=y-۱;
y- - ;
```

از عملگرهای ++ و -- می‌توان بدو صورت پیشوندی و پسوندی استفاده نمود.
در دستورالعمل‌های پیچیده عملگر پیشوندی قبل از انتساب ارزیابی میشود و عملگر پسوندی بعد از انتساب ارزیابی می‌شود.



مثال :

```
int x=۵;
y=++x * ۲;
```

پس از اجرای دستورالعملهای فوق :

```
y=۱۲
```

```
int  x=۵;
y=x++ * ۲;
```

پس از اجرای دستورالعملهای فوق :

```
y=۱۲
```

عملگر sizeof

Sizeof ازعملگرهای یکتائی می باشد و مشخص کننده تعداد بایت هائی است که یک نوع داده اشغال می‌کند.

مثال :

```
int   x;
cout << sizeof x ;
```

مقدار ۲ نمایش داده می‌شود .

```
cout << sizeof(float) ;
```

مقدار ۴ نمایش داده می شود.

عملگرهای جایگزینی محاسباتی

برای ساده‌تر نوشتن عبارتها در **C++** ، می‌توان از عملگرهای جایگزینی محاسبـــــاتی استفاده نمود.

+=	-=	*=	/=	%=

اولویت عملگرها

ارزیابی مقدار یک عبارت ریاضی براساس جدول اولویت عملگرها انجام می‌گردد. در ذیل جدول اولویت عملگرها براساس بترتیب از بیشترین اولویت به کمترین اولویت داده شده است.

چپ به راست	پرانتزها	( )
راست به چپ	عملگرهای یکتایی	sizeof ++ -- + -
چپ به راست	عملگرهای ضرب و تقسیم و باقیمانده	% / *
چپ به راست	عملگرهای جمع و تفریق	+ -
چپ به راست	عملگرهای درج و استخراج	>> <<
راست به چپ	عملگرهای جایگزینی و انتساب	= += -= *= /= %=

مثال ۱ :

(۵+۲) *(۶+۲**۲)/۲

با توجه به جدول اولویت عملگرها داریم که

۷ *(۶+۲**۲)/۲
۷*(۶+۴)/۲
۷* ۱۰ /۲
۷۰ /۲
۳۵

مثال ۲ :

```
int a=۶ , b=۲, c=۸, d=۱۲;
d=a++ * b/c ++;
cout << d << c << b << a;
```

خروجی :

```
۱ ۹ ۲ ۷
```

توضیحات (Comments)

توضیحات در برنامه باعث خوانائی بیشتر و درک بهتر برنامه میشود. بنابراین توصیه بر آن است که حتی الامکان در برنامه‌ها از توضیحات استفاده نمائیم. در **C++**، توضیحات بدو صورت انجام می‌گیرد که در اسلایدهای بعد به آن اشاره شده است.

توضیحات (Comments)

الف: این نوع توضیح بوسیله // انجام می‌شود. که کامپیوتر هر چیزی را که بعد از // قرار داده شود تا انتهای آن خط اغماض می‌نماید.

مثال :

```
c=a+b;//c is equal to sum of a  and b
```

ب:توضیح نوع دوم با */ شروع شده و به /* ختم می‌شود و هر چیزی که بین* / و /* قرار گیرد اغماض می‌نماید .

مثال :

```
/ * this is a program
to calcufate sum of
n integer numbers */
```

توابع کتابخانه

زبان **C++** مجهز به تعدادی توابع کتابخانه می‌باشد. بعنوان مثـال تعدادی توابع کتابخانه برای عملیات ورودی و خروجی وجود دارند. معمولاً توابع کتابخانه مشابه ، بصـورت برنامـه‌های هـدف (برنامـه ترجمه شده بزبان ماشین) در قالب فایلهای کتابخانه دسته بندی و مورد استفاده قرا رمی‌گیرند. این فایلها را فایلهای **header** می‌نامند و دارای پسوند **.h** می‌باشند.

نحوه استفاده از توابع کتابخانه ای

برای استفاده از توابع کتابخانه خاصی بایستـــــی نام فایل header آنرا در ابتدای برنامه در دستور #include قرار دهیم.

#include < اسم فایل header >

فایل هیدر	شرح	نوع	تابع
stdlib.h	قدرمطلق i	*int*	abs(i)
math.h	کسینوس d	*double*	cos(d)
math.h	e^x	*double*	exp(d)
math.h	$\log_e d$	*double*	log(d)
math.h	$\text{Log}_{10} d$	*double*	log۱۰(d)
math.h	سینوس d	*double*	sin(d)
math.h	جذر d	*double*	sqrt(d)
string.h	تعداد کرکترهای رشته s	*int*	strlen(s)
math.h	تانژانت d	*double*	tan(d)
stdlib.h	کداسکی کرکتر c	*int*	toascii(c)
stdlib.h	تبدیل به حروف کوچک	*int*	tolower(c)
stdlib.h	تبدیل به حرف بزرگ	*int*	toupper(c)

برنامه در C++

اکنون باتوجه به مطالب گفته شده قادر خواهیم بود که تعدادی برنامه ساده و کوچک به زبان C++ بنویسیم. برای نوشتن برنامه بایستی دستورالعملها را در تابع main( ) قرار دهیم و برای اینکار می‌توان به یکی از دو طریقی که در اسلایدهای بعد آمده است ، عمل نمود.

روش اول :

```
#include     <      >
int main( )
{
دستورالعمل ۱ ;
دستورالعمل ۲ ;
.
.
.
n  دستورالعمل;
return ۰ ;
}
```

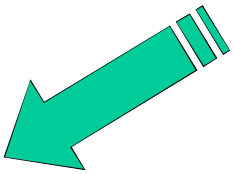
روش دوم :

```
#include     <       >
void main( )
{
دستورالعمل ۱ ;
دستورالعمل ۲ ;
.
.
.
n  دستورالعمل;
}
```

برنامه ای که پیغام C++ is an object oriented language را روی صفحه مانیتور نمایش می دهد.

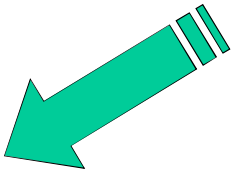
```
#include    <iostream.h>
int main( )
{
cout <<"C++ is an object oriented language \n" ;
return ۰ ;
}
```

برنامه زیر یک حرف انگلیسی کوچک را گرفته به حرف بزرگ تبدیل می‌نماید.



```
#include    <iostream.h>
#include    <stdlib. h>
int main( )
{
 char c۱ , c۲;
cout << "Enter a lowercase letter:"
cin >> c۱;
c۲ = toupper(c۱);
cout << c۲ << endl;
return ۰; }
```

دو عدد از نوع اعشاری را گرفته مجموع و حاصلضرب آنها را محاسبه و نمایش می‌دهد.



```
#include    <iostream.h>
int main( )
{
float        x,y,s,p ;
cin >> x >> y ;
s= x+y ;
p=x*y;
cout << s <<endl << p;
return ۰ ;
}
```

فصل دوم

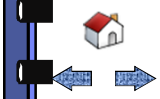
ساختارهای تصمیم گیری و تکرار

فهرست مطالب فصل دوم

عملگرهای رابطه ای

از این عملگرها برای تعیین اینکه آیا دو عدد با هم معادلند یا یکی از دیگری بزرگتر یا کوچکتر می‌باشد استفاده می‌گردد. عملگرهای رابطه‌ای عبارتند از:

مساوي	= =
مخالف	! =
بزرگتر	>
بزرگتر يا مساوي	> =
كوچكتر	<
كوچكتر يا مساوي	<=



عملگر شرطی

شکل کلی عملگر شرطی بصورت زیر می‌باشد:

```
expression _ test ? expression _ true : expression _ false
```

عملگر شرطی تنها عملگری در **C++** می‌باشد که دارای سه عملوند می‌باشد.

مثال ۱ :

```
int x=۱۰,y=۲۰,b;
b=(x>y) ? x : y ;
```

این دو دستور العمل باعث میشوند که ماکزیمم مقادیر **y** و **x** در **b** قرار بگیرد.

مثال ۲ :

```
x>=۱۰ ? cout << "passed" : cout << "failed" ;
```

اگر مقدار **x** بزرگتر یا مساوی ده باشد رشتهٔ **passed** در غیر اینصورت رشته **failed** نمایش داده میشود.

دستورالعمل شرطی

توسط این دستور شرطی را تست نموده و بسته به آنکه شرط درست یـا غلـط باشـد عکس‌العمـل خاصی را نشان دهیم.

```
if( عبارت )
{
        دستورالعمل ۱ ;
        .
        .
        دستورالعمل n;
}
else
{
        دستورالعمل ۱ ;
        .
        .
        دستورالعمل n;
}
```

مثال ١ :

```
if(x != y)
{
cout << x ;
++ x ;
}
else
        {
cout << y ;
- - y ;
         }
```

مثال ٢:

برنامه زیر یک عدد اعشاری را از ورودی گرفته جذر آنرا محاسبه می‌نماید.

```
#include    <iostream.h>
#include    <math . h>
int main( )
{
float  x,s;
cin >> x ;
if( x < ٠ )
cout << " x is negative" << endl ;
else
{
s = sqrt(x) ;
cout << s << endl ;
                }
return ٠;
}
```

عملگر کاما

تعدادی عبارت را می‌توان با کاما بهم متصل نمود و تشکیل یک عبارت پیچیده‌تری را داد. این عبارتها به ترتیب از چپ به راست ارزیابی شده و مقدار عبارت معادل عبارت n می‌باشد.

(عبارت n , ... , عبارت ۳ , عبارت ۲ , عبارت ۱)

مثال :

اگر داشته باشیم int a=۲ , b=۴ , c=۵ ; عبارت زیر را در نظر بگیرید:

(++ a , a+b, ++ c, c+b)

مقدار عبارت برابر است با b+c که معادل ۱۰ می‌باشد.

عملگرهای منطقی

با استفاده از عملگرهای منطقی می‌توان شرطهای ترکیبی در برنامه ایجاد نمود.
عملگرهای منطقی عبارتست از :

AND
OR
NOT

که در **C++** به ترتیب بصورت زیر نشان داده میشود.

&&
||
!

جدول درستی سه عملگر شرطی

And

a	b	a && b
true	true	True
true	false	False
false	true	False
false	false	False

Or

a	b	a || b
true	true	True
true	false	True
false	true	True
false	false	False

Not

a	!a
true	False
false	True

چند مثال :

```
if ((x= = ۵) ||(y != ۰))
   cout << x << endl ;
```

اگر x برابر با ۵ یا y مخالف صفر باشد مقدار x نمایش داده شود .

```
if(x)
 x = ۰ ;
```

اگر مقدار x مخالف صفر باشد، آنگاه x برابر با صفر شود .

برنامه زیر طول سه پاره‌خط را از ورودی گرفته مشخص می‌نماید که آیا تشکیل یک مثلث میدهد یا خیر؟

```
#include      < iostream.h >
int main( )
{
float  a, b, c;
cout << "Enter three real numbers" << endl ;
cin >> a >> b >> c; //
if(( a < b + c) &&(b < a+c) &&(c < a+b))
cout << "It is a triangle" ;
else
cout << "Not a triangle" ;
return  ۰ ;
}
```

دستورالعمل For

از دستور العمل **for** برای تکرار دستورالعملها استفاده میشود. شکل کلی دستور **for** بصورت زیر می‌باشد:

```
for (عبارت ۳ ; عبارت ۲ ;  عبارت ۱)
 {
  ; دستورالعمل ۱
  ; دستورالعمل ۲
  .
  .
  .
  n دستورالعمل;
 }
```


برنامه زیر عدد صحیح و مثبت n را از ورودی گرفته فاکتوریل آنرا محاسبه ونمایش می‌دهد.

```
#include        <iostream.h>
int     main( )
{
int  n, i ;
long fact = ۱ ;
cout << "Enter a positive integer number";
cin >> n;
for( i=۱; i<=n; ++i) fact *= i;
 cout << fact << endl;
return ۰ ;
}
```


برنامه زیر مجموع اعداد صحیح و متوالی بین ۱ تا n را محاسبه نموده و نمایش می‌دهد.

```
#include        <iostream.h>
int     main( )
{
int n, i=۱ ;
long s = ۰ ;
cin >> n ;
for(; i<=n; i++) s += i;
cout << s ;
return  ۰ ; }
```

برنامه زیر ارقام ۰ تا ۹ را نمایش می‌دهد.

```
#include    <iostream.h>
int main( )
{
int  j=۰ ;
for( ; j <= ۹ ; ) cout << j++ << endl;
return ۰ ;
}
```

برنامه زیر کلیه اعداد سه رقمی که با ارقام ۱ ، ۲ ، ۳ ایجاد می‌شوند را نمایش می‌دهد.

```
#include   <iostream.h>
int main( )
{
int  i,j,k,n;
for(i=۱; i<=۳; ++i)
for(j=۱; j<=۳; ++j)
for(k=۱; k<=۳; ++k)
{
n=i*۱۰۰ + j*۱۰+k;
cout << n << '\n' ;
}
return ۰ ;
}
```

فصل سوم

سایر ساختارهای تکرار

فهرست مطالب فصل سوم

دستورالعمل while

از این دستور العمل مانند دستورالعمل **for** برای تکرار یک دستورالعمل ساده یا ترکیبی استفاده می‌گردد. شکل کلی این دستور العمل بصورت زیر می‌باشد.

```
while( شرط)
{
 دستورالعمل ۱ ;
 دستورالعمل ۲ ;
    .
    .
 دستورالعمل n ;
}
```

تفاوت دستورهای while و for

دستورالعمل for زمانی استفاده میشود که تعداد دفعات تکرار از قبل مشخص و معین باشد. در صورتیکه تعداد دفعات تکرار مشخص نباشد بایستی از دستورالعمل while استفاده نمود.

مثال :

```
int x=۰
while(x<۵)
cout << x ++<< endl;
```

با اجرای قطعه برنامه فوق مقادیر زیر نمایش داده میشود :

۰
۱
۲
۳
۴

برنامهٔ فوق n مقدار از نوع اعشاری را گرفته میانگین آنها را محاسبه و در متغیر avg قرار می‌دهد.

```
#inclode   <iostream.h>
int main( )
{
int count = ۰ , n;
float  x, sum = ۰ , avg ;
cin >> n ; /*  تعداد مقادیر ورودی n*/
while(count < n){
cin >> x ;
sum += x ;
++ count ; }
avg = sum / n ;
cout << avg << endl;
return ۰ ; }
```

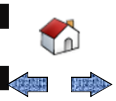
دستورالعمل do while

این دستور العمل نیز برای تکرار یک دستورالعمل ساده یا ترکیبی استفاده می‌شود. شکل کلی این دستورالعمل بصورت زیر می‌باشد.

```
do
{
 دستورالعمل ۱ ;
 دستورالعمل ۲ ;
    .
    .
 دستورالعمل n ;
} while( شرط);
```

تفاوت دستورهای while وdo while

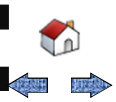
در دستورالعمل while ابتدا مقدار شرط ارزیابی شده اما در دستورالعمل do while ابتدا دستورالعمل اجرا شده سپس مقدار شرط ارزیابی می‌گردد. بنابراین دستورالعمل do while حداقل یک بار انجام میشود .



مثال :

```
#include   <iostream.h>
int main( )
{
int count = ۰;
do
cout << count ++<<endl ;
while(count <= ۹);
return ۰ ; }
```

ارقام ۰ تا ۹ را روی ده خط نمایش می‌دهد



دستورالعمل break

این دستورالعمل باعث توقف دستورالعملهای تکرار(for , while ,do while) شده و کنترل به خارج از این دستورالعملها منتقل می‌نماید.

مثال ۱ :

```
#include   <iostream.h>
int main( )
{
float x, s=۰/۰ ;
cin >> x ;
while(x <= ۱۰۰۰/۰) {
if(x < ۰/۰){
cout << "Error-Negative Value" ;
break;
}
s += x ;
cin >> x ;}
cout << s << endl ;
return ۰ ; }
```

مثال ٢:

```
#include   <iostream.h>
int main( )
{
int count = ٠ ;
while( ١ )
{
count ++ ;
if(count > ١٠ )
break ;
}
cout << "counter :   " << count << "\n";
return ٠ ;
}
```


مثال ٣:

```
#include   <iostream.h>
void  main( )
{
int count;
float x, sum = ٠;
cin >> x ;
for(count = ١; x < ١٠٠٠ . ٠; ++ count )
{
cin >> x ;
if(x < ٠/٠) {
cout << "Error – Negative value " <<endl;
break ;
}
sum += x ; }
cout << sum << '\n' ; }
```


مثال ۴:

```
#include   <iostream.h>
int main( )
{
float x , sum = ۰/۰ ;
do
{
cin >> x ;
if(x < ۰/۰)
    {
cout << "Error – Negative Value" << endl ;
break ;
    }
sum += x ;
} while(x <= ۱۰۰۰/۰);
cout << sum << endl ;
return ۰ ; }
```

دستورالعمل continue

از دستورالعمل **continue** می‌توان در دستورالعملهای تکرار **while** ، **do while** ، **for** استفاده نمود. این دستورالعمل باعث می‌شود که کنترل بابتدای دستورالعملهای تکرار منتقل گردد.

Continue

مثال ١:

```
#include   <iostream.h>
int main( )
{
float  x, sum = ٠/٠ ;
Do
 {
cin >> x ;
if(x < ٠ . ٠)
      {
cout  << "Error" << endl ;
continue ;
      }
sum += x ;
} while(x <= ١٠٠٠/٠ );
cout << sum ;
return ٠ ; }
```

مثال ٢:

```
#include   <iostream.h>
int main( )
{
int n , navg = ٠ ;
float  x, avg, sum = ٠ ;
cin >> n ; / * عبارت از تعداد اعداد ورودی n * /
for(int count = ١ ; count <=n; ++ count )
{
cin >> x ;
if(x < ٠ ) continue ;
sum += x ;
++ navg ;
}
avg = sum / navg;
cout << avg << endl ;
return ٠ ;
 }
```

دستورالعمل switch

همانطور که می دانید از دستورالعمل شرطی(**if else**) می توان بصورت تودرتو استفاده نمود ولی از طرفی اگر عمق استفادهٔ تو در تو از این دستورالعمل زیاد گردد، درک آنها مشکل میشود . برای حل ایـن مشـکل **C++** ، دسـتورالعمل **switch** که عملاً یک دستورالعمل چند انتخابی می‌باشد را ارائه نموده است.

switch

case

شکل کلی دستور العمل Switch

```
switch(عبارت)
{
case     valueone : statement;
                    break;
case      valuetwo: statement;
                    break;
:
case       valuen : statement;
                    break;
default: statement ;
}
```

مثال ١ :

```
#include   <iostream.h>
void main( )
{
unsigned int n ;
cin >> n;
switch(n)
{
    case  ٠:
          cout << "ZERO" << endl ;
          break;
    case  ١:
          cout << "one" << endl ;
          break ;
    case ٢:
          cout << "two" << endl ;
          break;
    default :
          cout << "default" << endl;
}      / * end of switch statement * /
}
```

مثال ٢ :

```
#include   <iostream.h>
void main( )
{
unsigned  int  n;
cin >> n ;
switch(n) {
case ٠ :
case ١:
case ٢:
       cout << "Less Than Three" << endl;
        break;
case ٣:
        cout << "Equal To Three" << endl ;
         break;
default:
        cout << "Greater Than Three" << endl;
            }
}
```

تابع cin.get() :

قطعه برنامه ذیل یک کرکتر را از صفحه کلید گرفته و نمایش می‌دهد.

```
char      x;
x = cin.get( );
cout << x ;
```

برنامهٔ ذیل یک سطر متن انگلیسی که به CTRL Z ختم میشود را گرفته دقیقاً نمایش می‌دهد.

```
#include      <iostream.h>
int main( )
{
char  x;
while((x = cin.get( ) !=EOF)
cout << x ;
return ۰ ;
 }
```

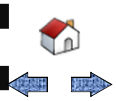
EOF به معنی End of File می‌باشد که در iostream.h تعریف شده و مقدار آن برابر با ۱- می‌باشد. مقدار آن در سیستم عامل DOS عبارتست از ctrl z .

در قطعه برنامه ذیل ازتابع cin.get() و دستور switch استفاده شده است.

```
char        x;
x = cin.get( );
switch(x)  {
case  ' r ' :
case  ' R ' :
            cout << "RED" << "\n" ;
            break ;
case  ' b ' :
case  ' B ' :
             cout << "BLUE" << endl ;
             break ;
case  ' y ' :
case  ' Y ' :
            cout <<  "YELLOW" << endl;
  }
```

برنامهٔ ذیل یک سطر متن انگلیسی را گرفته کرکترهای خالی (blank) آنرا حذف نموده و نمایش میدهد.

```
#include      <iostream.h>
int main( )
{
char next;
while((next = cin.get( ) ) !=EOF)
if(next != ' ')
cout  << next ;
return ۰ ;
 }
```



عملگر static_cast

از این عملگر برای تبدیل موقت یک نوع data به نوع دیگر استفاده می‌شود. این عملگر یک عملگر یکتائی می‌باشد.

مثال ۱:

```
int   x = ۲۵ ;
float  y ;
y = static_cast
< float >(x) ;
```

مقدار x موقتاً بصورت اعشاری در می آید و در نتیجه مقدار y برابر با ۲۵/۰ می شود. بایستی توجه داشت که نوع متغیر x عوض نمی شود بلکه موقتاً مقدار آن بصورت اعشاری در آمده است.

مثال ۲:

```
float x = ۱۴/۷۵ ;
cout <<
static_cast < int
>(x) << endl;
cout << x ;
```

ابتدا مقدار ۱۴ نمایش داده میشود و سپس مقدار ۱۴,۷۵ نمایش داده میشود.

جدول اولویت عملگرها

چپ به راست	( )
راست به چپ	sizeof - + -- ++ ( ) >Static_cast <
چپ به راست	% / *
چپ به راست	- +
چپ به راست	>> <<
چپ به راست	>= > <= <
چپ به راست	!= = =
راست به چپ	: ?
راست به چپ	%= /= *= -= += =
چپ به راست	,

فصل چهارم

اعداد تصادفی

فهرست مطالب فصل چهارم

اعداد تصادفی

مقادیر تصادفی یا شانسی در اکثر برنامه‌های کاربردی در زمینه شبیه سازی و بازیهای کامپیوتری نقش مهمی را ایفا می‌نمایند. برای ایجاد یک عدد تصادفی صحیح بین ۰ و ۳۲۷۶۷ بایستی از تابع **rand**( ) استفاده نمائیم.

rand( )

برنامه زیر ۱۰ عدد تصادفی بین ۰ و ۳۲۷۶۷ را ایجاد می‌نماید.

```
#include        <stdlib.h>
#include        < iostream.h>
int main( )
{
for(int j=۱; j<=۱۰; ++j)
cout << rand(  ) << '\n' ;
return ۰ ;
}
```

نکته :

اگر برنامهٔ فوق را چندبار اجرا نمائیم جواب یکسانی را از کامپیوتری‌می گیریم.
برای تصادفی کردن اعداد می‌بایستی از تابع **srand( )** استفاده نمائیم.
این تابع به یک آرگومان صحیح از نوع **unsigned** نیاز دارد.
به این آرگومان **seed** گفته می‌شود.

در اسلاید بعد برنامه قبلی را با تابع **srand( )** نوشته ایم.

برنامه زیر ۱۰ عدد تصادفی بین ۰ و ۳۲۷۶۷ را ایجاد می‌نماید. (srand( ))

```
#include          <stdlib.h>
#include          < iostream.h>
int main( )
{
unsigned     seed;
cout << "Enter seed value :  " ;
cin >> seed ;
srand(seed);
for(int  j=۱; j<=۱۰; ++j)
cout << rand( ) << ' \n ';
return ۰ ;
 }
```

برنامه زیر نتیجه پرتاب دو تاس را نمایش می‌دهد.

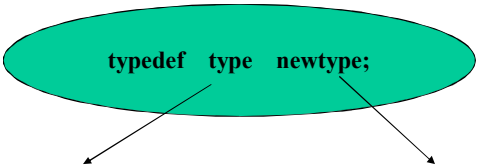
```
#include          < iostream.h>
#include          <stdlib.h>
int main( )
{
unsigned     seed, d۱, d۲;
cout << "Enter seed: " ;
cin >> seed ;
srand(seed) ;
d۱= ۱+rand( )% ۶ ;
d۲= ۱+rand( )% ۶ ;
cout << d۱ << "        "<< d۲ ;
return ۰ ;
 }
```

برنامه زیر ۱۰ اعداد شانسی بین ۰ و ۱ را نمایش می‌دهد.

```
#include        <stdlib.h>
#include        < iostream.h>
int main( )
{
unsigned seed ;
cout << "Enter seed: " ;
cin >> seed ;
srand(seed) ;
for(int i=۱; i<=۱۰; ++i)
cout << rand( ) / ۳۲۷۶۸/۰ << endl ;
return ۰ ;
}
```

تعریف نوع داده (typedef)

از **typedef** می‌توان برای تعریف نوع داده‌های جدید که معادل نوع داده‌های موجود باشد استفاده نمود. شکل کلی عبارتست از :



مثال :

```
typedef  int  integer;
```

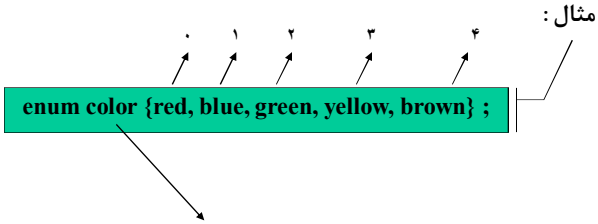
حال می‌توان **y** و **x** را بصورت زیر تعریف نمود :

```
integer  x,y;
```

داده‌های از نوع شمارشی

بمنظور معرفی داده‌های از نوع شمارشی از کلمه **enum** استفاده می‌گردد.

مثال :



color یک نوع داده شمارشی می‌باشد.

چند مثال :

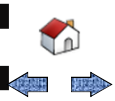
```
emum status {married, devorced, vidow, single};
status  a ;
a= single ;
```

```
enum days {sat, sun, mon, tue, wed, thr,
                                    fri};
```

```
enum bread {lavash, fantezi, taftoon, barbari};
```

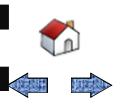
```
enum color { yellow, red=۲, brown, white };
color  x=brown;
```

۰ ۳ ۴



توجه :

بایستی در نظر داشت که داده‌های از نوع شمارشی در عملیات ورودی و خروجی شرکت نمی‌نمایند. بعبارت دیگر مقادیر داده‌های از نوع شمارشی بایستی در برنامه تعیین نمود. دستورالعلهای ورودی و خروجی مانند cin و cout در مورد داده‌های شمارشی نمی‌توان استفاده نمود.



فرمتهای مختلفه مقادیر خروجی

مقدار x بطور غیر علمی با نقطه اعشار ثابت نمایش داده می شود.

```
include <iomanip.h>

double   x=۱۰۵۰ ;
cout << setiosflags(ios  : : fixed | ios: : showpoint ) << setw(۲۳)
     << setprecision(۲) << x << endl ;
```

مقدار x با طول میدان ۲۳ نمایش داده می شود.

مقدار x با دو رقم اعشار نمایش داده می شود.

بنابراین مقدار x بصورت زیر نمایش داده می شود :

۱۰۵۰/۰۰ شانزده ستون خالی

فصل پنجم

آرایه ها

فهرست مطالب فصل پنجم

آرایه یک بعدی

آرایه یک فضای پیوسته از حافظه اصلی کامپیوتر می‌باشد که می‌تواند چندین مقدا را در خود جای دهد.

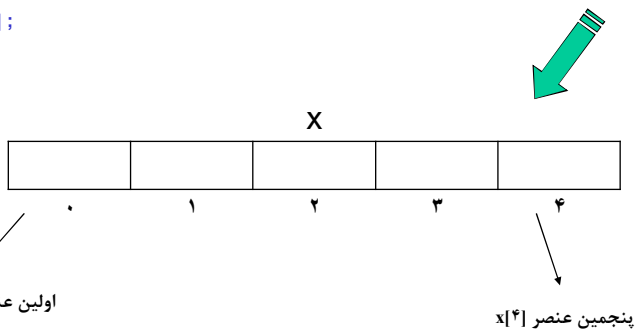
- کلیه عناصر یک آرایه از یک نوع می‌باشند.
- عناصر آرایه بوسیله اندیس آنها مشخص می‌شوند.
- در C++ ، اندیس آرایه از صفر شروع می‌شود.

کاربرد آرایه ها

آرایه‌ها در برنامه‌نویسی در مواردی کاربرد دارند که بخواهیم اطلاعات و داده‌ها را در طول اجرای برنامه حفظ نمائیم.

آرایه یک بعدی از نوع int

```
int  x[۵] ;
```



تخصیص مقادیر اولیه به عناصر آرایه :

```
int  x[۵]= {۴, ۲, ۵, ۱۷, ۳۰};
```

x

۴	۲	۵	۱۷	۳۰
۰	۱	۲	۳	۴

دریافت مقادیر عناصر آرایه :

```
int  x[۵];
for(int    i=۰; i<=۴; ++i)
cin >> x[ i ] ;
```

نمایش مقادیر عناصر آرایه :

```
for(int   i=۰; i<=۵; ++i) cout << x[ i ] ;
```

اگر تعداد مقادیر اولیه کمتر از تعداد عضوهای آرایه باشد عضوهای باقیمانده بطور اتوماتیک، مقدار اولیه صفر می‌گیرند.

```
int   x[۵] = {۱۲, ۵, ۷};
```

x

۱۲	۵	۷	۰	۰
۰	۱	۲	۳	۴

بایستی توجه داشت که آرایه‌ها به صورت ضمنی مقدار اولیه صفر نمی‌گیرند. برنامه نویس باید به عضو اول آرایه، مقدار اولیه صفر تخصیص دهد تا عضوهای باقی‌مانده بطور اتوماتیک، مقدار اولیه صفر بگیرند.

```
int   x[۵] = {۰} ;
```

x

۰	۰	۰	۰	۰
۰	۱	۲	۳	۴

دستور زیر یک آرایه یک بعدی شش عنصری از نوع float ایجاد می‌نماید.

```
float x[  ] = {۲/۴, ۶/۳, -۱۷/۱, ۱۴/۲, ۵/۹, ۱۶/۵} ;
```

X

۲/۴	۶/۳	-۱۷/۱	۱۴/۲	۵/۹	۱۶/۵
۰	۱	۲	۳	۴	۵

برنامه ذیل ۱۰۰ عدد اعشاری و مثبت را گرفته تشکیل یک آرایه میدهد سپس مجموع عناصر آرایه را مشخص نموده نمایش می‌دهد.

```
#include  <iostream.h>
#include   <iomanip.h>
int main( )
{
const int arrsize = ۱۰۰ ;
float x[ arrsize], tot = ۰/۰ ;
for(int   j=۰; j<arrsize; j++)
cin >> x[ j ];
for(j=۰; j<arrsize; j++)
cout << setiosflags(ios::fixed  ios :: showpoint ) << setw(۱۲) <<
    setprecision(۲) << x[ j ] << endl;
for(j=۰; j<arrsize; j++)
tot += x[ j ] ;
cout << tot ;
return ۰ ;
}
```

برنامه ذیل ۲۰ عدد اعشاری را گرفته تشکیل یک آرایه داده سپس کوچکترین عنصر آرایه را مشخص و نمایش می‌دهد.

```
#include  <iostream.h>
#include   <conio.h>
int main( )
{
float   x[۲۰], s;
int  j ;
clrscr( ) ;
for(j=۰; j<۲۰ ; ++j) cin >> x[ j ];
s = x[۰ ] ;
for(j=۱; j<۲۰; ++j)
if(x[ j] <s) s = x[ j ];
cout << s << endl;
return ۰;
}
```

برنامه زیر ۱۰۰ عدد اعشاری را گرفته بروش حبابی (Bubble sort) بصورت صعودی مرتب می‌نماید.

```
#include  <iostream.h>
#include  <conio.h>
int main ( )
{
float  x[۱۰۰] , temp;
int   i,j ;
clrscr(  );
for(i=۰; i<۱۰۰; ++i) cin >> x[i ];
for(i=۰; i<۹۹; i++)
for(j=i+۱  ; j<۱۰۰; j++)
if(x[ j ] < x[i ]{
temp = x[ j ] ;
x[ j ] = x[ i ];
x[ i ] = temp ;
}
for(i=۰; i<=۹۹; i++)
cout << x[ i ] << endl;
return ۰ ;
}
```

آرایه‌های دوبعدی (ماتریس‌ها)

ماتریسها بوسیله آرایه‌های دوبعدی در کامپیوترنمایش داده میشوند.

```
int a[۳][۴];
```

	ستون ۰	ستون ۱	ستون ۲	ستون ۳
سطر ۰	a[۰][۰]	a[۰][۱]	a[۰][۲]	a[۰][۳]
سطر ۱	a[۱][۰]	a[۱][۱]	a[۱][۲]	a[۱][۳]
سطر ۲	a[۲][۰]	a[۲][۱]	a[۲][۲]	a[۲][۳]

تخصیص مقادیر اولیه به عناصر آرایه :

```
int a[۳][۴]={ {۱،۲،۳،۴}, {۵،۶،۷،۸}, {۹،۱۰،۱۱،۱۲} } ;
```

	۰	۱	۲	۳
۰	۱	۲	۳	۴
۱	۵	۶	۷	۸
۲	۹	۱۰	۱۱	۱۲

نکته ۱ :

```
int   a[۳][۴]= { {۱}, {۲,۳} , {۴,۵,۶} } ;
```

	۰	۱	۲	۳
۰	۱	۰	۰	۰
۱	۲	۳	۰	۰
۲	۴	۵	۶	۰

نکته ۲ :

```
int   a[۳][۴]= {۱, ۲, ۳, ۴,۵ } ;
```

	۰	۱	۲	۳
۰	۱	۲	۳	۴
۱	۵	۰	۰	۰
۲	۰	۰	۰	۰

نکته ۳ :

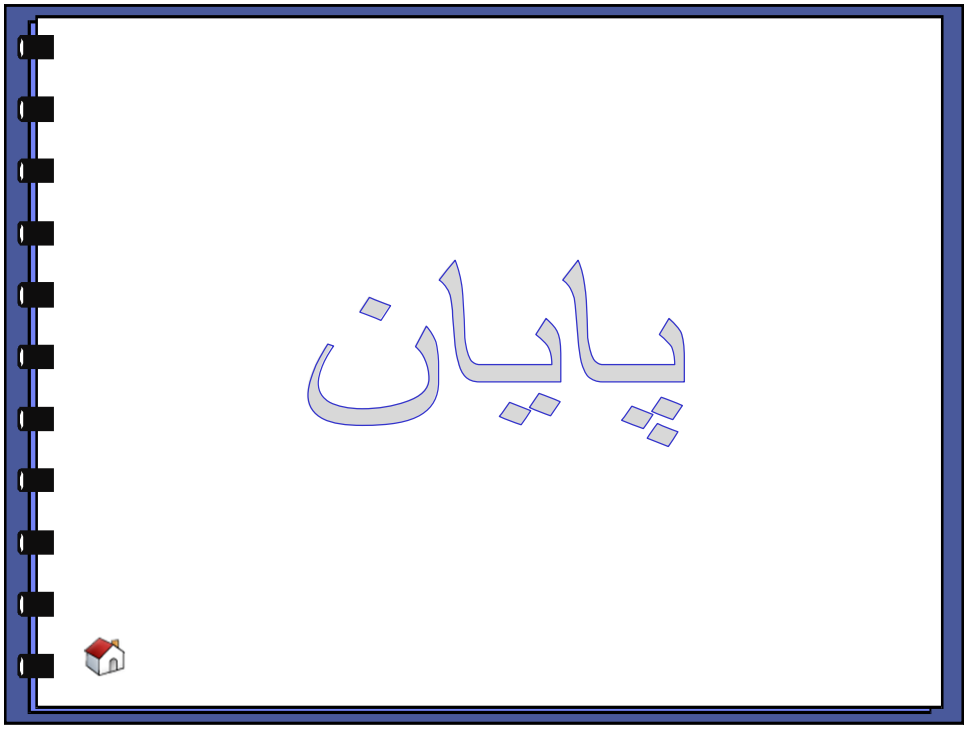
در یک آرایهٔ دواندیسی، هر سطر، در حقیقت آرایه‌ای یک اندیسی است. در اعلان آرایه‌های دواندیسی ذکر تعداد ستونها الزامی است.

```
int   a[  ][۴]={۱،۲،۳،۴،۵};
```

	۰	۱	۲	۳
۰	۱	۲	۳	۴
۱	۵	۰	۰	۰

برنامه زیر یک ماتریس ۳*۴ را گرفته مجموع عناصر آن را مشخص نموده و نمایش می‌دهد.

```
#include  <iostream.h>
#include  <conio.h>
int main( )
{
float   x[۳][۴], total= ۰/۰;
int   i, j ;
// generate matrix x.
for(i=۰; i<۳; ++i)
for (j=۰; j<۳; j++)
cin >> x[ i ][ j ];
// calculate the sum of elements.
for(i=۰; i<۳; ++i)
for(j=۰; j<۴; j++)
tot + = x [ i ][ j ];
cout << "total =  " << total << endl;
return ۰ ;
}
```


پایان