بسم الله الرحمن الرحیم

آموزش مقدماتی برنامه نویسی به زبان C++

مهندس براتیان

شکل کلی یک برنامه c++:

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

int main()
{

-
-
-
-
-
-


   return 0;
}
```

یک برنامه به زبان C++ از یک تا چند تابع تشکیل می شود که نام تابع اصلی main می باشد و دستورات هر تابع در داخل { } قرار می گیرند.ضمنا برنامه نویس باید نوع تابعی که ایجاد می کند را مشخص کند. اگر برنامه شما از چند تابع تشکیل شده باشد اولین تابعی که اجرا خواهد شد تابع main می باشد.

برای نوشتن و اجرای یک برنامه C++ می توان از نرم افزار های Turbo C++ ، Free-C ، .... استفاده کرد و همچنین درصورت عدم دسترسی به این نرم افزارها می توان به صورت آنلاین از طریق وب سایت های www.onlinegdb.com و www.w3schools.com کد نویسی ، اجرا و نتیجه را مشاهده کرد.

یک نمونه برنامه c++ دیگر:

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

int main()
{
-
-
-

   return 0;
}
//---------------------------------------------------------------
float func1()
{
-
-
-
return ;
}
//---------------------------------------------------------------
float func2()
{
-
-
-


Return;
}
```

متغیر(Variable): در برنامه نویسی از متغیرها برای ذخیره اطلاعات استفاده می شود. برای هر متغیر باید یک نام مجاز و مرتبط انتخاب کرد و ضمناً نوع متغیر را نیز باید تعیین کنیم.نوع متغیر مشخص می کند که قرار است چه نوع داده ای در آن ذخیره شود.

انواع داده ها در C++:

int, float, double,char, void, bool

از نوع int برای ذخیره اعداد صحیح مانند 128، 5، 4508 استفاده می شود.

نوع float برای ذخیره اعداد اعشاری مثل 12.5، 7805.11 بکار می رود.

نوع double برای اعداد اعشاری بزرگتر از float استفاده می شود.

نوع char برای ذخیره داده های کاراکتری مانند 'W' , 'z' , 'a' : بکار می رود.

از bool برای ذخیره مقادیر منطقی استفاده می شود (درستی یا نادرستی).

نوع void هیچ مقداری را نمی گیرد.

قواعد نامگذاری متغیرها:

- ✓ رای نامگذاری متغیرها از ترکیبی از حروف a تا z یا A تا Z ، ارقام(0 تا 9) و خط ربط (_) استفاده می شود.
- ✓ نام متغیر نباید با رقم (0-9) شروع شود
- ✓ نام می تواند هر طولی داشته باشد اما فقط 31 کاراکتر ابتدایی استفاده می شوند.
- ✓ برای نام گذاری متغیرها از کلمات رزرو شده نمی توان استفاده کرد.

نام متغیر	وضعیت	علت
temp	مجاز	
3Ab	غیر مجاز	با عدد شروع شده است
Ab3	مجاز	
for	غیر مجاز	کلمات رزرو شده
main	غیر مجاز	کلمات رزرو شده
Temp_4	مجاز	
Temp-4	غیر مجاز	خط تیره (منها)
A*B	غیر مجاز	*

نحوه تعریف متغیر:

برای استفاده از یک متغیر ابتدا باید آن را در برنامه تعریف نماییم که روش تعریف متغیر بصورت زیر است:

نوع متغیر نام متغیر;

```
int count;
```

متغیر count از نوع int تعریف شده است.یعنی میتوان در آن مقادیر صحیح ذخیره کرد.

```
float A,B;
```

A و B دو متغیر از نوع float هستند یعنی میتوان اعداد اعشاری در آنها ذخیره کرد.

روش های مقدار دهی به متغیرها:

1-مقدار دهی مستقیم به کمک دستور انتساب(=)

2-استفاده از دستور ورودی cin

مثال:مقدار دهی مستقیم

`int A;` `A=20;`	تعریف متغیر A از نوع صحیح ذخیره عدد 20 در متغیر A

نکته:در هنگام تعریف متغیر میتوان مقدار اولیه را نیز تعیین کرد.

```
int A=20;
float ave=15.5,radius=4.5;
```

نکته:در بعضی از کامپایلرها هنگام مقداردهی مستقیم به متغیر از نوع float بایستی از کاراکتر f در انتهای مقدار عددی استفاده کرد در غیر اینصورت خطا گرفته می شود.

```
float pi=3.14f;
```

```
float X;
X=3.15f;
```

آشنایی با دستور ورودی CIN:

cin>>نام متغیر;

`int a;` **`cin>>a;`**	**هنگامیکه کامپایلر به این دستور میرسد منتظر می ماند کاربر مقداری را وارد کند.پس از وارد کردن مقدار و زدن کلید Enter ، مقدار وارد شده در متغیر ذخیره می شود.**

آشنایی با دستور خروجی Cout:

کاربرد دستور cout

1- نمایش محتوای متغیرها و ثابت ها

2- نمایش متن و پیغام

کاربرد cout برای نمایش محتوای متغیر

مقدار 20 در خروجی نمایش داده می شود.	`int A;`
	`A=20;`
	`cout<<A;`

مثال:

خروجی:مقدار متغیر A و B در کنار هم چاپ می شوند. 2030	`int A=20,B=30;` `cout<<A<<B;`

مثال:

خروجی: 20 30	`int A=20,B=30;` `cout<<A<<endl<<B;`

کاربرد endl در دستور cout:بردن مکان نما به خط بعد در خروجی

کاربرد دستور cout برای چاپ پیام

```
cout <<”Message“;
```

متن hello world در خروجی نمایش داده می شود.	`cout<<”hello world”;`

کاراکتر های ویژه مورد استفاده در دستور cout:

کاراکتر	کاربرد
\n	رفتن به خط بعدی (معادل endl)
\t	مکان نما را به اندازه یک تب جلو میبرد(یعنی مانند آنکه یکبار دکمه تب در کامپیوتر کاربر زده شود).
\a	بوق (alert)سیستم را بصدا درمی آورد.
\\	چاپ یک کاراکتر \
\’	چاپ ‘
\”	چاپ “

خروجی دستورات زیر :

```
cout<<”Hello \t saeed \n Hello \t reza”;
```

خروجی:

```
Hello        saeed
Hello        reza
```

```
Cout <<” I Love \” Iran \” “;
```

خروجی:

```
I love “ Iran “
```

نکته مهم:برای استفاده از دستورات ورودی `cin` و `cout` بایستی در هدرفایل بسته به کامپایلر مورد استفاده عبارات زیر را باید بنویسید.

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

مثال1:در برنامه کامل زیر پیغام `hello world` در خروجی نمایش داده خواهد شد.

```
#include<iostrem.h>
using namespace std;
int main()
{

 cout <<”hello world”;
 return 0;

}
```

دلیل استفاده از `using namespace std;` در هدر برنامه:

اگر از این عبارت استفاده نکنیم آنگاه دستورات `cin`و `cout` را باید با فرم زیر استفاده کنیم.

```
std::cin
std::cout

Example:

int A;
std::cin>>A;

int B=20;
std::cout>>B;
```

مثال2:در برنامه زیر دو مقدار دریافت و حاصل جمع آنها نمایش داده می شود.

روش اول:

سورس برنامه	خروجی برنامه

#include<iostream.h>
using namespace std;
int main()
{
int a,b,c;
cin>>a;
cin>>b;
c=a+b;
cout<<c;
 	
return 0;
 	
}
	20
	30
	50

روش دوم(روش بهتر):

سورس برنامه	خروجی برنامه

#include<iostream.h>
using namespace std;
int main()
{
  cout<<”please enter a:”;
  cin >>a;
 	
  cout<<”please enter b:”;
  cin>>b;
 	
  c=a+b;
 	
  cout<<endl<<”sum=”<<c;
 	
return 0;
}
	please enter a:20
	please enter b:30
	sum=50

مثال 3:در برنامه زیر شعاع دایره دریافت محیط و مساحت دایره محاسبه و نتیجه نمایش داده می شود.

سورس برنامه	خروجی برنامه

#include<iostream.h>
using namespace std;
int main()
{
  float r,area,prime;
  cout<<"please enter radius:";
  cin >>r;
 	
  prime=2*3.14*r;
  area=3.14*r*r;
 	
  cout<<"\n prime="<<prime;
  cout<<"\n area="<<area;
  return 0;
}
	please enter radius:3
	prime=18.84
	area=28.26

نحوه تعریف ثابت در c++:

ثوابت مقادیر در برنامه هستند که در طول اجرای برنامه ثابت هستند و تغییر مقدار آن ها امکان پذیر نیست .

در زبان C++ دو روش ساده برای تعریف ثابت وجود دارد:

- استفاده از کلمه کلیدی `const`
- استفاده از پیش پردازنده `#define`

برای تعریف ثوابت با دستور #define به صورت زیر قبل از تابع main() عمل می شود :

#define نام ثابت مقدار

مثال:

```
#define   p   3.14

#define   Temp  100
```

مثال:برنامه محاسبه مساحت مستطیل با طول 10 و عرض 5

```
#include<iostream>
using namespace std;
#define LENGTH 10
#define WIDTH  5
#define NEWLINE '\n'
int main()
{
   int area  ;
   area = LENGTH * WIDTH;
   cout >> area;
   cout >> NEWLINE;
   return;0
{
```

نحوه تعریف ثابت به کمک Const

```
const        نوع        نام ثابت=مقدار ;
```

const float pi=3.14;

const int n=100;

مثال 3:در برنامه زیر شعاع دایره دریافت محیط و مساحت دایره محاسبه و نتیجه نمایش داده می شود.

سورس برنامه	خروجی برنامه

#include<iostream.h>
using namespace std;
int main()
{
  float r,Area,Prime;
  const float Pi=3.141592;
  cout<<"Please Enter Radius:";
  cin >>r;
 	
  Prime=2*Pi*r;
  Area=Pi*r*r;
 	
  Cout<<"\n Prime="<<prime;
  Cout<<"\n Area="<<Area;
  return 0;
}
	Please Enter Radius:3
	Prime=18.84
	Area=28.26

انواع عملگرها در C++

♦عملگرهای محاسباتی

عملگرهای محاسباتی، اعمال محاسباتی را بر روی عملوندها انجام میدهند. این عملگرها در جدول زیر ذکر شده اند :

عملگر	نام
+	جمع
-	تفریق
*	ضرب
/	تقسیم اعشاری
%	باقیمانده تقسیم صحیح
++	افزایش یک واحد
--	کاهش یک واحد

مثال: نتیجه عبارات:

با فرض اینکه سه متغیر از نوع صحیح باشند. Int A=20,B=8,C;	
نتیجه	عبارت
C=20+8=28	C=A+B;
C=20-8=12	C=A-B;
C=20*8=160	C=A*B;
C=20%8=4	C=A%B;
A=20+1=21	++A;
B=8-1=7	--B;
با فرض اینکه سه متغیر از نوع اعشاری باشند. float A=20,B=8,C;	
نتیجه	عبارت
C=20/8=2.5	C=A/B;

نکته در مورد عملگرهای ++و--:

عملگر کاهش (--) یک واحد از عملوندش کم میکند و نتیجه را در آن عملوند قرار میدهد و عملگر افزایش (++) یک واحد به عملوندش اضافه میکند. عملگر افزایش و کاهش چه قبل از عملوند باشند و چه بعد از آن، عملکرد آنها یکسان است. اما عملکرد آنها در عبارات محاسباتی با هم متفاوت است.

مثال:

Int A=20,B=8;

نتیجه	عبارت
در هر دو حالت مقدار A=21	++A; یا A++;
در هر دو حالت B=7	--B; B--;

اگر عملگرهای افزایش و کاهش در یک عبارت محاسباتی وارد شوند اگر عملگر ++ ویا – قبل از عملوند ظاهر شوند و یا بعد از عملوند ظاهر شوند نتیجه متفاوتی ایجاد می کنند.

اگر عملگرهای افزایش و کاهش قبل از عملوند باشند اول عمل ++ و یا -- انجام و سپس ادامه عبارات انجام می شود ولی اگر عملگر افزایش و کاهش بعد از عملوند باشد ابتدا عبارت محاسباتی انجام و سپس در عملوند مربوطه افزایش یا کاهش اعمال خواهد شد.

اول به X یک واحد اضافه(X=4) سپس با 7 جمع و نتیجه در Y دخیره می شود. X=4 Y=11	X=3; Y=++X + 7;
اول مقدار X با 7 جمع (3+7)و نتیجه در Y ذخیره می شود و در پایان به X اضافه می شود. Y=10 X=4	X=3; Y=X++ + 7;

عملگرهای انتساب

از ترکیب عملگرهای محاسباتی و عملگر '=' ، مجموعه دیگری از عملگرها ایجاد میشودکه عمل محاسباتی و انتساب را انجام میدهد. این عملگرها در جدول زیر آمده اند. تقدم این عملگرها پایینتر از سایر عملگرهاست.

معادل	مثال	نام	عملگر
A=A+B;	A+=B;	انتساب جمع	+=
A=A-B;	A-=B;	انتساب تفریق	-=
A=A*B;	A*=B;	انتساب ضرب	*=
A=A/B;	A/=B;	انتساب تقسیم	/=
A=A%B;	A%=B;	انتساب باقیمانده تقسیم	%=

♦عملگرهای رابطه ای

عملگرهای رابطه ای، رابطه بین عملوندها را مشخص میکنند. اعمالی مثل تساوی دو مقدار، کوچکتر یا بزرگتر بودن، مقایسه با صفر و غیره، توسط عملگرهای رابطه ای مشخص میشوند. عملگرهای رابطه ای در جدول زیر نشان داده شده اند. عملگر '==' در دستورات شرطی برای مقایسه دو مقدار مورد استفاده قرار میگیرد.

نام	عملگر
>	بزرگتر
>=	برزگتر مساوی
<	کوچکتر
<=	کوچکتر یا مساوی
==	متساوی
!=	امساوی

مثال:

```
int A=20,B=8;
bool C;
C=(A==B);
```

نتیجه:چون مقدار A با مقدار B برابر نیست مقدار 0 در C به منزله False ذخیره می شود.

مثال:

```
bool a = 2 < 3;

// a is true
```

```
 bool b = 2 > 3;

// b is true

 bool c = a && b;

// c is false

 bool d = a || b;

// d is true

 bool e = !c;

// e is true

bool f = e && d;

// f is true
```

♦عملگرهای منطقی

عملگرهای منطقی بر روی عبارات منطقی عمل میکنند. عبارات منطقی دارای دو ارزش درستی و نادرستی اند. در زبان C++ ، ارزش نادرستی با مقدار صفر و ارزش درستی با مقدار غیر صفر مشخص میشود. روش دیگر تعیین مقادیر درستی و نادرستی، استفاده از ثوابت true و false است .به معنی ارزش درستی و false به معنی نادرستی است. عملگرهای منطقی در جدول زیر آمده اند. ترتیب قرار گرفتن آنها در جدول، از تقدم بالا به پایین است.

نام	عملگر
not	!
and	&&
or	||

نتیجه عملگر '!' وقتی درست است که عملوند آن دارای ارزش نادرستی باشد. نتیجه عملگر '&&' وقتی درست است که هر دو عملوند ارزش درستی داشته باشند و نتیجه عملگر '||' وقتی نادرست است که هر دو عملوند ارزش نادرستی داشته باشند (در بقیه موارد نتیجه آن ارزش درستی دارد) .

♦عملگرهای بیتی(مطالعه عملگرهای بیتی اختیاری و فعلا نیاز نیست)

وجود عملگرهای بیتی در C++ موجب شد تا بسیاری از کارهای **زبان اسمبلی** در C++ انجام شود.
عملگرهای بیتی برای تست کردن، مقدار دادن یا شیفت دادن و سایر اعمال بر روی مقادیری به کار میروند که در یک بایت (char) یا کلمه (int) ذخیره شده اند. عملگرهای بیتی را نمیتوان با انواع float, double, long doubleو void یا سایر انواع پیچیده به کار برد. عملگرهای بیتی در جدول زیر نشان داده شده اند .

نام	عملگر
AND	&
OR	|
XOR یای انحصاری	^
NOT نقیض	~
RIGHT SHIFT شیفت به راست	<<
LEFT SHIFT شیفت به چپ	>>

نتیجه عملگر '&' وقتی یک است که هردو بیت عملوندهایش یک باشند. نتیجه عملگر '|' وقتی صفر است که هر دو بیت صفر باشند. نتیجه '^' وقتی یک است که یکی از بیت ها صفر و دیگری یک باشد. عملگرهای شیفت بر روی یک عملوند عمل میکنند و بیتهای آن را به سمت راست یا چپ شیفت میدهند. هنگام شیفت دادن به راست، بیتها از سمت راست خارج میشوند و از سمت چپ به تعداد مورد نظر، صفر وارد میشود. در شیفت به چپ بیتها از سمت چپ خارج شده، به تعداد لازم صفر از سمت راست وارد میشود.

مثال:

```
int a = 60;            //  60 = 0011 1100

int b = 13;            //  13 = 0000 1101

int c = 0;



c = a & b;             //  12 = 0000 1100



c = a | b;             //  61 = 0011 1101



c = a ^ b;             //  49 = 0011 0001



c = ~a;                // -61 = 1100 0011



c = a << 2;            // 240 = 1111 0000



c = a >> 2;            //  15 = 0000 1111
```

دستور شرطی if:

دستورات شرطی برای انجام کار ها و تصمیم گیری های مختلف بر اساس شرایط مختلف به کار می رود.

حالت اول

```
if(شرط)
{
دستورات

}
```

اگر شرط برقرار باشد مجموعه دستورات داخل آکولادها انجام می شود و در غیر اینصورت هیچ دستوری اجرا نمی شود

حالت دوم

```
if(شرط)
  {
دستورات
  }
else
  {
دستورات
  }
```

اگر شرط برقرار باشد دستورات داخل آکولادهای اول اجرا می شود وگرنه دستورات داخل آکولادهای بخش else انجام می شود.

حالت سوم

```
if(شرط)
 {
دستورات
 }
else if(شرط)
{
دستورات
}
else if (شرط)
{
دستورات
}else
{
دستورات
}
```

مثال:در تمرین زیر یک عدد دریافت می شود و مشخص می کند عدد زوج و یا فرد است.

راه حل:برای اینکه بفهمیم یک عدد زوج است یا نه آن عدد را بر 2 تقسیم می کنیم اگر باقیمانده تقسیم صفر باشد یعنی عدد زوج (even)است در غیر اینصورت عدد فرد(odd) است(عملگر ٪ باقیمانده تقسیم را بدست می آورد).

```
using namespace std;
int main()
{
  int n,b;
  cout<<"Enter Number:";
  cin >>n;

  b=n%2;(  2 محاسبه باقیمانده تقسیم عدد بر)
  if(b==0)
   {
     Cout<<"Number Is Even";
   }
  else
   {
     Cout<<"Number Is Odd";
   }

  Return 0;
}
```

نکته:هنگام کد نویسی اگر داخل آکولاد فقط یک دستور بود می توانیم آکولادها را حذف کنیم.

به اینصورت:

```
if(b==0)
     Cout<<"Number Is Even";
  else
     Cout<<"Number Is Odd";
```

اجرای برنامه:

Enter Number:5

Number Is Odd

اجرای برنامه:

Enter Number:24

Number Is Even

مثال:در برنامه زیر یک عدد دریافت و سپس معین می کند که عدد مثبت positive ، عدد منفی Negative ویا صفر Zero است.

```
using namespace std;
int main()
{
  int n;
  cout<<”Enter Number:”;
  cin >>n;

  if(n>0)

     Cout<<”Number Is Positive”;

  else if(n<0)

     Cout<<”Number Is Negative”;

  else
     cout <<”Nember Is Zero”;

  Return 0;
}
```

اجرای برنامه:

Enter Number:25

Number is Positive

اجرای برنامه:

Enter Number: 0

Number is Zero

اجرای برنامه:

Enter Number: -14

Number is Negative