

نام خدا



سری های آموزشی آشنایی با زبان برنامه نویسی ++C

قسمت اول : مبانی ++C

ویرایش : 2

### آشنایی مقدماتی با زبان ++C :

++C در اوایل دهه ی 1980 در آزمایشگاه T&AT توسط **بیارنه استراس تروپ (دانمارکی)** به منظور الحاق شیوه ی شی گزایی در زبان برنامه نویسی C طراحی گردید ...

### شروع به برنامه نویسی ++C:

اولین کاری که باید در برنامه نویسی ++C انجام دهید نوشتن تابع `main` است . تابع `main` تابع اصلی برنامه است . کد هایی که در داخل تابع `main` نوشته می شوند در ابتدای برنامه اجرا می شوند .

صورت کلی تابع `main` به صورت زیر است :

```
int main()
{
    دستورات
    return 0;
}
```

### متغیر :

خانه هایی از حافظه هستند که می توانند مقادیر مختلفی را بر اساس نوع آنها در خود نگهداری کنند .

متغیر ها و نحوه ی مقدار دهی به آنها در ++C به این صورت تعریف می شوند :

```
Variable type      variable name;
variable type      variable name1,variable name2,...,variable N;
Variable type      variable name=variable value;
```

مثال : در زیر مشاهده می کنید که متغیر majidonline و majid از نوع int تعریف شده اند و همچنین برای متغیر sadeghkhafan که از نوع int تعریف شده است مقدار اولیه ی 20 نسبت داده شده است .

```
int main()
{
    int majidonline,majid;

    int sadeghkhafan=20;

    Return 0;
}
```

\*توجه : در طول یک برنامه مقدار یک متغیر می تواند تغییر کند .

در زیر انواع متغیر ها در ++C را مشاهده می کنید :

نوع داده	حافظه (بایت)	محدوده
char	۱	۱۲۷ تا ۱۲۸
signed char	۱	۱۲۷ تا ۱۲۸
unsigned char	۱	۰ تا ۲۵۵
int	۲	برای سیستم ۱۶ بیتی بین ۳۲۷۶۸- تا ۳۲۷۶۷
unsigned int	۲	برای سیستم ۱۶ بیتی بین ۰ تا ۶۵۵۳۵
short int	۲	۳۲۷۶۷ تا ۳۲۷۶۸
unsigned short int	۲	۰ تا ۶۵۵۳۵
long int	۴	۲۱۴۷۴۸۳۶۴۷ تا ۲۱۴۷۴۸۳۶۴۸
unsigned long int	۴	۰ تا ۴۲۹۴۹۶۷۲۹۵
float	۴	اعداد اعشاری کوچک
double	۸	اعداد اعشاری بزرگ
long double	۱۰	اعداد اعشاری خیلی بزرگ

**آرایه های یک بعدی :** با طور کلی آرایه ها نوعی پیشرفته تر از متغیر ها هستند که می توانند چندین مقدار را در خود ذخیره کنند .

در مورد آرایه ها در قسمت 6 بطور مفصل تری صحبت خواهیم کرد.

**ثابت ها :** همانطور که از نام آنها معلوم است ، مقادیری هستند ثابت که در برنامه برای راحتی بعضی کار ها از آنها استفاده می کنیم .  
برای تعریف ثابت ها از کلمه کلیدی `#define` استفاده می شود .

صورت کلی تعریف ثابت ها به صورت زیر است :

```
#define نام ثابت مقدار ثابت ;
```

مثال : در زیر ثابت `tel` با مقدار `110` تعیین شده است :

```
#define tel 110;
```

توجه داشته باشید که مقدار یک ثابت در طول برنامه تغییر نمی کند .

### سازمان برنامه ها در ++C :

مانند زبان C هر برنامه ی ++C در چند فایل گسترده می شود . نوعی از این فایل ها سرفایل نام دارند که دارای پسوند `.h` هستند و از آنها برای ذخیره ای اعلان ها استفاده می شود .  
سرفایل ها خود نیز دو گروهند :

1- بعضی از سرفایل ها در خود سیستم تعریف شده اند . مانند : `<iostream.h>`

2- عده ی دیگری از سرفایل ها توسط کاربر تعیین می شود یعنی هرگاه بخواهیم کد های داخل یک فایل را به کدهای برنامه بیفزاییم ، از آنها ی استفاده می کنیم .

سر فایل ها با استفاده از دستور پیش پردازنده ی `include` در فایل های مربوطه گنجانده می شوند .

**دستور include :** هر گاه بخواه کد های داخل یک فایل را به برنامه بیفزاییم ، از دستور اینکداد (`include`) استفاده می کنیم .

صورت های کلی استفاده از دستور `include` به صورت زیر است :

```
#include <Parham>

#include "Parham"
```

تفاوت این دو حالت در این است که :

در حال اول کامپایلر ( compiler ) یا همون چیزی که برنامه رو تجزیه و تحلیل کرده و به اجرا در میاره ( در دایرکتوری تعریف شده ی include ها دنبال فایل می گردد اما در حالت دوم کامپایلر در داخل دایرکتوری جاری دنبال فایل Header می گردد.  
(اگر چیز زیادی متوجه نشدید ، نگران نباشید ! مطمئن باشید در ادامه با برنامه های نمونه و آزمایشات مختلف متوجه این تفاوت ها خواهید شد .)

### دستورات ورودی و خروجی :

یکی از تفاوت های مهم C و ++C در همین قسمت دستورات ورودی و خروجی ( I/O ) می باشد که بجا اینکه مانند C از دستورات printf و scanf استفاده کند ، از دستورات cout (بخوانید : سی اوت ) و cin (بخوانید : سی این ) استفاده می کند .

ساختار کلی برای استفاده از دستورات ورودی در ++C بصورت زیر است :

```
cin>>value1>>value2>>value3...;
```

برای مثال در کد زیر کامپایلر با دیدن دستور cin منتظر ورود مقادیری مانند mambolearn و Ahmadzadeh می باشد :

```
#include <iostream.h>

int main()
{

    int mambolearn,Ahmadzadeh;
    cin>>mambolearn>>Ahmadzadeh;

    return 0;
}
```

ساختار کلی برای استفاده از دستورات خروجی در ++C بصورت زیر است :

```
cout<<value1<<value2<<value3...;
```

برای مثال در کد زیر کامپایلر با دیدن دستور cout مقادیری متغیری مانند Mahmoodi را چاپ خواهد کرد :

```
#include <iostream.h>

int main()
{

    int Mahmoodi=123;
    cout<<Mahmoodi;
}
```

توجه : در این ساختار ، برای مثال mahmoodi دو متغیر هستند که در واقع مقادیر داخل آنها چاپ می شود . یعنی :

```
123456789
```

مثال : ابتدا متغیر nomre را از نوع کاراکتری می سازیم و بعد محتوای آنرا در جمله ای به کار می بندیم . به دستورات زیر توجه کنید :

```
#include <iostream.h>

int main()
{
    int nomre;

    cout<<"Lotfan yek nomre vared
konid !! (az 20)";

    cin>>nomre;

    cout<<"Sadegh e Jedari is a
"<<nomre<<" boy !";

    return 0;
}
```

در این قسمت ابتدا یک متغیر با نام nomre ساخته ایم . بعد در خروجی چاپ کرده ایم که " لطفا یک نمره وارد کنید " . سپس نمره ی مربوط به او را در داخل متغیر nomre قرار داده ایم . و بعد از آن آن را همراه با یک متن چاپ می کنیم . برای مثال اگر نمره ی وارد شده 0 باشد ، در خروجی خواهیم داشت :

```
Sadegh e Jedari is a 0 boy !
```

شاید پرسید این عبارت آخریه چیه ؟ ( ; return 0 )

در اون بالا در تعریف main نوشتیم ( int main( ) این یعنی خروجی این تابع int هست و با دستور return 0 ، مقدار 0 رو به عنوان خروجی تابع در نظر گرفتیم . حالا اگر نخواهید دستور return رو به کار ببرید ، باید در تعریف main به جای نوشته قبلی بنویسید ( void main( ) . این قسمت مربوط به مبحث تابع هاست که فعلا لازم نیست این ها متوجه شوید ! فقط در همین حد بدانید چرا و برای چه این عبارت را نوشته ایم . )

### چند تا از علامات بدرد بخور:

علامت	توضیح
\n	رفتن به یک خط پایین تر : میتوان گفت که همان کار دکمه enter را در نرم افزار word انجام میدهد
//	توضیح برای کد مورد نظر فقط برای یک سطر : با استفاده از این می توانید مانند نمونه ی نشان داده شده عباراتی را در مقابل کد مورد نظر تان بنویسید تا در مشاهده های بعدی راحت تر و سریعتر کار بکنید . مثا اینکه این کدی که اینجا نوشته اید برای چه است و ...
/* */	توضیح برای کد مورد نظر به تعداد سطر های دلخواه : این هم نوعی از همان علامت بالاست که در توضیح های طولانی از آن استفاده می شود و طرز استفاده ی آن با قبلی فرق دارد.

برای متوجه شدن کاربرد دقیق هریک از این نشانه ها به کد زیر توجه فرمایید :

```
#include <iostream.h>
int main()
{
    int nomre; // here we made a variable !
    cout<<" Lotfan\n";
    cout<<" nomreya riazi e khod ra \n";
    cout<<" vared konid !! \n";
    cin>>nomre; /* here we will give the client a number that
    shows his or her mark in mathematics
    and we will use it in futur */
    cout<<"Your mark is not very bad ! : "<<nomre<<" \n";
    return 0;
}
```

در نهایت اگر نمره ی وارد شده 20 باشد ، در نهایت این چاپ خواهد شد :

```
Lotfan

nomreya riazi e khod ra

vared konid !!

20

Your mark is not very bad ! : 20
```

یک مثال دیگر :

```
#include <iostream.h>
int main()
{
    cout<<"Welcome to \n\nnC++ farsi
    \ne-learning!!\n";
    return 0;
}
```

نتیجه :

```
Welcome to  
  
C++farsi  
e-learning!!
```

## پایان قسمت اول !

نویسنده : دانیال خشابی

ویرایش و صحت مطالب : نوید مردوخ روحانی

[www.mrh.ir](http://www.mrh.ir)

[www.majidonline.com](http://www.majidonline.com)

کپی رایت :: تیر 1385

آخرین تغییرات : مهر 85

ارائه ی این مطلب فقط با ذکر منبع و دو سایت بالا مجاز است !