بنام خدا

آشنایی با زبان برنامه نویسی++C

قسمت اول : مبانی++C

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 گرایی 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	۱۲۸- تا ۱۲۷
signed char	۱	۱۲۸- تا ۱۲۷
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	۳۲۷۶۸- تا ۳۲۷۶۷
unsigned short int	۲	۰ تا ۶۵۵۳۵
long int	۴	۲۱۴۷۴۸۳۶۴۸- تا ۲۱۴۷۴۸۳۶۴۷
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 ذخیرهی 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 آنهای 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 (بخونید : 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 درواقع 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 نام وارد کنید " . سپس نمره ی مربوط به او را در داخل متغیرnomre 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:

توضیح	علامت
رفتن به یک خط پایین تر : میتوان گفت که همان کار دکمه enter را در نرم افزار word انجام میدهد	\n
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 نظرتان بنویسید تا در مشاهده های بعدی راحت تر و سریعتر کار بکنید . مثا اینکه این کدی که انجا 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 کابرد 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<<" nomreye 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 mathmatics
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

 nomreye 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\n\nC++ farsi
\ne-learning!!\n";
return 0;
}
```

عملگر های ریاضی در C++ :

عملگر های ریاضی در C++ به صورت زیر تعریف می می شوند :

علامت عملگر در زبان C++	نام عملگر
+	جمع
-	تفریق
*	ضرب
/	تقسیم
%	باقیمانده

بدیهی است که حاصل این عملگر ها می تواند صحیح یا اعشاری باشد .

در زبان C++ برای انجام عملیات جمع و تفریق با 1 دو عملگر دیگر نیز وجود دارد که هدف از ایجاد آنها عموم راحت تر کردن کار بوده است.

علامت عملگر در زبان C++	نام عملگر
++	جمع خود محتوای متغیر با 1
--	کم کردن 1 واحد از محتوای متغیر

نکته ی مهم در استفاده از این دو عملگر اخیر توجه به نقش های مختلف آن در اولویت های مختلف است .
یعنی اینکه a++ با ++a تفاوت خواهد داشت .و عملگر -- نیز به همین ترتیب است .

برای اینکه بهتر متوجه منظورم بشوید به مثال های زیر توجه بفرمایید :

```
#include <iostream.h>

int main()
   {
   int c;

c = 5;
cout << c << endl;     // print 5
cout << c++ << endl;   // print 5 then postincrement
cout << c << endl << endl; // print 6

c = 5;
cout << c << endl;    // print 5
cout << ++c << endl;  // pre increment then print 6
cout << c << endl;    // print 6

return 0;
   }
```

در خروجی برنامه ی فوق خواهیم داشت :

```
5
5
6

5
6
6
```

توجه : endl خط فرمان را سطر پایین تر می آورد . فرق آن با \n در این است که \nدر داخل " " نوشته می شود اما endl خودش به تنهایی نگاشته می شود .

مسئله ی 1 : برنامه ای بنویسید که دو عدد را گرفته و مجموع آنها را در خروجی چاپ کند .

```
// program for suming up twe numbers !
#include <iostream.h>
int main()
{
int num1,num2,sum;
cout<<"please enter the first integer number ! : " ;
cin>>num1;
cout<<"please enter the second integer number ! : ";
cin>>num2;
sum=num1+num2;
cout<<" The sum is "<<sum<<" \n";
return 0;
}
```

در بحث عملگر های ریاضی اولویت بعدی عملگر ها خود نیز مسئله ی مهمی به شمار می آید. اولویت عملگر ها به صورت زیر است :

عملگر ها	ترتیب ی اولویت
( ) پرانتز ها	1
* یا /	2
%	3
+ یا -	4

چند نمونه دیگر از عملگر های ریاضی که برای آسانی کار ارائه شده اند :

شکل دیگر عملگر	مثال	علامت عملگر
a=a+5	a+=5	+=
a=a-5	a-=5	-=
a=a*5	a*=5	*=
a=a/5	a*=5	/=
a=a%5	a%=5	%=

برای اینکه با طرز کار با این عمگر ها آشنا شوید ، به مثال زیر توجه فرمایید :

```
#include <iostream.h>

int main()
{
int a;

a=6;
a+=6;
cout<<a<<endl;
a=6;
a-=6;
cout<<a<<endl;
a=6;
a*=6;
cout<<a<<endl;
a=6;
a/=6;
cout<<a<<endl;
a=6;
a%=6;
cout<<a<<endl;

return 0;
}
```

مسئله ی 2 : برنامه ای بنویسید که حقوق پایه و تعداد فرزندان یک کار گر را از ورودی گرفته و حقوق کل وی را از فرمول زیر بدست آورد :

حقوق کل = حقوق پایه + فرزندان * 10

جواب :

```
#include <iostream.h>

int main()
   {
      int  child,salary,wholesalary;

      cout<<" Enter yor child's number: ";

      cin>>child;

      cout<<" Enter your salary :";

      cin>>salary;

      wholesalary=salary+10*child;

      cout<<"your whole salary is : "<<wholesalary<< endl;

return 0;
   }
```

آرایه های چند وجهی :

آرایه های چند بعدی ، نوع پیشرفته تری از آرایه ها هستند که می توانند اطلاعات بیشتری را در خود ذخیره کنند . و در عوض نیز کار های پیشرفته تری را انجام دهند . (فعلا تا همین کافی است ! در قسمت های بعد بیشتر درباره کاربرد های اینها آشناخواهید شد .)

ساختار کلی برای استفاده از آرایه های چند بعدی به صورت زیر است :

```
 Type Arayname
[size1][size2]...[sizeN];
```

برای مثال در در نمونه ی زیر آرایه ی kami دارای 2 بعد می باشد که دارای 2 سطر و ستون است . می خواهیم مقادیر اولیه ی این آرایه را بگیریم و 1 مقدار به آنها اضافه کنیم و بعد مقادیر آرایه ی kami را چاپ کنیم :

```
#include <iostream.h>

int main()
   {

int kami[2][2];

cin>>kami[1][1];
cin>>kami[2][1];
cin>>kami[1][2];
cin>>kami[2][2];

kami[1][1]++;
```

```
kami[2][1]++;
kami[1][2]++;
kami[2][2]++;

cout<<"kami [1][1]:
"<<kami[1][1]<<endl;
cout<<"kami [2][1]:
"<<kami[2][1]<<endl;
cout<<"kami [1][2]:
"<<kami[1][2]<<endl;
cout<<"kami [2][2]:
"<<kami[2][2]<<endl;

return 0;

   }
```

عملگر های منطقی :

در جدول زیر انواع عملگر های مقایسه ای یا منطقی را مشاهده می فرمایید:

نام عملگر	علامت عملگر در زبان C++
AND	&&
OR	||
NOT	!
کوچکتر	<
کوچکتر یا مساوی	<=
بزرگتر	>
بزرگتر یا مساوی	=>
مقایسه	==
نا مساوی	!=
انتصاب شرطی	?:

بیشترین استفاده ی از عملگر های منطقی یا مقایسه ای در ساختار های شرطی است . برای آشنایی با ساختار های شرطی به مطلب بعدی توجه فرمایید .

ساختار های تصمیم گیری در C++ :

1- دستور if-else: زمانی از این ساختار استفاده می شود که شرط ها کم باشد.
2- دستور switch-case: زمانی از این ساختار استفاده می شود که تعداد تصمیم گیری ها زیاد باشد .

ساختار شرطی if :

ساختار کلی شرطی if به صورت زیر است :

```
If (condition1)
{
دستورات قسمت اول ;
}
Else
{
دستورات قسمت دوم;
}
```

توجه کنید که ساختار بالا یک ساختار کلی می باشد و ممکن است در حالات شکل آن تغییر کند . مثل حالات زیر ، آنها را به خاطر بسپارید ! :

1- دستورات قسمت اول یا دستورات قسمت دوم یا هردو فقط شامل ی دستور باشند :

```
If (condition1)
دستور قسمت اول ;
Else
{
دستورات قسمت دوم;
}
```

```
If (condition1)
{
دستورات قسمت اول ;
}
Else
دستور قسمت دوم;
```

```
If (condition1)
دستور قسمت اول ;
Else
دستور قسمت دوم;
```

2- در بعضی مواقع استفاده از قسمت دوم این ساختار تصمیم گیری (else) لازم نیست . یعنی اینکه شما فقط قصد استفاده از قسمت ifرا دارید :

```
If (condition)
{
دستورات ;
}
```

توجه کنید که در صورتی هم که دستورات شما شامل فقط یک دستور باشد ، لازم نیست که از دو آکولاد استفاده کنید :

```
If (condition)
دستور ;
```

مسئله 3 : برنامه ای بنویسید که 2 عدد دریافت کند و بزرگترین آنها را بنویسد .
جواب :

```
#include <iostream.h>
int main()
   {
      int  num1,num2;
      cout<<" Enter your  first  number: ";
      cin>>num1;
      cout<<" Enter your second number : ";
      cin>>num2;

      if (num1>num2)
      cout<<num1<<" is bigger ! ";
      else
      cout<<num2<<" is bigger ! ";

return 0;
   }
```

مسئله 4 : برنامه ای بنویسید که یک عدد را از ورودی گرفته و مشخص کند که آن عدد زوج است یا فرد .

جواب :

```
#include <iostream.h>
int main()
   {
      int  num;
      int rest;

      cout<<" Enter your  number: ";
      cin>>num;
      rest=num % 2;


      if (rest!=0)
      cout<<num<<" is  fard(odd) ! ";
      else
      cout<<num<<" is  zoj(even) ! ";

return 0;
   }
```

جواب تمرینات شماره ی قبل :

تمرین 1 : برنامه ای را بنویسید که سه عدد را گرفته و برگترین آنها را تعیین کند . (فقط با دو if)

```
#include <iostream.h>
int main()
   {
      int  a,b,c,max;



      cout<<" Enter your  numbers ";
      cin>>a>>b>>c;
    max=a;
      if (b>max)
            max=b;
      if (c>max)
            max=c;
    cout<<" The max is " <<max<<endl;

return 0;
   }
```

تمرین 2 : برنامه ای بنویسید که سه عدد را گرفته و بزرگترین و کوچکترین آنها را تعیین کند . (فقط با سه if)

```
#include <iostream.h>
int main()
   {
      int  a,b,c,max,min;
      cout<<" Enter your  numbers ";
      cin>>a>>b>>c;
    max=a;
      min=a;
      if (b>a)
      {
            max=b; min=a;
      }
      else
      {
            max=a; min=b;
      }
      if (c>max)
            max=c;
      if (c<min)
            min=c;

    cout<<" The max is " <<max<<endl;
   cout<<" The min is " <<min<<endl;
return 0;
   }
```

تمرین 3: برنامه ای بنویسید که سه عدد را گرفته و آنها را به ترتیب بزرگتر تا کوچکتر مرتب کرده و در خروجی چاپ کند .

```
#include <iostream.h>
int main()
   {
   int a,k,b,c;
   cout<<"enter 3 numbers :  ";
   cin>>a>>b>>c;
   if ( b>a)
   {
         k=a; a=b; b=k;
   }
    if ( c>a)
   {
         k=a; a=c; c=k;
   }
    if ( c>b)
   {
         k=b; b=c; c=k;
   }
   cout<<"max number is : "<<a<<endl;
   cout<<"mid number is : "<<b<<endl;
   cout<<"min number is : "<<c<<endl;

return 0;
   }
```

ساختار شرطی swich-case:

زمان که تعداد شرط ها زیاد باشد از این ساختار استفاده خواهیم کرد .
ساختار کلی آن به شکل زیر است :

```
switch (variable)                          } 1
  {
        case 'value1-1':                   }
        case 'value2-1':                   }
        ...                                }
        case 'value N-1':                  } 2
                یک یا چند دستور;           }
                break;                     }

        case 'value1-2':                   }
        case 'value2-2':                   }
        ...                                }
        case 'value N-2':                  } 3
                یک یا چند دستور;           }
                  break;                   }

        ...

        case 'value1-N':                   }
        case 'value2-N':                   }
        ...                                }
        case 'value N-N':                  } 4
                یک یا چند دستور;           }
                break;                     }

       default:                            }
                دستور یا دستورات;          } 5
}
```

توضیح :
ساختار کلی دستور شرطی بدین صورت است که ابتدا یک متغیر را در نظر می گیرد = switch (variable) - (قسمت 1) ، سپس شرط می کند اگر مقدار این متغیر برای مثال value1-1 یا value1-2 ، value1-3 یا ... باشد ، یک یا چند دستور را اجرا کند (قسمت 2) . یا اگر مقدار این متغیر برای مثال value2-1 یا value2-2 ، value2-3 یا ... باشد ، یک یا چند دستور دیگر را اجرا کند (قسمت 3) . یا ... (قسمت 4) . که این قسمت ها توسط دستور break; از هم جدا می شوند . در نهایت یک قسمت داریم که اگر مقدار متغیر هیچ یک از مقادیر شرط شده نبود ، آن سری از دستورات را اجرا کند (قسمت 5) .

مسئله 8 : برنامه ای بنویسید که اگر کاربر هر یک از حروف A B C D را وارد کند بنویسد : Big character
اگر حروف a b c d را وارد کند بنویسد : little Character
اگر اعداد 1 تا 4 را وارد کند بنویسد : a figure !
اگر غیر از این حروف را وارد کند ، بنویسد : unknown character !

جواب :

```
#include <iostream.h>
int main()
   {
      char  a;
      cout<<" Enter your  selected character ! :  ";
      cin>>a;

switch (a)
{
      case 'A':
      case 'B':
      case 'C':
      case 'D':
            cout<<"Big character !  ";
            break;
      case '1':
      case '2':
      case '3':
      case '4':
              cout<<" a figure ! ";
            break;
      case 'a':
      case 'b':
      case 'c':
      case 'd':
              cout<<" little character  ! ";
            break;
              default:
            cout<<"unknown character ! ";
            break;
}

return 0;
   }
```

ساختار های حلقه ها در زبان C++ :

در زبان C++ سه نوع ساختار حلقه (گردش) وجود دارد :

1- for
2- do while
3- while

در ادامه به بررسی هر سه حلقه خواهیم پرداخت .

ساختار حلقه ی for :

ساختار کلی حلقه ی for به صورت زیر است :

```
for( مقدار تغییر متغیر ;مقدار نهایی ;مقدار اولیه )
{
        دستور یا دستورات
}
```

توجه : در صورت مجموعه دستورات شما فقط شامل یک دستور باشد می توانید از آکولاد ها صرف نظر کنید .

مسئله 9 : برنامه ای بنویسید که تا اعداد طبیعی کوچکتر از 100 را چاپ کند .

```
#include <iostream.h>
int main()
   {

      cout<<"figures which are less than 100 : \n";

      for(int i=0; i<100; i++)
      {
            cout<<i<<endl;
      }
return 0;
   }
```

مسئله ی 10 : برنامه ای بنویسید که مجموع اعداد 1 تا 99 را بنویسد .

```
#include <iostream.h>
int main()
   {

      int s;
      cout<<"majmo is : ";

      for(int i=0; i<100; i++)
      {
            s=s+i;
      }

      cout<<s;

return 0;
   }
```

مسئله ی 11 : برنامه ای بنویسید که اعداد زوج بین 100 و 1000 را چاپ کند .

```
#include <iostream.h>
int main()
   {

      int s,k;
      cout<<"even numbers between 100 and 1000 are : ";

      for(int i=100; i<1001; i++)
      {
            k=i%2;
            if (k==0)
            cout<<i<<" ";
      }

return 0;
   }
```

مسئله 12 : برنامه ای بنویسید که حاصلضرب اعداد 5 تا 10 را در خروجی چاپ کند .

```
#include <iostream.h>
int main()
   {
      long int  p;

      p=1;
      for(int i=5; i<10; i++)
      {
            p=p*i;

      }

       cout<<" zarbe adade 5 ta 10 : " <<p;


return 0;
   }
```

ساختار حلقه ی while :

```
while(   شرط اجرای حلقه  )
{
 دستورات برنامه
}
```

مسئله ی 13 :برنامه ای بنویسید که دنباله ی زیر را تا عدد 9999 چاپ کند :
10 30 50 70 90 ...

```
#include <iostream.h>
int main()
   {
      long int a;

      a=10;
      while (a<9999){
      cout<<a<<"    ";
      a+=20;
      }

return 0;
   }
```

مسئله ی 14 : برنامه ای بنویسید که تا تعدادی عدد مثبت را گرفته و مجموع آنها را حساب کند . (شرط پایان کار ، وارد کردن عدد صفر از طرف کاربر خواهد بود) .

```
#include <iostream.h>
int main()
   {
      long int s,a;
      cout<<"enter your numbers :  "<<endl;
      cin>>a;
      s=a;
      while (a>0){
      cin>>a;
      s+=a;
      }
      cout<<"Sum is  "<<s;


return 0;
   }
```

ساختار حلقه ی do while:

```
do{
 دستورات برنامه
}while( شرط اجرای کد حلقه )
```

فرق اساس حلقه ی do while با while در این است که در حلقه ی فوق ، دستورات برای یک بار بدون توجه به شرط برنامه اجرا می شوند .

برای مثال مسئله 14 را یکبار دیگر با حلقه ی do while می نویسیم :

```
#include <iostream.h>
int main()
   {
      int s,a;
      cout<<"enter your numbers :  "<<endl;
      s=0;

      do{
      cin>>a;
      s=a+s;
      } while (a>0);

      cout<<"Sum is  "<<s;


return 0;
   }
```

مسئله ی 15 : برنامه ای بنویسید که تعدادی عدد مثبت را از ورودی گرفته و بزرگترین و کوچکترین آنها را تعیین کند . (شرط پایان کار وارد کردن عدد صفر است)

```
#include <iostream.h>
int main()
   {
      int s,max,min,a;
      cout<<"enter your numbers :  "<<endl;
      cin>>a;
      max=a; min=a;
      while (a>0){
            if (a>max)
            max=a;
            if (a<min)
            min=a;
      cin>>a;
      }

      cout<<"max is  "<<max;
      cout<<"min is  "<<min;

return 0;
   }
```

تمرینات :

1- (مسئله ی 16): برنامه ای بنویسید که دستور فاکتوریل را انجام دهد . یعنی اینکه عددی را از ورودی گرفته و فاکتوریل آنرا را حساب کند .
راهنمایی :

$a! = 1*2*3*...*a$ و $0! = 1$

2- (مسئله ی 17):برنامه ای بنویسید که 10 عدد از ورودی گرفته و بزرگترین آنها را مشخص کند .

3- (مسئله ی 18): برنامه ای بنویسید که عددی را گرفته و مشخص کند که آیا عدد اول است یا نه .

4- (مسئله ی 19): برنامه ای بنویسید که عددی را از ورودی گرفته و مشخص کند که عدد گرفته شده تام است یا نه .
(راهنمایی : عدد تام عددی است که مجموع مقسوم علیه های کوچکتر از خودش ، برابر خودش باشد)
5- (مسئله ی 20): برنامه ای بنویسید که اعداد اول 1 تا 5000 را چاپ کند .
6- (مسئله ی 21):برنامه ای بنویسید که عددی را گرفته و مقسوم علیه های آنرا چاپ کند . (می توانید برنامه را گسترش دهید تا تعداد و مجموع مقسوم علیه ها را هم چاپ کند .)
7- (مسئله ی 22):برنامه ای بنویسید که مجموع مضارب 5 را بین 1 و 100 چاپ کند .
8- (مسئله ی 23):برنامه ای بنویسید که دو عدد را از ورودی بگیرد و اعداد بین آنها را چاپ کند .
9- (مسئله ی 24):برنامه ای بنویسید که تعداد مضارب 7 و 5 را در بازه ی بین 1 تا 10000 چاپ کند .
(توضیح : اعداد مورد نظر هم باید مضرب 7 باشد و هم مضرب 5)
10-(مسئله ی 25):برنامه ای بنویسید که 10 عدد را گرفته و میانگین و حاصلجمع آنها ر ا به ما بدهد .

11- (مسئله ی 26):دنباله ی فیبوناچی به صورت زیر است :

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

این دنباله به صورت زیر تعریف می شود :

$F_1=1$	$F_2=1$	$F_n = F_{n-1}+F_{n-2} \qquad n \geq 3$

برنامه ای بنویسید که با دریافت مقدار صحیح n ، مقدار F_n را برای دنباله ی فیبوناچی چاپ کند .

12- (مسئله ی 27) : برنامه ای بنویسید که تعدادی عدد گرفته و مجموع مربعات آن را حساب کند .