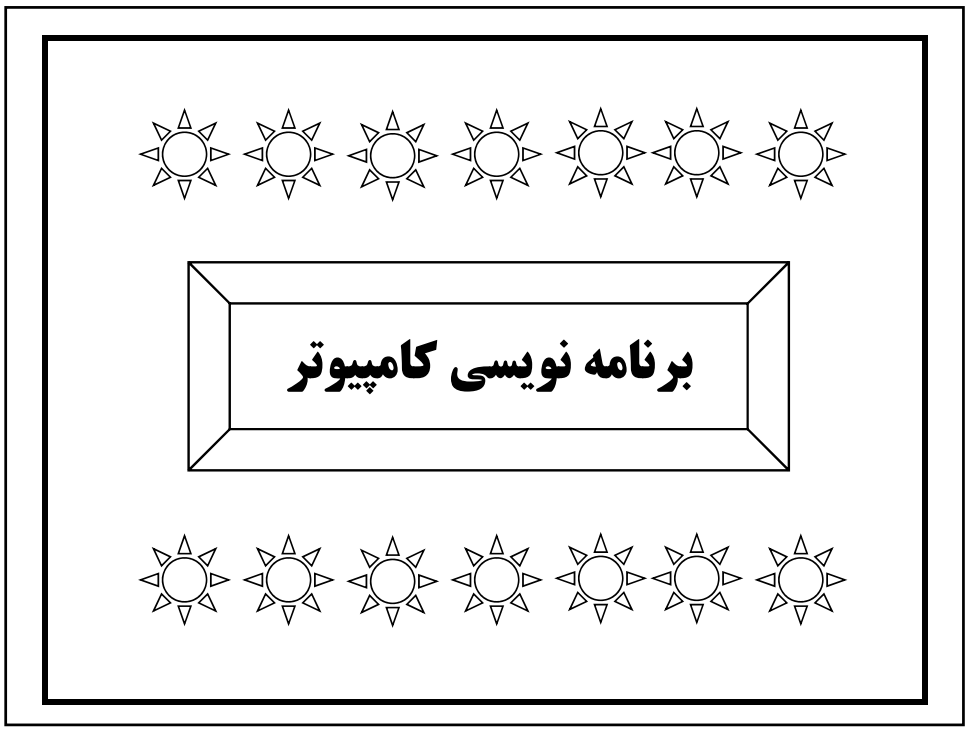


یک برنامه ساده

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

int main()
{
cout<<"Hello C++";
return 0;
}
```

یک برنامه ساده

```
/*
A simple program.
This program contains all of the
basic elements
*/

#include "stdafx.h"
#include <iostream>
using namespace std;

int main()
//main is where program execution begins.
{
cout<<"Hello C++";
return 0;
}
```

یک برنامه ساده

```
/*
A simple program.
This program contains all of the
basic elements
*/

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

int main()
//main is where program execution begins.
{
cout<<"Hello C++";
getch();
return 0;
}
```

توضیح Comment

۱–Multiline comment : توضیحات چند خطی نامیده می‌شوند که با یک /* شروع و با */ به پایان می‌رسد.

۲– Single-line comment: توضیح یک خطی نامیده می‌شود که با // شروع شده و در همان خط پایان می‌یابد.

محتویات توضیح توسط کامپایلر نادیده گرفته می‌شود.

main( )

۱- همه برنامه‌ها در C++ ترکیبی از یک یا چند تابع می‌باشند.
۲- تنها تابعی که تمام برنامه‌ها دارند تابع main می‌باشد.
۳- main جایی است که اجرای برنامه از آن‌جا شروع می‌شود.

header files

۱- زبان C++ چندین فایل تعریف می‌کند که فایل‌های سرآمد یا header files می‌گویند.

۲- هر فایل سرآمد شامل اطلاعاتی است که برای برنامه ضروری می‌باشند.

۳- iostream برای پشتیبانی از سیستم ورودی و خروجی می‌باشد. برای استفاده از cout از این فایل سرآمد استفاده می‌شود. محتویات فایل iostream.h به برنامه اضافه می شود.

۴- conio.h برای استفاده از تابع getch مورد نیاز می‌باشد.

cout

یک شناسه از پیش تعریف شده است که مخفف console output می‌باشد و برای نمایش روی صفحه نمایش به کار می‌رود.

- ✓ انتهای تمام دستورات در C++ به سمی‌کولون (;) ختم می‌شود.

stdafx.h

یکprecompiled header file است که شامل فایل های پیش نیاز پروژه های C++ در محیط visual c++ است.

getch()

این تابع باعث می‌شود تا زمانیکه از ورودی کاراکتری دریافت نشده است اجرای برنامه متوقف گردد.

return 0

با این دستور تابع main پایان می‌یابد و مقدار صفر به پروسه فراخواننده main که معمولا سیستم‌عامل است بازگردانده می‌شود.

مقدار صفر نشان دهنده آن است که برنامه به طور عادی پایان یافته است. مقادیر دیگر نشان‌دنده آن است که برنامه به علت خطایی پایان یافته است. return یکی از کلیدواژه‌های C++ می‌باشد که برای بازگرداندن مقداری از یک تابع به کار می‌رود.

using namespace std

حوزه نام std بصورت عمومی تعریف می شود. در صورتی که این عبارت استفاده نشود باید قبل از تمام دستورات ورودی و خروجی std:: اضافه گردد.

دومین برنامه ساده

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

using namespace std;
int main()
{
int num;
num=99;
cout<<"The value is:";
cout<<num;
getch();
return 0;
}
```

تعریف متغیر

- متغیر محل نامگذاری شده‌ای از حافظه است که می‌توان مقداری در آن قرار داد.
- محتوای یک متغیر قابل تغییر است و ثابت نیست.
- برای تعریف متغیر ابتدا نوع متغیر و سپس نام دلخواهی برای آن تعیین می‌کنیم.

```
type   variable;
```

- با علامت = مقدار ۹۹ را داخل متغیر num قرار داده می‌شود.
- می‌توان چند متغیر را همزمان تعریف نمود.

```
int a,b;
```

نوع داده‌های اصلی

کلمه کلیدی	دامنه	شرح
bool	مقدار درست یا نادرست	true و false
char	۱۲۷ تا ۱۲۸−	کاراکترهای ۸ بیتی 'A'
int	۳۲۷۶۷ تا ۳۲۷۶۸−	عدد صحیح
float	۴/۳E+۳۸ تا ۴/۳E−۳۸	مقدار اعشاری
double	۷/۱E+۳۰۸ تا ۷/۱E−۳۰۸	مقدار اعشاری با دقت مضاعف
void	اعمال نمی‌شود	بیانگر یک عبارت بدون مقدار
wchar_t	۰ تا ۶۵۵۳۵	کاراکترهای عریض (زبان چینی)

سومین برنامه ساده

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

using namespace std;
int main()
{
cout<<"First line.\n";
cout<<"Second line.\n";
cout<<"Third line.\n";
getch();
return 0;
}
```

کاراکتر خط جدید (newline)

خروجی برنامه

First line.
Second line.
Third line.

سومین برنامه ساده

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

using namespace std;
int main()
{
cout<<"One\nTwo\nThree";
getch();
return 0;
}
```

خروجی برنامه

One
Two
Three

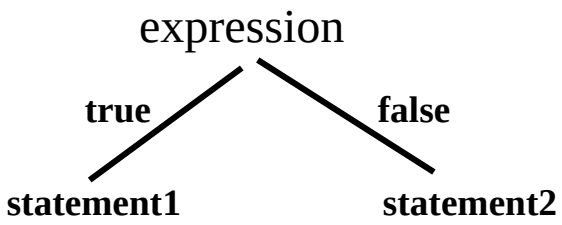
دریافت اطلاعات از کاربر

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

using namespace std;
int main()
{
int a,b;
cin>>a>>b;
cout<<"The value a is:"<<a<<"\n";
cout<<"The value b is:"<<b<<"\n";
getch();
return 0;
}
```

به جای "\n" می توان از endl استفاده کرد.

دستور if

```
if (expression) statement1;
      else statement2;
```



استفاده از if

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

using namespace std;
int main()
{
float num1,num2;
int choice;
cout<<"Enter values:";
cin>> num1>>num2>>choice;
if (choice==1) cout<<num1+num2;
if (choice==2) cout<<num1-num2;
getch();
return 0;
}
```

استفاده از else

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

using namespace std;
int main()
{
float num1,num2;
int choice;
cout<<"Enter values:";
cin>> num1>>num2>>choice;
if (choice==1) cout<<num1+num2;
else cout<<num1-num2;
getch();
return 0;
}
```

استفاده از بلوک کد

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

using namespace std;
int main()
{
float num1,num2;
int choice;
cout<<"Enter values:";
cin>>choice;
if (choice==1) {
cin>> num1>>num2;
cout<<num1+num2;
}
getch();
return 0;
}
```

بزرگترین و کوچکترین عدد

```
#include "stdafx.h"
#include <iostream>

using namespace std;

int main()
{
    cout<<"Minimum int, Maximum int\n";
    cout<<INT_MIN<<", "<<INT_MAX <<"\n";
    cout<<"unsigned int \n"<<UINT_MAX <<"\n";
    system("pause");
    return 0;
}
```

Output

```
Minimum int, Maximum int
-2147483648, 2147483647
unsigned int
4294967295
Press any key to continue . . .
```

System("pause") به جای getch() استفاده شده است.

زوج / فرد

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
    int num;
    cout<<"Enter a number:";
    cin>>num;
    if(num%2==0)
        cout<<"The number is even.";
    else
        cout<<"The number is odd.";
    system("pause");
    return 0;
}
```

Output

```
Enter a number:11
The number is odd.Press any key to continue . . .
```

دو عدد و یک عملگر

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main(){
    float a,b;
    char op;
    cout<<"Enter numbers:\n";
    cin>> a >> b;
    cout<<"Enter operation:\n";
    cin>>op;
    if(op=='+')
        cout<<a + b;
    if(op=='-')
        cout<<a - b;
    if(op=='*')
        cout<<a * b;
    if(op=='/')
        cout<<a / b;
    system("pause");
    return 0;}
```

Output

```
Enter numbers:
6
4
Enter operation:
/
1.5Press any key to continue . . .
```

عملگر sizeof

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

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

عملگرهای منطقی

عمل	عملگر
AND	&&
OR	||
NOT	!

دستور switch

شکل کلی دستور به صورت زیر است:

```
switch(expression){
      case(val1):
      instructions
      break;
      case(val2):
      instructions
      break;
      default:
      instructions
}
```

به کار بردن break

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

using namespace std;
int main()
{
int num;
cin>> num;
switch (num) {
    case 1:
    cout<< "First Case"; break;
    case 2:
    cout<< "Second Case";break;
    case 3:
    cout<< "Third Case"; break;
    case 4:
    cout<< "Forth Case"; break;
    case 5:
    cout<< "Fifth Case"; break;
    default:
    cout<<"Nothing";
    }
getch();
return 0;
}
```

استفاده از عملگرهای افزایشی و کاهشی

i=i+1;	=	i++;
i=i-1;	=	i--;

استفاده از Assignment operators

*= += -= /=

x=x + y <= x+=y

استفاده از عملگرهای افزایشی و کاهشی

می‌توان عملگرها را پیش از متغیرها هم به کار برد. اما معنای متفاوتی دارند.

j=i++; ⇨ ابتدا مقدار i به j نسبت داده شده و سپس i یک واحد اضافه می‌شود.

j=++i; ⇨ ابتدا مقدار i یک واحد اضافه می‌شود و سپس مقدار i به j نسبت داده می‌شود.

مثال

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
int i, j, k, l;
i=15;
j=20;
k=i++;
l=++j;
cout<<i<<"\n"<<j<<"\n"<<k<<"\n"<< l<<"\n";
system("pause");
return 0;
}
```

Output

```
16
21
15
21
Press any key to continue . . .
```

مثال

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
int i, j, k, l;
i=15;
j=20;
k=i--;
l=--j;
cout<<i<<"\n"<<j<<"\n"<<k<<"\n"<<l<<"\n";
system("pause");
return 0;
}
```

Output

```
14
19
15
19
Press any key to continue . . .
```

:: Scope Resolution operator

```
#include "stdafx.h"
#include <iostream>

using namespace std;

int x = 10;
int main()
{
    int x = 14;
    x+=5;
    cout<<x<<"\n";
    {
        int x = 20;
        cout<<x<<"\n";
        cout<<::x<<"\n";
    }
    system("pause");
    return 0;
}
```

Output

```
19
20
10
Press any key to continue . . .
```

حلقه for

حالت کلی این دستور به صورت زیر می‌باشد.

for (initialization; expression; increment) statement;

- **مقداردهی اولیه متغیر کنترل کننده حلقه**
- **شرط حلقه**
- **افزایش یا کاهش مقدار متغیر کنترل کننده حلقه**

حلقه for

حالت کلی این دستور به صورت زیر می‌باشد.

for (initialization; expression; increment) statement;

- **initialization**: تنها یکبار و پیش از شروع حلقه اجرا می‌گردد.
- **expression**: در آغاز هر تکرار انجام می‌شود.
- **increment**: پس از اجرای بدنه حلقه، اجرا می‌شود.

برنامه ای بنویسید که اعداد بین ۱ تا ۱۰ را نمایش دهد.

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
int num;
for (num=1; num<11; num= num+1)
    cout<< num<<" ";
system("pause");
return 0;
}
```

Output

```
1 2 3 4 5 6 7 8 9 10 Press any key to continue . . .
```

حلقه for

کاهش و افزایش متغیر کنترل کننده می‌تواند بیش از یک واحد باشد.

```
for(num=10; num>0; num= num-4)
```

```
for(num=0; num<11; num= num+4)
```

عدد صحیح و مثبت n را دریافت کرده فاکتوریل آنرا محاسبه ونمایش دهد.

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
int  n, i ;
long fact = 1 ;
cout << "Enter a positive integer number\n";
cin >> n;
for( i=1; i<=n; ++i)
    fact *= i;
cout << fact<<"\n" ;
system("pause");
return 0;
}
```

Output

```
Enter a positive integer number
5
120
Press any key to continue . . .
```

دوعدد را دریافت کرده و عدد اول را به توان عدد دوم برساند و نتیجه را نمایش دهد. x^y

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
int x,y,p=1;
cin>>x>>y;
for(int n=1;n<=y;n++)
    {
        p=p*x;
    }
cout<<"Power is:"<<p<<"\n";
system("pause");
return 0;
}
```

Output

```
2
3
Power is:8
Press any key to continue . . .
```

به کار بردن for

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
int num;
for (num=11; num<11; num= num+1)
    cout<< num<<" ";
system("pause");
return 0;
}
```

Output

?

به کار بردن بلوک کد در for

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
int num, sum, prod;
sum=0;
prod=1;
for (num=1; num<11; num= num+1){
    sum= sum + num;
    prod= prod * num;
}
cout<< "product:"<< prod<< "\tsum:"<< sum <<"\n";
system("pause");
return 0;
}
```

Output

```
product:3628800 sum:55
Press any key to continue . . .
```

کاهش به جای افزایش در for

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
int num, sum, prod;
sum=0;
prod=1;
for (num=10; num>0; num= num-1){
    sum= sum + num;
    prod= prod * num;
}
cout<< "product:"<< prod<< "\tsum:"<< sum <<"\n";
system("pause");
return 0;
}
```

Output

```
product:3628800 sum:55
Press any key to continue . . .
```

حلقه های تو در تو

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

int main()
{
    system("color 5");
    for(int i=1; i<=10; i++)
        for(int j=1; j<=10; j++)
            cout<<i*j<<"\t";
        cout<<"\n";

    system("pause");
    return 0;
}
```

رنگی کردن خروجی

Output

1	2	3	4	5	6	7	8	9	10
2	4	6	8	10	12	14	16	18	20
3	6	9	12	15	18	21	24	27	30
4	8	12	16	20	24	28	32	36	40
5	10	15	20	25	30	35	40	45	50
6	12	18	24	30	36	42	48	54	60
7	14	21	28	35	42	49	56	63	70
8	16	24	32	40	48	56	64	72	80
9	18	27	36	45	54	63	72	81	90
10	20	30	40	50	60	70	80	90	100

Press any key to continue . . .

مقادیر منطقی در زبان C

==0 => false

!=0 => true

استفاده از مقادیر منطقی

```
#include "stdafx.h"
#include <iostream>
using namespace std;
int main()
{
bool p,q;
p=true;
q=true;
cout<<"p=true,"<<"q=true"<<"\t";
cout<<"(p&&q):"<<(p&&q)<<" ";
cout<<"(p||q):"<<(p||q)<<"\n";
p=true;
q=false;
cout<<"p=true,"<<"q=false"<<"\t";
cout<<"(p&&q):"<<(p&&q)<<" ";
cout<<"(p||q):"<<(p||q)<<"\n";
p=false;
q=true;
cout<<"p=false,"<<"q=true"<<"\t";
cout<<"(p&&q):"<<(p&&q)<<" ";
cout<<"(p||q):"<<(p||q)<<"\n";
p=false;
q=false;
cout<<"p=false,"<<"q=false"<<"\t";
cout<<"(p&&q):"<<(p&&q)<<" ";
cout<<"(p||q):"<<(p||q)<<"\n";
system("pause");
return 0;
}
```

Output

```
p=true,q=true   (p&&q):1 (p||q):1
p=true,q=false  (p&&q):0 (p||q):1
p=false,q=true  (p&&q):0 (p||q):1
p=false,q=false (p&&q):0 (p||q):0
Press any key to continue . . .
```

عدد دریافتی اول است؟

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
    int num, i, is_prime;
    cin>>num;
    is_prime=1;
    for (i=2; i<=num/2; i++)
        if(!(num%i)) is_prime=0;
    if(is_prime) cout<<"The number is prime.";
    else cout<<"The number is not prime.";
    system("pause");
    return 0;
}
```

فرض می‌شود عدد اول است.

حلقه while

حالت کلی این دستور به صورت زیر می‌باشد.

while (expression) statement;

شرط حلقه

تا زمانیکه شرط برقرار باشد دستورات اجرا می‌شود.
شرط حلقه در ابتدای حلقه کنترل می‌شود.

اعداد فرد بین num و صفر

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
    int num;
    cin>> num;
    while(num) {
    if (num%2) cout<<num<< " ";
    num--;
    }
    system("pause");
    return 0;
}
```

Output
5
5 3 1 Press any key to continue . . .

حلقه do

حالت کلی این دستور به صورت زیر می‌باشد.

```
do {
        statements
} while (expression);
```

شرط حلقه

تا زمانیکه شرط برقرار باشد دستورات اجرا می‌شود.
شرط حلقه در انتهای حلقه کنترل می‌شود.
حلقه حداقل یکبار اجرا خواهد شد.

استفاده از break برای خروج از حلقه

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
    int i;
    for(i=1; i<100;i++){
        cout<< i<< " ";
    if(i==10) break;
    }
    system("pause");
    return 0;
}
```

Output
1 2 3 4 5 6 7 8 9 10 Press any key to continue . . .

استفاده از continue برای اجرای دور بعدی حلقه

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
    int x;
    for(x=1; x<100;x++){
         continue;
        cout<< x;
    }
    system("pause");
    return 0;
}
```

Output
Press any key to continue . . .

حلقه goto

حالت کلی این دستور به صورت زیر می‌باشد.

goto label;

label:

برچسب

با دیدن goto به خطی می‌رود که برچسب یکسانی دارد.

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
    int i;
    i=1;
    again:
        cout<<i<<" ";
        i++;
    if(i<10) goto again;
    system("pause");
    return 0;
}
```

Output
1 2 3 4 5 6 7 8 9 Press any key to continue . . .

نگاهی دقیق‌تر به نوع داده‌ها

با استفاده از توصیف‌کننده‌های نوع (type modifiers) می‌توان نوع داده‌های char, int, float, double را تعدیل کرد. توصیف‌کننده‌های نوع عبارتند از :

signed, unsigned, long, short

توصیف کننده نوع پیش از نوع داده آورده می‌شود. long int i;

signed (علامت‌دار)

signed را می‌توان در مورد انواع int و char به‌کار برد.

استفاده از signed برای اعداد صحیح کار زایدی است زیرا به طور پیش فرض اعداد صحیح علامت‌دار هستند.

بسته به نوع پیاده‌سازی یک char می‌تواند علامت‌دار یا بدون علامت باشد. در صورتی که بدون علامت باشد اعداد مثبت بین ۰ تا ۲۵۵ را نگاه می‌دارد. درصورتی که به عنوان signed باشد می‌تواند اعدادی در دامنه ۱۲۸- تا ۱۲۷ را نگاه می‌دارد. signed char x;

long

long را می‌توان در مورد نوع داده‌های int و double به‌کار برد.

long بر روی int طول آن را بر حسب بیت دوبرابر می‌کند. اگر در محیطی طول int ۱۶ بیت باشد با استفاده از long طول آن ۳۲ بیت می‌شود.

long بر روی double طول double را نیز بر حسب بیت دوبرابر می‌کند.

long int g;

short

short را می‌توان تنها در مورد نوع داده int به‌کار برد.

short بر روی int طول آن را بر حسب بیت نصف می‌کند. اگر در محیطی طول int ۳۲ بیت باشد با استفاده از short طول آن ۱۶ بیت می‌شود.

unsigned (بدون علامت)

unsigned را می‌توان در مورد نوع داده‌های int و char به‌کار برد.
این توصیف‌کننده را همراه توصیف‌کننده‌های short و long نیز می‌توان مورد استفاده قرار داد.

signed & unsigned

۱- تفاوت اعداد صحیح علامت‌دار در نحوه تفسیر بیت با مرحله بالاتر می‌باشد.
۲- اگر عدد صحیح بدون علامت باشد از تمامی بیت‌ها برای نگه‌داری مقادیر استفاده می‌شود.
۳- اگر عدد صحیح علامت‌دار باشد در آن صورت کامپایلر کدی تولید می‌کند که فرض می‌نماید از بیت با مرتبه بالاتر به عنوان علامت استفاده می‌شود. اگر ۰ باشد عدد مثبت و اگر ۱ باشد عدد منفی است.
۴- عموما اعداد منفی با روش two's complement (مکمل دو) بیان می‌شوند.

ترکیبات ممکن از توصیف‌کننده‌ها

نوع	بیت	دامنه
char	۸	۱۲۸– تا ۱۲۷
unsigned char	۸	۰ تا ۲۵۵
signed char	۸	۱۲۸– تا ۱۲۷
int	۱۶	۳۲۷۶۸– تا ۳۲۷۶۷
unsigned int	۱۶	۰ تا ۶۵۵۳۵
signed int	۱۶	۳۲۷۶۸– تا ۳۲۷۶۷
short int	۱۶	مثل int
unsigned short	۱۶	مثل unsigned int
signed short int	۱۶	مثل short int
unsigned long	۳۲	۰ تا ۴۲۹۴۹۶۷۲۹۵
long	۳۲	۲۱۴۷۴۸۳۶۴۸ – تا ۲۱۴۷۴۸۳۶۴۷

ترکیبات ممکن از توصیف‌کننده‌ها

نوع	عرض
long int	۳۲
unsigned long	۳۲
signed long int	۳۲
float	۳۲
double	۶۴
long double	۸۰

نکته
ممکن است در محیط کاری شما long و short اصلا تاثیری نداشته باشد. بستگی به کامپایلر دارد.

آرایه‌ها و رشته‌ها

یک آرایه مجموعه‌ای از متغیرهای به‌هم مرتبط است که به وسیله یک نام مشترک مشخص می‌شود.

آرایه یک بعدی فهرستی از متغیرهایی از یک نوع واحد است که با استفاده از یک نام مشترک به همه آنها مراجعه می‌شود.

به هر متغیر یک آرایه یک عنصر آرایه گفته می‌شود.

حالت کلی آرایه‌ها به صورت زیر می‌باشد:

```
type var_name[size];
```

برای دسترسی به هر عنصر آرایه باید از شماره آن عنصر به عنوان index استفاده نمود.

آرایه‌ها و رشته‌ها

اندیس تمام آرایه‌ها از صفر شروع می‌شود و تا size-1 ادامه دارد.

```
int d[20];
d[0]=100;
```

مقدار عنصر صفر آرایه را برابر با ۱۰۰ قرار می‌دهد.

برای مقداردهی آرایه‌ها نیاز به حلقه‌های for داریم. مقادیر عناصر را باید تک به تک قرار داد.

مشکلی که وجود دارد این است که بر روی ایندکس آرایه‌ها عمل بررسی دامنه (bound checking) صورت نمی‌پذیرد.

به عنوان برنامه‌نویس خود باید مراقب این مساله باشید.

مثال:

```
int t[20];
t[25]=14;
```

برنامه‌اي بنويسيد كه 30 عدد صحيح مثبت را دريافت كرده و max آنها را بيابيد، نمايش دهيد.

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
    int i, max=0;
    int n[30];
    for(i=0; i<30;i++)
    {
        cin>>n[i];
        if(n[i]>max) max=n[i];
    }

    cout<<"Maximum is:"<<max;
    system("pause");
    return 0;
}
```

در زمان معرفی یک متغیر می‌توان به آن مقدار اولیه داد.

برنامه‌اي بنويسيد كه 50 عدد صحيح مثبت را دريافت كرده و آنها را به ترتيب عكس دريافت نمايش دهيد.

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
    int i;
    int n[50];
    for(i=0; i<50;i++)
        cin>>n[i];

    for(i=49; i>=0;i--)
        cout<<n[i]<<"\t";
    system("pause");
    return 0;
}
```

رشته‌ها (strings)

متداول‌ترین کاربرد آرایه‌های یک بعدی رشته‌ها می‌باشند.

هر رشته به صورت یک آرایه کاراکتری ختم شده به تهی null-terminated تعریف می‌شود.

آرایه‌ای که قرار است یک رشته را نگه دارد باید به کاراکتر تهی ختم شود به این معنی که باید یک بایت بزرگتر در نظر گرفته شود تا کاراکتر تهی در آن قرار گیرد. مقدار کاراکتر تهی صفر است.

```
char name[11];
```

حداکثر ۱۰ کاراکتر را نگه می‌دارد.

هر ثابت رشته‌ای نیز به یک کاراکتر تهی ختم می‌شود که کامپایلر به طور اتوماتیک آن را به انتهای رشته اضافه می‌کند.

چند تابع رشته‌ای استاندارد

```
strcpy(to,from);
```
این تابع رشته موجود در from رابه to کپی می‌کند. محتوای from تغییری پیدا نمی‌کند.

```
char str[80];
strcpy(str,"hello");
cout<<str;
```

این تابع دامنه آرایه مقصد را چک نمی‌کند و باید خود اندازه آن را بررسی کنید. (به همراه کاراکتر ختم‌کننده تهی null)

```
strcpy(str,"");
```
رشته‌ای با طول صفر ایجاد می‌کند. به چنین رشته‌ای رشته تهی (null string) می‌گویند.

```
strcat(to,from);
```
این تابع رشته موجود در from رابه to می‌چسباند . محتوای from تغییری پیدا نمی‌کند.

چند تابع رشته‌ای استاندارد

```
char str[80];
strcpy(str,"hello");
strcat(str,"Dear");
cout<<str;
```
این تابع دامنه آرایه مقصد را چک نمی‌کند و باید اندازه آن را بررسی کنید.

```
strcmp(s1,s2);
```
این تابع دو رشته را مقایسه می‌کند
اگر دو رشته یکسان باشد، تابع مقدار صفر می‌گیرد.
اگر s1<s2، مقداری کوچکتر از صفر برمی‌گرداند.
اگر s1>s2، مقداری بزرگتر از صفر برمی‌گرداند.

```
cout<<strcmp("one","one");
```

```
strlen(str);
```
این تابع طول یک رشته را برحسب تعداد کاراکترهای آن باز می‌گرداند.
کاراکتر ختم کننده تهی را به حساب نمی‌آورد.

مثال

```
#include "stdafx.h"
#include <iostream>

using namespace std;
int main()
{
    char str1[80], str2[80];
    int i;
    cout<<"Enter two strings.\n";
    cin>>str1>>str2;
    cout<<"length of str1 is:"<<strlen(str1)<<"\n";
    cout<<"length of str2 is:"<<strlen(str2)<<"\n";
    i=strcmp(str1,str2);
    if(!i) cout<<"the strings are equal";
    if (strlen(str1)+strlen(str2)<80){
        strcat(str1, str2);
        cout<<str1<<"\n";}
    strcpy(str1,str2);
    cout<<str1<<"\n";
    system("pause");
    return 0;
}
```

toupper() , tolower()

int toupper(int ch); کاراکتر ورودی را به معادل بزرگ خود تبدیل می‌کند.

int tolower(int ch); کاراکتر ورودی را به معادل کوچک خود تبدیل می‌کند.

باید از header file ، ctype.h استفاده نمود.

یکی از کاربردهای متداول آن پشتیبانی از interface است.

به این معنی که در دریافت ورودی از کاربر مفید است.

مثال

```
#include "stdafx.h"
#include <iostream>
using namespace std;
int main()
{
    char command[80];
    int i, j;
    for(;;){
        cout<<"operation:add,sub,mul,div,mod,quit\n";
        cin>>command;
        for(i=0;i<strlen(command);i++)
            command[i]=tolower(command[i]);
        if(!strcmp(command, "quit"))
           break;
        cout<<"Enter two No.\n";
        cin>>i>>j;
        if(!strcmp(command, "add"))
            cout<<i+j<<"\n";
        if(!strcmp(command, "sub"))
            cout<<i-j<<"\n";
        if(!strcmp(command, "mul"))
            cout<<i*j<<"\n";
        if(!strcmp(command, "div"))
            cout<<i/j<<"\n";
        if(!strcmp(command, "mod"))
            cout<<i%j<<"\n" ;}
    system("pause");
    return 0;
}
```

برنامه‌اي بنويسيد كه دو آرايه 5 عنصری را دريافت كرده مجموع دو آرايه را در آرايه سوم قرار دهد.

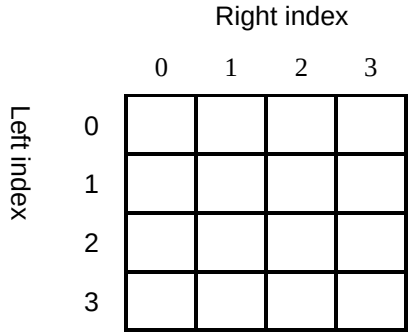
```
#include "stdafx.h"
#include <iostream>
using namespace std;

int main()
{
    int a[5], b[5], c[5], i;
    for(i=0;i<5;i++)
    {
        cin>>a[i];
        cin>>b[i];
    }
    for(i=0;i<5;i++)
    {
        c[i]=a[i]+ b[i];
        cout<<a[i]<<"+"<<b[i]<<"="<<c[i]<<"\t";
    }
    system("pause");
    return 0;
}
```

آرایه های چندبعدی

یک آرایه دوبعدی از چپ به راست و یک ردیف یک ردیف مورد دستیابی قرار می‌گیرد.

```
int count[10][12];
```

سطر ستون



نمایش ذهنی از آرایه‌های دوبعدی

مثال

```
#include "stdafx.h"
#include <iostream>
using namespace std;
int main()
{
    int two_d[4][5],i,j;
    for(i=0;i<4;i++)
     for(j=0;j<5;j++)
        two_d[i][j]=i*j;
    for(i=0;i<4;i++)
    {
     for(j=0;j<5;j++)
        cout<<two_d[i][j]<<"\t";
     cout<<"\n";
    }
    system("pause");
    return 0;
}
```

Sizeof(int)

4*5*4=80

Output

0	0	0	0	0
0	1	2	3	4
0	2	4	6	8
0	3	6	9	12

Press any key to continue . . .

مقداردهی اولیه آرایه ها

```
type array-name[size]={value-list};
```

```
char k[5]={1,4,9,16,25};
```

0	1	2	3	4
1	4	9	16	25

```
k[0] =>  1
```

```
k[4] =>  25
```

```
char m[5]={0};
```

0	1	2	3	4
0	0	0	0	0

```
int B[2][3]={{1},{4,9}};
```

0	1	2
1	0	0
4	9	0

```
int B[2][3]={{1},{4,9}};
```

0	1	2
1	4	9
0	0	0

مثال

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

int main()
{
    double radians[5][2]={
        1.0, 0.0175,
        2.0, 0.0349,
        3.0, 0.0524,
        4.0, 0.0698,
        5.0, 0.0873
    };
    double degrees;
    int i;
    cin>>degrees;
    for(i=0; i<5; i++)
        if (radians[i][0]==degrees)
        {
            cout<<radians[i][1]<<"\n";;
            break;
        }
    system("pause");
    return 0;
}
```

Output
4
0.0698
Press any key to continue . . .

آرایه‌های بدون اندازه

```
int pwr[]={1,2,4,8,16,32,64,128};
char prompt[]="Enter your name";
```

آرایه‌هایی که صراحتا ابعاد آنها مشخص نشده است.تعداد کاراکترها و یا اعداد را شمرده و اندازه آرایه را آن تغییر می‌دهد.

```
int sq[][3]={
1, 2, 3,
4, 5, 6,
7, 8, 9
};
```

مزیت این نحوه معرفی آرایه‌ها این است که می‌توان جدول را بدون آنکه ابعاد آرایه را تغییر دهیم، بلندتر یا کوتاهتر نماییم.

جداول رشته‌ای

```
char names[10][40];
```

۱۰ رشته هر کدام با طول حداکثر ۴۰ کاراکتر (با احتساب کاراکتر ختم کننده تهی)

```
cout<<names[0];
```
اولین رشته جدول

```
cin>>names[2];
```
دریافت سومین رشته جدول

توابع کاراکتری ...is

تابع	کاری که انجام می‌دهد
isalnum(ch)	حروف الفبا یا یک رقم باشد، true برمی‌گرداند.
isalpha(ch)	حروف الفبا باشد، true برمی‌گرداند.
isdigit(ch)	یک رقم باشد، true برمی‌گرداند.
isspace(ch)	فاصله خالی باشد، true برمی‌گرداند.

ctype.h

برنامه‌ای بنویسید که تعداد فاصله‌های یک متن را اعلام می‌کند.

```
#include "stdafx.h"
#include <iostream>
#include <ctype.h>
#include <string.h>

using namespace std;
int main()
{
    char str[]="This is a test";
    int i, spaces;
    spaces=0;
    for(i=0; i<strlen(str); i++)
        if(isspace(str[i]))
            spaces++;
    cout<<spaces<<"\n";
    system("pause");
    return 0;
}
```

```
Output
3
Press any key to continue . . .
```

ثابت‌های کاراکتری خاص

کد	معنا
\b	backspace
\t	tab
\”	کاراکتر گیومه
\’	کاراکتر آپستروف
\n	newline
\?	کاراکتر علامت سوال
\\	back slash کاراکتر
\a	Beep صدای

تمرین

برنامه‌ای بنویسید که
- سه عدد دریافت کرده بزرگترین آن‌ها را نمایش دهد.
- عددی را از ورودی دریافت کرده، قدر مطلق آن را نمایش دهد.
- عددی را از ورودی دریافت کرده، بخشپذیری آن بر ۳ و ۵ را بررسی نماید.
- دو عدد را دریافت کرده و بدون استفاده از متغیر کمکی تعویض نماید.
- سه عدد را دریافت کرده به صورت صعودی مرتب کند.
- یک عدد بین ۱ تا ۷ دریافت کرده معادل روز هفته را نمایش دهد.
- شعاع دایره را دریافت کرده، محیط و مساحت دایره را محاسبه کرده و نمایش دهد.
- جدول ضرب ۱۰*۱۰ را نمایش دهد.
- دو عدد را دریافت کرده، اعلام کند آیا این دو عدد متوالی هستند یا خیر؟
- دو عدد دریافت کرده، خارج قسمت و باقیمانده صحیح تقسیم را نمایش دهید.
- دو عدد را دریافت کرده، اعداد مابین این دو عدد را نمایش دهد. a<b
- یک عدد را دریافت کرده، مقلوب آن را نمایش دهد.
- تعدادی عدد دریافت کرده، مجموع آن‌ها را نمایش دهد. آخرین عدد صفر است.
- ضرایب یک معادله درجه ۲ر ا دریافت کرده، ریشه‌های آن را درصورت وجود محاسبه کرده و نمایش دهد.

تمرین

برنامه‌ای بنویسید که
- نمایش کاراکترهای **a** تا **d** به همراه کد اسکی آنها.
- تعداد و مجموع ارقام یک عدد دریافتی را نمایش دهد.
- مقداری را به عنوان رمز دریافت کند، در صورتی که رمز درست بود اعلام کند به برنامه خوش آمدید در غیراینصورت اعلام کند رمز اشتباه است.
- ۲۰ جمله اول سری فیبوناچی را نمایش دهد.
- مضارب ۷ بین ۱ تا ۱۰۰ را نمایش دهد.
- اعداد فرد کوچکتر از ۲۰ را چاپ کند.
- ۱۰۰ عدد را دریافت کرده، میانگین آن‌ها را محاسبه کرده و نمایش دهد.
- ۵۰ عدد را دریافت کرده، بزرگترین و کوچکترین عدد را نمایش دهد.
- یک عدد را دریافت کرده فاکتوریل آن را محاسبه کرده و نمایش دهد.
- نمره دانشجوئی را از ورودی دریافت کرده، با توجه به مقدار نمره یکی از خروجی های زیر را نمایش دهد.

Grade	output
17 – 20	A
14 - 17	B
12 – 14	C
10 – 12	D
0 – 10	F

تمرین

برنامه‌ای بنویسید که
- یک متن حداکثر ۱۰۰ کاراکتری را در یافت کرده، تعداد تکرار **A**، تعداد تکرار **DA** را نمایش دهد و تمام **K**ها را با **L**جایگزین کند.
- یک رشته راخوانده و تمام حروف بزرگ به کوچک تبدیل نماید.
- یک آرایه ۲۰ عنصری را دریافت کرده بزرگترین عنصر را به همراه اندیس آن نشان دهد.
- دو ماتریس ۴*۴ را از ورودی دریافت کرده، تفاضل آن دو را در خروجی نمایش دهد.
- یک آرایه ۱۰ عنصری را دریافت کرده اعلام کند هر عنصر زوج است یا فرد.
- یک ماتریس ۴*۴ را دریافت کرده و عناصر قطر اصلی را یک و بقیه را صفر قرار دهد.
- یک آرایه ۵ عنصری دریافت کرده یک واحد به آنها اضافه کرده و نمایش دهد.
- دو ماتریس ۴*۳ و ۵*۴ را دریافت کرده، حاصلضرب آن دو را نمایش دهد.
- نام ۴۰ دانش‌آموز و نمره آن‌ها را از ورودی دریافت کرده، نام و نمره دانش‌آموزانی که نمره آن‌ها بیشتر از ۱۸ است را نمایش دهد.
- یک عدد در مبنای ۱۰ را دریافت کرده معادل آن را در مبنای ۲نمایش دهد.
- آرایه ۲ بعدی ۱۰ در ۱۰ را با مقادیر جدول ضرب، مقدار دهی کرده و نمایش دهد.
- توسط آرایه ، نمودار افقی برای اعداد {۵ ۱۳و۸ و۱۰ و۵ و۱۱} رسم کند .
- یک آرایه را دریافت و آرایه دیگر به صورت وارونه کپی کرده، آرایه دوم را نمایش دهد.

تابع

```
#include "stdafx.h"
#include <iostream>

using namespace std;
void Areas(int base,int height);
int main()
{
    Areas(10, 20);
    Areas(5, 6);
    Areas(8, 9);
    system("pause");
    return 0;
}

void Areas(int base, int height)
{
    cout<<"Area is:"<<base*height/2<<"\n";
}
```

Prototype

argument

Parameters

تابع

تابع شامل یک یا چند دستور بوده و عمل بخصوصی را انجام می‌دهد.

تابع دارای نامی است و با همین نام فراخوانی می‌شود.

در یک تابع نمی‌توان تابع دیگری ایجاد کرد اما می‌توان تابع دیگری را فراخوانی کرد.

برای استفاده مجدد تابع نوشته شده، بهتر است برای تابع header file ساخته و با پسوند h. در فولدر include که مخصوص header file ها است ذخیره نمود.

prototype شامل سه چیز است: نوع مقدار بازگشتی تابع، تعداد پارامترهای تابع، نوع داده پارامترهای آن تابع

پیش‌الگوی یک تابع (function prototype) برای معرفی آن تابع، پیش از آنکه تعریف شود، استفاده می‌گردد.

تابع

مقدار بازگشتی توابع باید مشخص شود. اگر مشخص نشود اکثرا به عنوان int درنظر گرفته می‌شود. درصورتی‌که تابع دارای مقدار بازگشتی نباشد نوع مقدار بازگشتی void تعریف می‌شود(تابع return ندارد).

به مقداری که به یک تابع ارسال می‌شود argument می‌گویند.

حد بالای آرگومانها توسط کامپایلری که از آن استفاده می‌شود تعیین می‌گردد، اما هر کامپایلر استانداردی حداقل ۲۵۶ آرگومان را قبول می‌کند.

متغیرهایی که آن آرگومانها را دریافت می‌کنند نیز باید معرفی شوند، به این متغیرها parameters گفته می‌شود.

تابع

```
#include "stdafx.h"
#include <iostream>

using namespace std;
float Areas(float base,float height);
int main()
{
    float S,x,y;
    cin>>x>>y;
    S=Areas(x,y);
    cout<<"The area is:"<<S<<"\n";
    system("pause");
    return 0;
}

float Areas(float base, float height)
{
 return base*height/2;
}
```

```
Output
3
4
The area is:6
Press any key to continue . . .
```

توابع ریاضی

استفاده از توابع ریاضی با فایل سرآمد math.h

تابع در زبان C	عملیات ریاضی
double cos(double t)	cos t
double exp(double t)	e^t
double fmod(double x, double y)	x%y
double log10(double t)	$\log^t$
double pow(double x, double y)	x^y
double sin(double t)	Sin t
double sqrt(double t)	$\sqrt{t}$

در C++ می‌توان توابعی به صورت inline تعریف کرد که واقعا فراخوانی نشوند، بلکه در هر نقطه‌ای که فراخوانی شده‌اند کد آنها قرار داده شود. این توابع سریعتر اجرا می‌شوند. پیش از استفاده حتما باید تعریف شود.

```
#include "stdafx.h"
#include <iostream>

using namespace std;

inline int even(int x)
{
    return !(x%2);
}
int main()
{
    if(even(10)) cout<<"10 is even \n";
    if(even(11)) cout<<"11 is even \n";
    if(even(12)) cout<<"12 is even \n";
    system("pause");
    return 0;
}
```

```
Output
10 is even
12 is even
Press any key to continue . . .
```

تابع

زمانی که به انتهای تابع و { یا به دستور return برسیم تابع به روتین فراخواننده خود باز می‌گردد. return می‌تواند مقداری را به روتین فراخواننده‌اش بازگرداند.

یک تابع می‌تواند چندین دستور return داشته باشد.

تابعی که خودش را فرابخواند recursive نامیده می‌شود.

جملات فیبوناچی تا جمله n

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

int fibo(int x);
int main()
{
    int i, n;
    cin>>n;
    for (i=1;i<=n;i++)
        cout<<fibo(i)<<"\t";
     system("pause");
     return 0;
}
int fibo(int x)
{
    if (x<=2) return 1;
    else return fibo(x-1)+fibo(x-2);
}
```

میدان دید (scope)

```
int myabs(int x);
int main()
{
    int i;
    for (i=1;i<=8;i++)
    {
        int n;
        cin>>n;
        cout<<myabs(n)<<"\n";
    }
    system("pause");
    return 0;
}
int myabs(int x)
{
    if (x<=0) return -1 * x;
    else return x;
}
```

static

```
#include "stdafx.h"
#include <iostream>
using namespace std;
void f();
int main()
{
    int i;
    for (i=1;i<=10;i++)
        f();
    system("pause");
    return 0;
}
void f(){
    static int count=0;
    count++;
    cout<<count<<"\n";
}
```

Output

```
1
2
3
4
5
6
7
8
9
10
Press any key to continue . . .
```

static باعث می‌شود که یک متغیر محلی بین دو فراخوانی تابع دست نخورده باقی بماند. تنها یکبار مقداردهی اولیه می‌شود آن هم زمانیکه برای نخستین بار وارد بلوک می‌شود.

روشهای انتقال پارامترهای تابع

call by value, call by reference

value: در این روش مقدار آرگومان به پارامتر صوری آن سابروتین کپی می‌شود بنابراین تغییراتی که روی مقادیر پارامترهای آن سابروتین اعمال می‌شود روی آرگومانهایی که جهت فراخوانی آن سابروتین به کار برده می‌شود تاثیری **نخواهد داشت**.

reference: در این روش به جای مقدار آرگومان آدرس آن به پارامتر مورد نظر کپی می‌شود بنابراین تغییراتی که روی مقادیر پارامترهای آن سابروتین اعمال می‌شود روی آرگومانهایی که جهت فراخوانی آن سابروتین به کار برده می‌شود تاثیر **خواهد داشت**.

```
#include "stdafx.h"
#include <iostream>
using namespace std;
void f(double x, double y);
int main(){
     double i,j;
     i=6.45;
     j=9.35;
     f(i,j);
     cout<<i<<"\t"<<j<<"\t";
     system("pause");
     return 0;
}
void f(double x, double y){
     x=x+3;
     y=y-3;
     cout<<x<<"\t"<<y<<"\t";;
}
```

Output
9.45 6.35 6.45 9.35 Press any key to continue . . .

```
#include "stdafx.h"
#include <iostream>
using namespace std;
void f(double &x, double &y);
int main(){
     double i,j;
     i=6.45;
     j=9.35;
     f(i,j);
     cout<<i<<"\t"<<j<<"\t";
     system("pause");
     return 0;
}
void f(double &x, double &y){
     x=x+3;
     y=y-3;
     cout<<x<<"\t"<<y<<"\t";
}
```

Output
9.45 6.35 9.45 6.35 Press any key to continue . . .

const

اگر پیش از نوع داده یک متغیر توصیف کننده const آورده شود محتوای آن از سوی برنامه تغییر نمی‌کند. (مگر سخت‌افزاری)

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

int main(){
    const int ten=10;
    cout<<ten*5<<"\n";
    system("pause");
    return 0;
}
```

typedef

typedef OldDataType NewName;

با استفاده از typedef برای یک نوع داده موجود نام تازه‌ای ایجاد کرد.

```
typedef int length;
typedef length depth;
depth d;
typedef short int myint;
```

استفاده از عملگر ؟

```
if(condition) var=exp1;
else var=exp2;
```

```
var=condition ? exp1:exp2;
```

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

int main(){
    int i;
    cout<<"Enter a number.\n";
    cin>>i;
    i=i>0 ? 1: -1;
    cout<<i<<"\n";
    system("pause");
    return 0;
}
```

تمرین

برنامه‌ای بنویسید که
- ۵۰ اسم را دریافت کرده، تعداد اسامی که با **B** شروع می‌شود را نمایش دهد.
- عدد را دریافت کرده، تعداد اعدادی را که به ۴ ختم می‌شوند، نمایش دهد.
- دو رشته را از ورودی دریافت کرده، به هم پیوند دهد و در خروجی نمایش دهد.
- مقادیر **k** و **n** را دریافت کرده، و تابع زیر را با فراخوانی تابع فاکتوریل محاسبه کرده و نمایش دهد. **y=n!k!/(k!-(n-k)!)**
- با استفاده از فراخوانی تابع مقدار تابع زیر را محاسبه کرده و نمایش دهد.

$$y = \begin{cases} x^3 + 4x + 5 & x \geq 0 \\ 1 & x < 0 \end{cases}$$

- عناصر آرایه را گرفته مکعب آنها را محاسبه و در همان عنصر ذخیره نمایید.
- با استفاده از تابع بازگشتی فاکتوریل عدد دریافتی را محاسبه کرده و نمایش دهد.
- با استفاده از تابع محیط و مساحت مستطیل را محاسبه نماید.
- ۱۰ عدد را از ورودی دریافت کرده و در یک آرایه قرار دهد. سپس یک عدد دیگر را خوانده و در آرایه جستجو کند و وجود یا عدم وجود آنرا در آرایه مشخص کند.
- یک عدد بین ۱ تا ۱۲ دریافت کرده معادل ماه مربوطه را نمایش دهد.
- ۱۰ عدد مرتب شده را از ورودی دریافت کرده و در یک آرایه قرار دهد. سپس یک عدد دیگر را خوانده و در جای مناسب در آرایه قرار دهید.

cin.get()

این تابع یک کاراکتر را از صفحه کلید می‌گیرد. برای استفاده از این تابع در ابتدای برنامه باید از فایل سرآمد `iostream` استفاده شود.

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

`cin.get (`طول آرایه و نام آرایه`);`

```
cin.get (S,15);
```

در این روش کامپایلر آنقدر از کاربر حرف می گیرد تا به طول آرایه تعریف شده برسد.ممکن است کلمه مورد نظر کاربر ۳ حرفی باشد برای حل این مشکل از روش زیر استفاده کنیم: `cin. get (S,15,'*');`

کاراکتر جدا کننده در واقع شرط پایان است که خودمان تعیین می کنیم و یادمان باشد همواره ۱کاراکتر از کاراکترهای داده شده کم می شود چون از آنجا شرط تمام است.

مثال

یک سطر متن انگلیسی که به CTRL Z ختم میشود را دریافت کرده و نمایش می‌دهد.

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

int main(){
    char  x;
    while((x = cin.get( )) !=EOF)
        cout << x ;
    system("pause");
    return 0;
}
```

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.

#define

از define جهت تعریف یک شناسه و یک دنباله کاراکتری استفاده می‌شود طوری که در source برنامه هرجا با آن شناسه برخورد شود آن دنباله کاراکتری جانشین آن گردد.

```
#define macro-name character-sequence
```

```
#define UP 1
#define GETFILE "Enter File Name"
#define myfor for(int k=10;k<15;k++)
```

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

int main(){
    #define myfor for(int k=10;k<15;k++)
    myfor
        cout<<k<<"\t";
    system("pause");
    return 0;
}
```