

یک برنامه ساده

```
/*  
A simple program.  
This program contains all of the  
basic elements  
*/  
  
#include "stdafx.h"  
#include <iostream>  
using namespace std;  

```

یک برنامه ساده

```
/*  
A simple program.  
This program contains all of the  
basic elements  
*/  
  
#include "stdafx.h"  
#include <iostream>  
#include <conio.h>  

```

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 آن جا 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 می‌شود.

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

cout

یک شناسه از پیش تعریف شده است که مخفف console output می باشد و برای نمایش روی صفحه نمایش به کار می رود.
 ✓ انتهای تمام دستورات در C++ به سمی کولون (;) ختم می شود.

stdafx.h

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

getch()

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

return 0

با این دستور تابع main پایان می یابد و مقدار صفر به پروسه فراخواننده main که معمولا سیستم عامل است بازگردانده می شود.

مقدار صفر نشان دهنده آن است که برنامه به طور عادی پایان یافته است. مقادیر دیگر نشان دهنده آن است که برنامه به علت خطایی پایان یافته است. return یکی از کلیدواژه های C++ می باشد که برای بازگرداندن مقداری از یک تابع به کار می رود.

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	false و true
char	۱۲۷ تا ۱۲۸-	کاراکترهای ۸ بیتی 'A'
int	۳۲۷۶۷ تا ۳۲۷۶۸-	عدد صحیح
float	۳۸+۴/۳E تا ۳۸-۴/۳E	مقدار اعشاری
double	۳۰۸+۷/۱E تا ۳۰۸-۷/۱E	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)

فروجهی 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

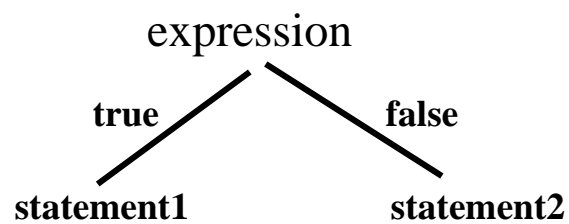
```
#include "stdafx.h"
#include <iostream>
#include <conio.h>

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
استفاده endl
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;  
}
```



```

#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 else

```

#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;
}

```

استفاده از بلوک کد

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;
}
```

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

Output

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

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

از عملگرهای یکتائی می باشد و مشخص کننده تعداد بایت هائی است که یک نوع داده اشغال می کند.

```

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

می توان 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 می شود.
j=++i;	⇒	ابتدا مقدار i یک واحد اضافه می شود و سپس مقدار i به j نسبت داده می شود.

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>

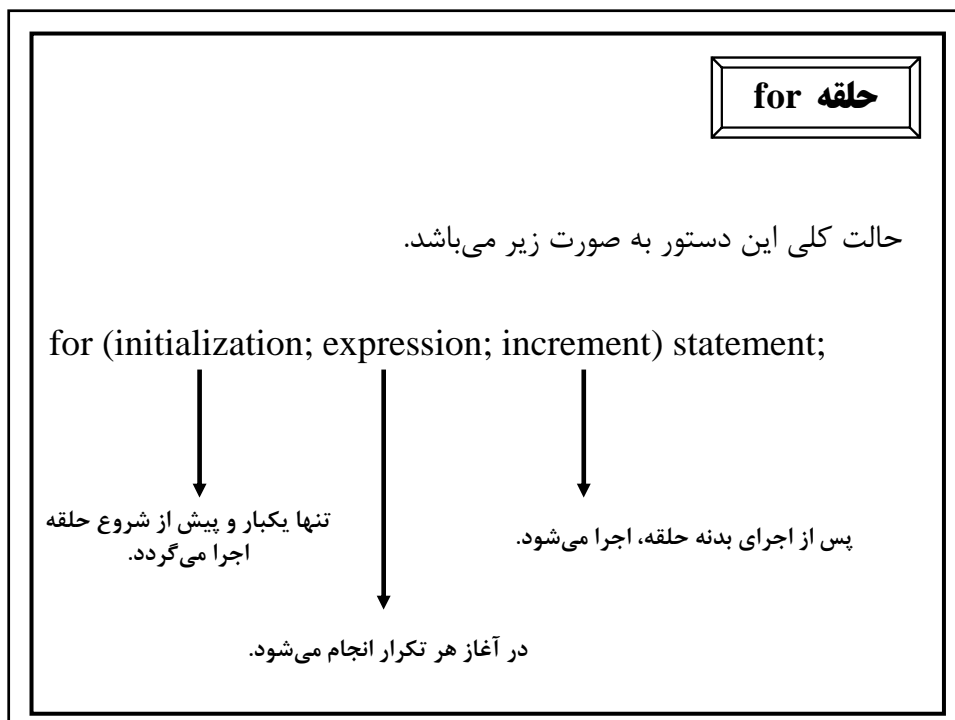
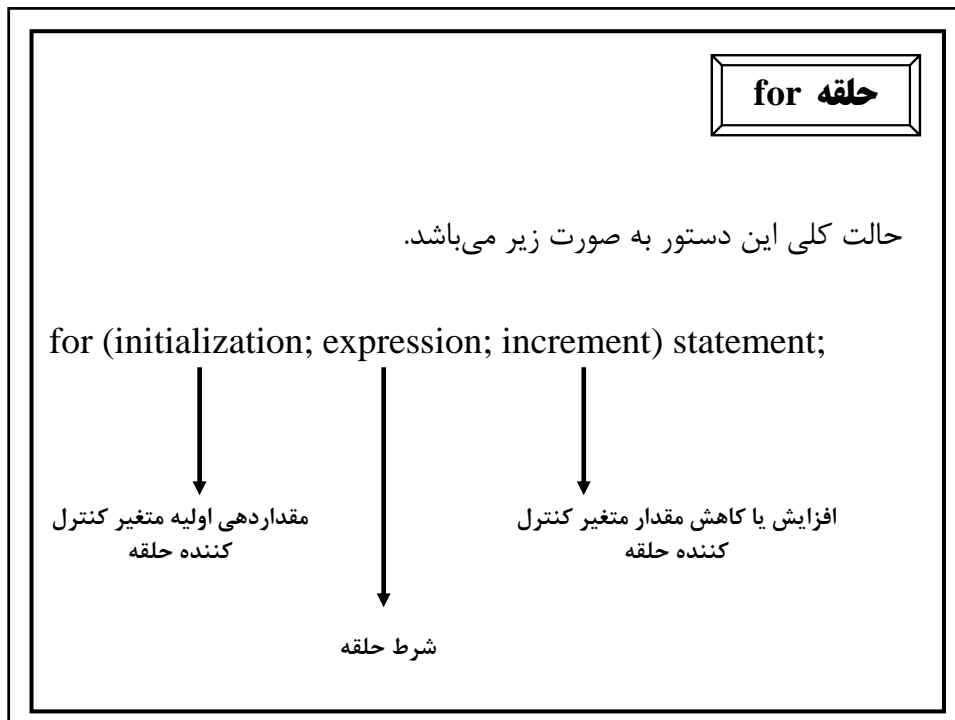
using namespace std;
int main()
{
int i, j, k, l;
i=15;
j=20;
k=i--;
l=--j;
cout<<i<<"\n"<<j<<"\n"<<k<<"\n"<<l<<"\n";
system("pause");
return 0;
}

```

Output

14
19
15
19

Press any key to continue . . .



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 می تواند 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 و نمایش 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 ...

دو عدد 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` \Rightarrow `false`

`!=0` \Rightarrow `true`

```
#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;
}
```

استفاده از مقادیر منطقی

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; فرض می شود عدد اول است.
    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;
}
```

حلقه while

حالت کلی این دستور به صورت زیر می باشد.

while (expression) statement;

↓
شرط حلقه

تا زمانی که شرط برقرار باشد دستورات اجرا می شود.

شرط حلقه در ابتدای حلقه کنترل می شود.

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 می باشد.

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

↓
شرط حلقه

تا زمانیکه شرط برقرار باشد دستورات اجرا می شود.
شرط حلقه در انتهای حلقه کنترل می شود.
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 می باشد.

```
#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;
}
```

goto label;

label:

↓

برچسب

با دیدن goto به خطی می رود که برچسب یکسانی دارد.

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

نگاهی دقیق تر به نوع داده ها

با استفاده از توصیف کننده های نوع (type modifiers) می توان نوع داده های char, int, float, double را تعدیل کرد. توصیف کننده های نوع عبارتند از :

signed, unsigned, long, short

توصیف کننده نوع پیش از نوع داده آورده می شود.

long int i;

signed (علامت دار)

signed را می توان در مورد انواع char و int به کار برد.

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

بسته به نوع پیاده سازی یک char می تواند علامت دار یا بدون علامت باشد. در صورتی که بدون علامت باشد اعداد مثبت بین ۰ تا ۲۵۵ را نگاه می دارد. در صورتی که به عنوان signed باشد می تواند اعدادی در دامنه ۱۲۸- تا ۱۲۷ را نگاه می دارد. signed char x;

long

long را می‌توان در مورد نوع داده‌های int و double به کار 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 به کار 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 به کار 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- عموماً اعداد منفی با روش two's complement (مکمل دو) بیان می‌شوند.

ترکیبات ممکن از توصیف کننده‌ها

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

ترکیبات ممکن از توصیف کننده‌ها

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 بهم 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];
```

مقدار عنصر صفر آرایه را برابر با ۱۰۰ قرار می‌دهد. `d[0]=100;`

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.

```
int t[20];
```

مثال:

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

برنامه‌ای بنویسید که ۳۰ عدد صحیح مثبت را دریافت کرده و max آنها را بیابید، نمایش دهید.

```

#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.

برنامه‌ای بنویسید که ۵۰ عدد صحیح مثبت را دریافت کرده و آنها را به ترتیب عکس دریافت نمایش دهید.

```

#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` را به `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` را به `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 ختم‌کننده 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()

کاراکتر ورودی را به معادل بزرگ خود تبدیل می‌کند.

```
int toupper(int ch);
```

کاراکتر ورودی را به معادل کوچک خود تبدیل می‌کند.

```
int tolower(int ch);
```

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.

```
#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;
}
```

مثال

برنامه‌ای بنویسید که دو آرایه ۵ عنصری را دریافت کرده مجموع دو آرایه را در آرایه 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

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

سطر

ستون

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

Right index

	0	1	2	3
Left index	0			
	1			
	2			
	3			

نمایش ذهنی از آرایه های 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

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

0	1	2
1	0	0
4	9	0

char m[5]={0};

0	1	2	3	4
0	0	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 صراحتاً 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 رشته‌های

`char names[10][40];`

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

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

`cout<<names[0];`
اولین رشته جدول

ثابت‌های کاراکتری خاص

معنا	کد
tab	\t
newline	\n
کاراکتر back slash	\\

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 . . .

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 10×10 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 ۲ را دریافت کرده، ریشه‌های آن را در صورت 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.
- ❖ نمره دانشجویی 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 دریافت کرده، تعداد تکرار **A**، تعداد تکرار **DA** را نمایش دهد و تمام **K**ها را با **L** جایگزین کند.
- ❖ یک رشته را خوانده 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 ۳*۴ و ۴*۵ را دریافت کرده، حاصلضرب آن دو را نمایش دهد.
- ❖ نام ۴۰ دانش آموز و نمره آنها را از ورودی دریافت کرده، نام و نمره دانش آموزانی که نمره آنها بیشتر از ۱۸ است را نمایش دهد.
- ❖ یک عدد در مبنای ۱۰ را دریافت کرده معادل آن را در مبنای ۲ نمایش 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، ۱۳، ۸ و ۱۰ و ۵ و ۱۱} 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 می دهد.

تابع دارای نامی است و با همین نام فراخوانی می شود.

در یک تابع نمی توان تابع دیگری ایجاد کرد اما می توان 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

پیش الگوی یک تابع (function prototype) برای معرفی آن تابع، پیش از آنکه تعریف شود، استفاده می گردد.

تابع

مقدار بازگشتی توابع باید مشخص شود. اگر مشخص نشود اکثرا به عنوان `int` در نظر گرفته می‌شود. در صورتی که 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 استاندارد 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 . . .

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

در C++ می توان توابعی به صورت inline تعریف کرد که واقعا فراخوانی نشوند، بلکه در هر نقطه ای که فراخوانی شده اند کد آنها قرار داده شود. این توابع سریعتر اجرا می شوند. پیش از استفاده حتما باید تعریف شود.

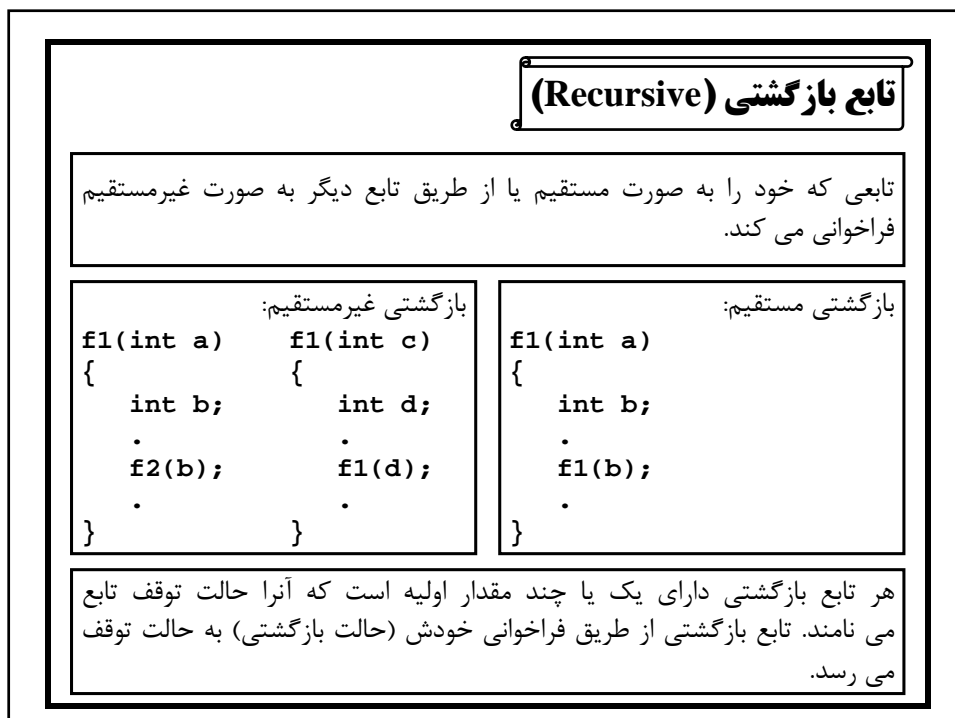
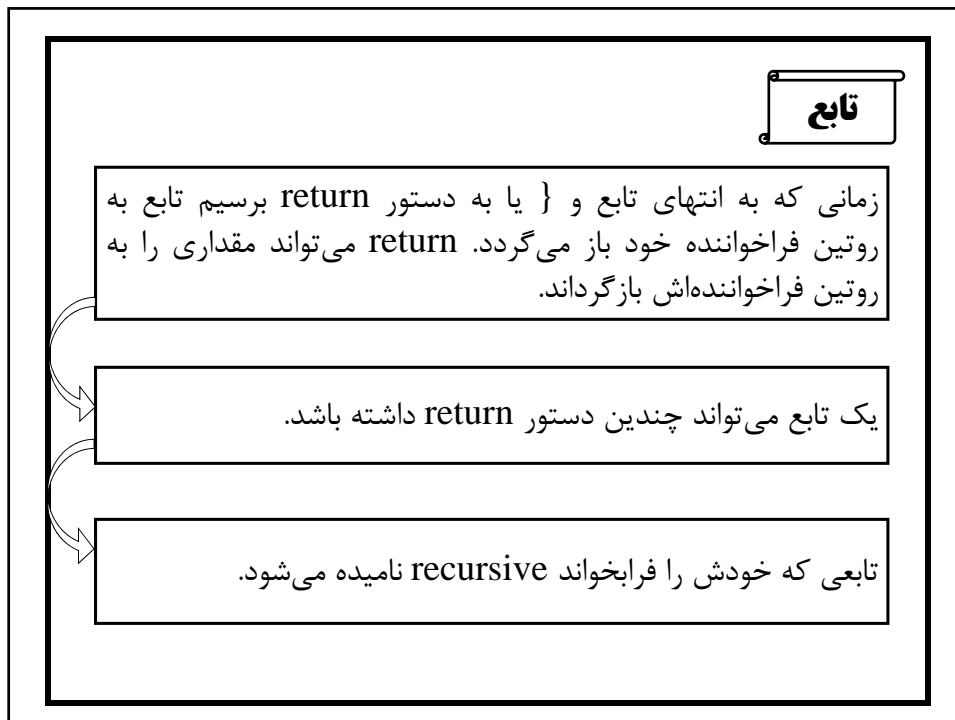
```
#include "stdafx.h"
#include <iostream>

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 ...



جملات فیوناچی 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;
}
```

روشهای انتقال پارامترهای تابع

call by value, call by reference

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

call by value

```
#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...

call by reference

```
#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...