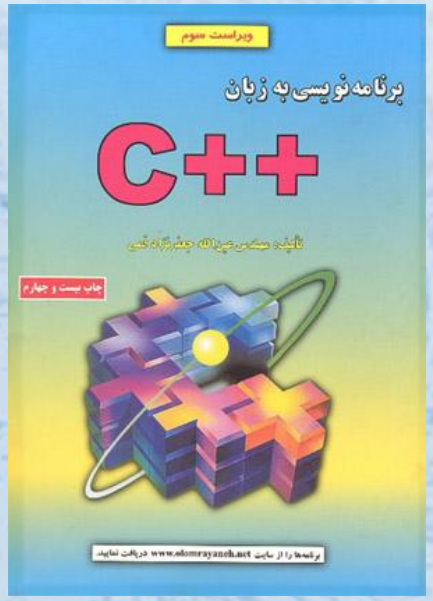
برنامه نویسی C++

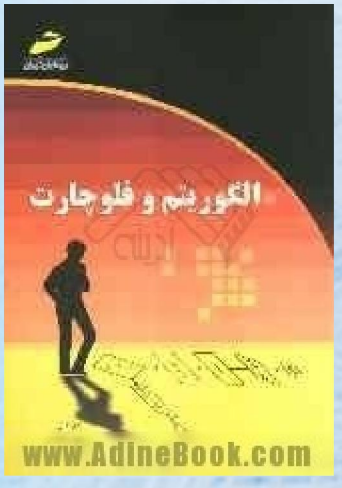
– مراجع

برنامه نویسی به زبان C++

نویسنده: عین الله جعفرنژاد قمی
انتشارات: علوم رایانه



الگوریتم و فلوچارت
نویسنده: بهرام غلامی
انتشارات: دیباگران تهران



C++برنامه نويسي به زبان
نويسنده: جعفرنژاد قمي

فصل اول

مبانی زبان C++

تاریخچه مختصر زبان C++

- در دهه ۱۹۷۰ در آزمایشگاه‌های بل زبانی به نام C ایجاد شد. انحصار این زبان در اختیار شرکت بل بود تا این که در سال ۱۹۷۸ توسط Kernighan و Richie شرح کاملی از این زبان منتشر شد و به سرعت نظر برنامه‌نویسان حرفه‌ای را جلب نمود.
- هنگامی که بحث شی‌گرایی و مزایای آن در جهان نرم‌افزار رونق یافت، زبان C که قابلیت شی‌گرایی نداشت ناقص به نظر می‌رسید تا این که در اوایل دهۀ ۱۹۸۰ دوباره شرکت بل دست به کار شد و Bjarne Stroustrup زبان C++ را طراحی نمود.

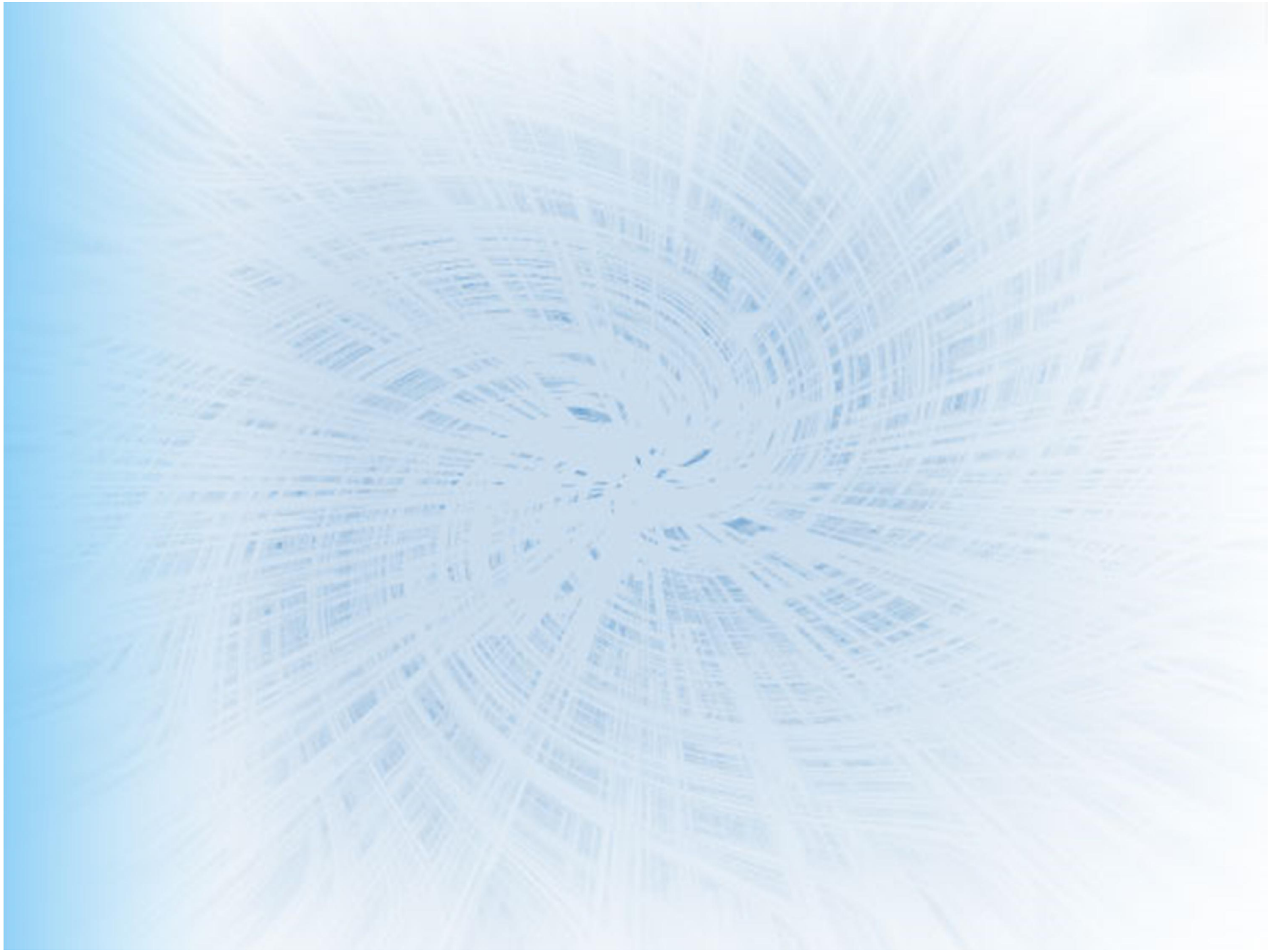
تاریخچه مختصر C++

- C++ ترکیبی از دو زبان C و Simula بود و قابلیت‌های شی‌گرایی نیز داشت از آن زمان به بعد شرکت‌های زیادی کامپایلرهایی برای C++ طراحی کردند. این امر سبب شد تفاوت‌هایی بین نسخه‌های مختلف این زبان به وجود بیاید و از قابلیت سازگاری و انتقال آن کاسته شود.
- به همین دلیل در سال ۱۹۹۸ زبان C++ توسط موسسهٔ استانداردهای ملی آمریکا (ANSI) به شکل استاندارد و یک‌پارچه درآمد. کامپایلرهای کنونی به این استاندارد پایبندند.

تفاوت زبان C و C++

- زبان C یک زبان همه منظوره است. دستورالعمل‌هاي اين زبان بسيارشبيه عبارات جبري و نحو آن شبيه جملات انگليسي مي باشد. اين امرسبب مي‌شود که C یک زبان سطح بالا باشد که برنامه‌نويسي درآن آسان است.

- C++ که ازنسل C است، تمام ويژگي‌هاي C را به ارث برده است. اما برتري فني ديگري هم دارد: C++ اکنون «شي‌گرا» است. مي‌توان با استفاده ازاين خاصيت، برنامه‌هاي شي‌گرا توليد نمود. برنامه‌هاي شي‌گرا منظم و ساخت‌يافته‌اند، قابل روزآمد کردن‌اند، به سهولت تغييرو بهبود مي‌يابند و قابليت اطمينان و پايداري بيشتري دارند.

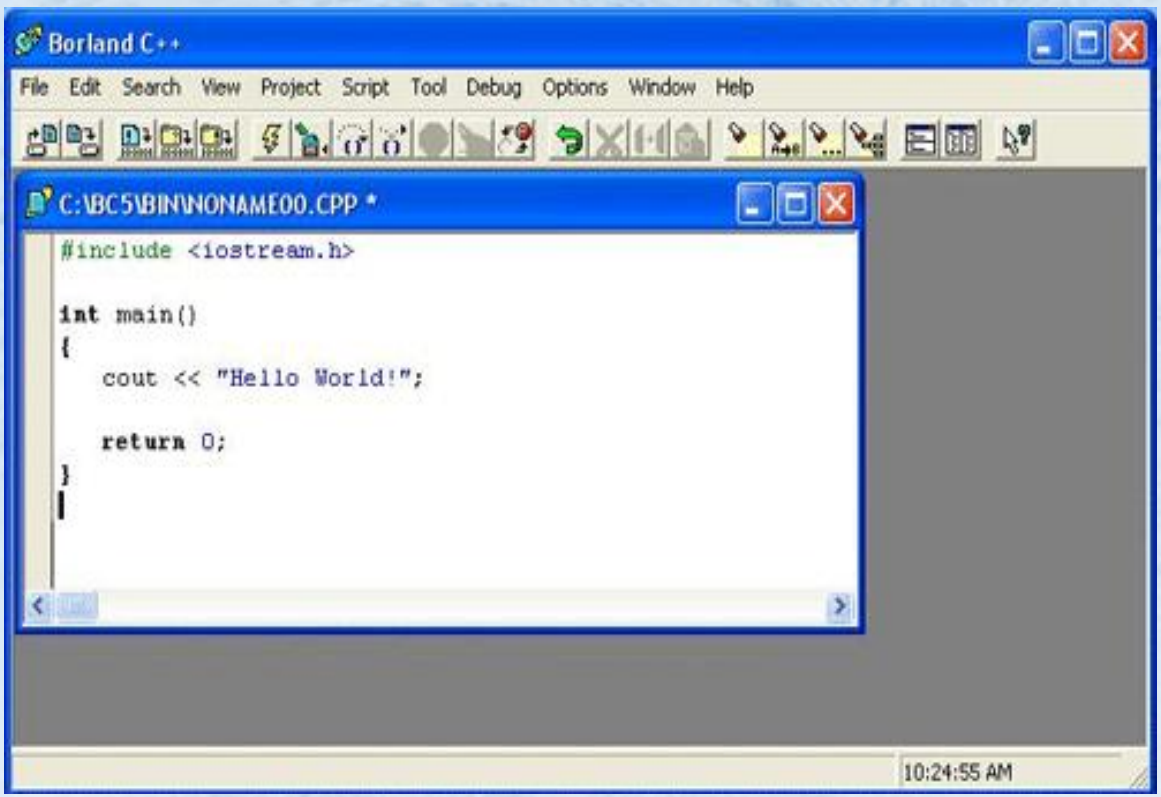


معرفی ساختاری زبان C++:

- ✓ C++ عموماً از سه بخش تشکیل شده است :

1. -محیطی برای نوشتن برنامه و ویرایش آن.
2. - کامپایلر C++.
3. - کتابخانه استاندارد C++.

در آغاز عکسی از برنامه Borland C++را نشان می دهم تا با محیط نرم افزار C++آشنا شوید . هرچند نرم افزارهای دیگه هم در این خصوص وجود دارند مانند Visual C++یا Turbo C++


Borland C++
File Edit Search View Project Script Tool Debug Options Window Help
C:\BC5\BIN\NONAME00.CPP *
#include <iostream.h>
int main()
{
cout << "Hello World!";
return 0;
}
10:24:55 AM

ویژگی های زبان C++:

- C++ نسبت به حروف حساس است. یعنی a و A را یکی نمی‌داند.
- هر دستور زبان C++ به ; ختم مي‌شود.
- حداکثر طول يك دستور 255 کاراکتر است .
- هر دستور مي‌تواند در يك يا چند سطر ادامه داشته باشد.
- در هر سطر مي‌توان چند دستور را تايپ کرد.
- توضيحات مي‌تواند در بين */ و /* قرار بگيرند يا بعد از // ظاهر شوند .

ویژگی های زبان c++:

- بین حروف نام متغیر نمی‌توان از کاراکتر فاصله استفاده کرد
- زبان C++ دارای تعدادی کلمات کلیدی است که نمی‌توان از این کلمات به عنوان نام متغیر استفاده کرد.
- C++ زبان برنامه نویسی سیستم است.
- C++ زبان کوچکی است. تعدادکلمات کلیدی این زبان اندک است.
- درزبان c++ هیچ محدودیتی برای برنامه نویس وجود ندارد.
- برنامه نویس خود می‌تواند انواع جدیدی ازداده ها را ایجاد نماید. (با استفاده ازکلماتی مانندsigned,unsigned,short,long) که می‌توان این کلمات را با انواع int به کاربرد.

کلمات کلیدی زبان C++ عبارتند از:

char	cdecl	case	break	auto	asm
delete	default	_cs	continue	const	class
_es	enum	else	_ds	double	do
for	float	_fastcall	far	_export	extern
int	inline	if	huge	goto	friend
operator	new	near	long	_loadds	interrupt
return	register	public	protected	private	pascal
_ss	sizeof	signed	short	_seg	_saveregs
typedef	this	template	switch	struct	static
while	volatile	void	virtual	unsigned	union

داده ها در زبان c++:

- **در زبان c++ شش نوع داده اصلی وجود دارد که عبارتند از :**

❖ char: برای ذخیره داده های کاراکتری مانند 'W' و 'a' و... بکار می رود.

❖ int: برای ذخیره اعداد صحیح مانند 7 4 123 و... بکار می رود.

❖ float: برای ذخیره اعداد اعشاری بکار می رود.

❖ double: برای اعداد اعشاری بزرگتر از float استفاده میشود.

❖ void : هیچ مقداری را برنمیگرداند.

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

تعریف متغیر:

- خانه‌هایی ازحافظه هستند که داده‌ها در آنها قرار می‌گیرند.
- درطول یک برنامه مقدار متغیر می‌تواند تغییر کند.
- برای نامگذاری متغیرها از ترکیبی از حروف a تا z یا A تا Z ، ارقام و خط ربط (_) استفاده کرد بطوریکه اولین کاراکتر آنها رقم نباشد.
- ۳۱ کاراکتر اول آن مورد استفاده قرار می‌گیرد.

بعضی از اسامی مجاز و غیر مجاز برای متغیرها

اسامی مجاز	اسامی غیر مجاز
test test23 S_1	1test grade.1 .sum

روش اعلان متغیرها :

نوع نام متغیر ;

داده

```
int     x,y;
float   m,n;
double  d1;
bool    b;
char    ch1,ch2;
```

مقدار دادن به متغیرها

بعد از تعریف متغیرها باید مقداری به آنها نسبت دهیم:

روش مقداردادن به متغیرها:

1)هنگام اعلان متغیر

```
int   x , y = 5 ;
char   ch1='a', ch2='m';
```

۲)پس از اعلان نوع متغیر و با دستور انتساب (=)

```
int  x ;
flout   f1;
x=1;
f1=15.2;
```

- 3)دستورات ورودی
- Int x , y ;
- cin >> x >> y ;

تعریف ثابت‌ها و روش اعلان و مقداردهی به آنها:

ثابت‌ها مقادیری هستندکه دربرنامه وجوددارند ولی قابل تغییرنیستند.نامگذاری برای ثوابت از قانون نامگذاری برای متغیرها تبعیت می‌کند.

روش اعلان (تعریف) ثابت‌ها:

1. استفاده ازدستور #define
2. استفاده ازدستور const

استفاده از دستور # define:

#define <نام ثابت> <مقدار>

#define M 100

توجه

- دقت شود که در انتهای دستور **#define** علامت ; قرار نمی‌گیرد.

استفاده از دستور const:

Const <نوع داده ها> <نام ثابت> = <مقدار>;

Const int n=80;

تعریف عملگر و انواع آن در c++

- برای انجام عملیات بر روی داده ها از عملگرها استفاده می کنیم. عملگرها نمادهایی هستند که عملیاتی مانند جمع،ضرب، کوچکتری و از این قبیل را روی داده ها انجام می دهند که عبارتند از :
- انتساب (=)
- از این عملگر برای نسبت دادن یک مقدار به یک داده استفاده می شود .

b=8;

نحوه عملکرد این عملگرد به این شکله که مقدار سمت راست تساوی را در سمت چپ قرار میدهد.

انواع عملگرها

- **عملگرهاي محاسباتي: اعمال محاسباتي را روي عملوند انجام مي‌دهند.**

نام عملگر	علامت عملگر	مثال
تفریق یا منهای یکانی	-	x-y یا - x
جمع	+	X+y
ضرب	*	X*y
تقسیم	/	x/y
باقیمانده تقسیم	%	X%y
افزایش	++	x++ یا ++x
کاهش	--	x-- یا --x

- مثال
- اگر هر یک از عملگرهای + , -, *, /, ٪ از نوع صحیح باشد نتیجه عمل از نوع صحیح می‌باشد.

4 + 3 = 7

4 – 3 = 1

4 * 3 = 12

4 / 3 = 1

4 % 3 = 1

تمرین) a چه مقداری می‌گیرد؟

```
int  a;
a=1/3 + 2/3 + 1/3;
**********************
double  a;
a= (4+6+9)/3;
```

مثال

اگر حداقل یکی ازعملوندهای عملگرهای +, -, *, / از نوع اعشاری باشد نتیجه عمل اعشاری خواهد بود.

2.0+3=6.0

5*2.0=10.0

5.0/2=2.5

3.0-2=1.0

تمرین) xچه مقداری می‌گیرد.

```
double  x;
x=(4.0 + 6.0 + 9.0)/3.0
**********************
int  x;
x=(4.0 + 6.0 + 9.0)/3.0
```

تمرین) دستورات زیر را در نظر بگیرید:

```
int  a,b;
double  x;
a=(int) 1.0 / 3;
x=(int) 1.0 / 3;
```

عملگرهای کاهش و افزایش

سه دستورالعمل زیر معادلند:

```
x--;
--x;
x=x-1;
```

****

سه دستورالعمل زیر باهم معادلند:

```
x++;
++x;
x=x+1;
```

****

- دستورات زیر را در نظر بگیرید:

```
int   x,y;
x=10;
y=++x;
```

- با اجرای دستور دوم مقدار ۱۰ در x قرار می‌گیرد و با اجرای دستو ۳ ابتدا یک واحد به x اضافه شده و سپس مقدار حاصل در متغیر y قرار می‌گیرد یعنی y مساوی ۱۱ می‌شود.

- int x,y,m;
- x=10;
- y=4;
- m=++x+y++
- واضح است که: m=15 , y=5

- دستورات زیر را در نظر بگیرید:

```
int  a;
int  b,c;
b=2,c=3;
a=(b+c+4);
```

اگر دستور سوم نباشد در دستور چهارم بجای b,c یک مقدار الکی گذاشته شده وa یک مقدار الکی می‌شود.

عملگرهاي رابطه‌اي :

ارتباط بين عملوندها را مشخص مي كنند

نا م عملگر	علامت عملگر	مثال
بزرگتر	>	X > y
بزرگتر مساوی	>=	X >= y
کوچکتر	<	X < y
کوچکتر یا مساوی	<=	X <= y
متساوی	==	X == y
نامساوی	!=	X != y

عملگرهاي منطقي :

- بر روي عبارات منطقي عمل مي‌كنند. عبارات منطقي داراي ارزش درستي و نادرستي‌اند.

نام عملگر	علامت عملگر	مثال
نقيض يا (not)	!	!x
و يا (and)	&&	X > y && m < p
يا (or)	||	X > y || m < p

عملگرهاي تركيبي :

- از تركيب عملگرهاي محاسباتي و عملگر(=) بوجود مي‌آيند.عملگرهاي تركيبي اعمال محاسباتي و انتساب را انجام مي‌دهند.

نام عملگر	علامت	مثال
انتساب جمع	+=	a += 1
انتساب تفريق	-=	a -= 1
انتساب ضرب	*=	a *= 2
انتساب تقسيم	/=	a /= 2
انتساب باقی مانده تقسيم	%=	a%= 2

- نحوهٔ عمل این عملگرها به شکل زیر است:

- m += 8; → m = m + 8;
- m -= 8; → m = m - 8;
- m *= 8; → m = m * 8;
- m /= 8; → m = m / 8;
- m %= 8; → m = m % 8;

عملگرهاي بيتي :

علامت	نام عملگر
&	و (and)
|	یا (OR)
^	یای انحصاری (XOR)
~	نقیض(NOT)
>>	شیفت به راست
<<	شیفت به چپ

عملگرهاي متفرقه &

و * و ؟ و ، و sizeof و( )

عملگر ؟: این عملگر، عبارتی را ارزیابی کرده و بر اساس آن عبارت (درستی یا نادرستی)، نتیجه عبارت دیگر را در متغیری قرار می دهد

<عبارت ۳> : <عبارت ۲> ؟ <عبارت ۱>=متغیر

اگر <عبارت ۱> دارای ارزش درشتی باشد مقدار ارزیابی شده <عبارت ۲> در متغیر قرار می‌گیرد وگرنه مقدار <عبارت ۳> در متغیر قرار می‌گیرد.

دستورات زیر را در نظر بگیرید:

- Int x,y;
- X=3;
- Y=x>2 ? x*4 : x/3;

واضح است که y=12

تقدم عملگرها در حالت کلی

()

! ~ ++ -- sizeof

* / %

+ -

<< >>

< <= > >=

== !=

&

^

|

&&

||

?

= += -= *= /= %=

,

- a = 20-3*4+19%(3*(2+1));
- 20-3*4+19%(3*3) ابتدا عبارت داخل پرانتز به دلیل بالاترین تقدم بررسی می شود. اینجا ما دو پرانتز تودرتو داریم پس از پرانتز داخلی شروع می کنیم.

20-3*4+19%9با محاسبه مقدار اولین پرانتز، دوباره عبارت داخل آنرا بدلیل وجود پرانتز دوم و تقدم آن نسبت به دیگر عملگرها محاسبه می کنیم.

20-12+19%9 حالا از بین عملگرهای موجود * و ٪ از تقدم بالاتری برخوردارند. بدلیل یکسان بودن تقدم این دو از چپ شروع کرده و با رسیدن به هر کدام از این دو عملگر مقدار عبارت را محاسبه میکنیم که در این مثال ابتدا * محاسبه می شود.

20-12+1سپس نوبت به ٪ میرسد.

- 8+1 اکنون عملگرهای + و – در عبارت باقی می مانند که بدلیل یکسانی تقدم اولین عملگر از چپ یعنی – ابتدا محاسبه می گردد.
- ۹ و در پایان عملگر + محاسبه شده که در نهایت به جواب ۹ می رسیم.

مثال

(5+2) * (6+2*2) /2

با توجه به تقدم عملگرها داریم:

7 *(6+2*2) /2

7*(6+4) /2

7*10 / 2

35

توابع

از فایلهای سرآیند بعنوان کتابخانه های C++یاد می کنند که از قبل نوشته شده اند و ما برای استفاده از برخی از توابع و روالها از آنها استفاده می کنیم . کامپایلر فقط کلمات کلیدی را می شناسد و همانطور که گفته شد برای استفاده از یک سری دستورات و توابع مانند دستورات ورودی و خروجی و ... باید از این فایلهای سرآیند استفاده نماییم و اگر استفاده نکنیم امکان برنامه نویسی بوجود نخواهد آمد. این نکته را هم خاطر نشان می کنم که پسوند این فایلها hمی باشد.

نحوه استفاده از توابع

- برای استفاده از توابع خاصی باید نام فایل header انرا در ابتدای برنامه در دستور # include قرار دهیم.

#include < header نام فایل>

به عنوان مثال, هر برنامه که بخواهد اطلاعاتی را از صفحه کلید بخواند و یا اطلاعاتی را در صفحه نمایش چاپ کند باید فایل iostream.h را به برنامه ضمیمه کند.

#include <iostream.h>

ساختار برنامه در C++

```
#include < فایل  سرآیند >

Int main( )

{

اعلام  متغیرها

دستورات  اجرایی

Return 0;

}
```

چاپ اطلاعات با cout

Cout بری چاپ اطلاعات در صفحه نمایش به کار می رود. و در فایل iostream.h قرار دارد. Cout به صورت زیر قابل استفاده است:

Cout << عبارت ۱ >> عبارت ۲ >>...;

مثال:

Cout << “this is a book.”;

- Cout عبارت را به سطر بعدی منتقل نمی‌کند. به عنوان مثال:

```
cout << "This is a text.";
cout << "This is a book.";
```

به صورت زیر در صفحه نمایش نشان داده میشود:

This is a text. This is a book.

برای اینکه عبارتی را در چند خط نمایش دهیم، برای انتقال به هر خط جدید از علامت \n استفاده میکنیم .

مثال

برنامه‌ای که دو مقدار صحیح را در صفحه نمایش چاپ می‌کند.

```
#include  <iostream.h>
#include  <conio.h>
int main()
{
int   x=8, y=3;
cout  << “x= “<<x <<“\n”;
cout  <<“y= “<<y;
getch();
return 0;
}
```

خواندن اطلاعات با cin

cin برای خواندن اطلاعات از صفحه کلید به‌کار می‌رود و در فایل iostream.h قرار دارد و به صورت زیر به‌کار می‌رود:

cin >> متغیر ۱ >> متغیر ۲ >>...;

مثال:

```
cin >> a >> b;
```

مثال

دو عدد از نوع اعشاری را گرفته مجموع و حاصلضرب انها را چاپ کند.

```
#include  <iostream.h>
#include  <conio.h>
int main()
{
float  x,y,s,p;
cin >> x>> y;
s=x+y;
p=x*y;
cout  <<“s=“<< s<<endl;
cout <<“p=“<< p;
getch();
return 0;
}
```

```
#include <iostream.h>
#include <conio.h>
int main()
 {
 int a, b;
cout << “vared konid meghdar a ra:";
 cin >> a;
cout << “vared konid meghdar b ra:";
 cin >> b;
cout << "a:" << a << "\n";
cout << "b:" << b;
 return 0;
 }
```

مثال

سه عدد را از ورودی خوانده سپس میانگین آنها را محاسبه و چاپ کند.

```
#include  <iostream.h>
#include  <conio.h>
int main()
{
int  x,y,m;
float  miangin;
cout << “vared kon se adad:”;
cin >> x>> y>> m;
miangin=(float)(x+y+m)/3;
cout << miangin;
getch();
return 0;
}
```

کاراکترهای کنترلی که در cout استفاده می‌شوند:

\n : مکان نما را به سطر بعدی می‌برد.

\t : مکان نما را به ابتدای **هشت** محل (**ستون**) بعدی می‌برد.

\a : **بوق سیستم** را به صدا در می‌آورد.

**** : کاراکتر (\) را چاپ می‌کند.

\" : کاراکتر " را چاپ می‌کند.

\v : مکان نما را به ابتدای **هشت سطر** بعد می‌برد.

\b : کاراکتر قبل از خودش را **حذف** می‌کند.

\r : کلیدِ **Enter** را مشخص می‌کند.

\? : علامت **?** را چاپ می‌کند.

\: : علامت (:) را چاپ می‌کند.

خواندن کاراکترها با استفاده از تابع get():

تابع **get()** به صورت زیربه کار می‌رود:

```
ch=cin.get();
```

مثال: برنامه‌ای که یک کاراکتر را از ورودی خوانده در صفحه نمایش چاپ کند.

```
#include <iostream.h>
#include <conio.h>
Int main()
{
  char ch;
  cout <<“yek character vared konid:”;
  ch=cin.get();
  cout << ch;
  getch();
  return 0;
}
```

فصل دوم

- ساختارهای کنترلی

ساختار تکرار for

از دستورالعمل for برای تکرار دستورالعملها استفاده می‌شود.شکل کلی دستور for به دو صورت زیر می‌باشد.

روش اول:

```
For ( مقدار اولیه اندیس حلقه; شرط حلقه ; گام حرکت )
{
دستور 1 ;

دستور2 ;

دستور n ;
}
```

روش دوم استفاده از دستور for:

- این دستور برای ایجاد حلقه تکرار بی‌نهایت (شرط پایان ندارد) مورد استفاده قرار می‌گیرد.
- برای خاتمه دادن به اجرای حلقه بی‌نهایت از کلیدهای ctrl+Break استفاده می‌شود.

```
FOR (; ; )
{
1 دستور ;
2دستور ;
n دستور ;
}
```

چنانچه فقط یک دستور درحلقه وجود داشته باشد :

for (مقدار اولیه حلقه ; شرط حلقه ; گام حرکت)

دستور ;

for(; ;) حلقه بی نهایت

دستور ;

چنانچه فقط یک دستور درحلقه وجود داشته باشد :

for (مقدار اولیه حلقه ; شرط حلقه ; گام حرکت)

دستور ;

for(; ;) حلقه بی نهایت

دستور ;

مثال

- اعداد ۱ تا ۱۰ را به صورت نزولی چاپ کند.

```
#include  <iostream.h>
#include  <conio.h>
int main()
{
int  I;
for (i=10; i>0; i--)
cout  << i<<“  “;
getch();
return 0;
}
```

مثال

برنامه‌ای که جمله‌ای را از ورودی خوانده و تعداد حروف آن را شمارش نماید. انتهای جمله به نقطه ختم می‌شود.

```
#include <iostream.h>
#include <conio.h>
int main()
{
Int  c;
clrscr();
cout << “yek jomleh vared konid:” << endl;
for ( c=0 ; cin.get()!=‘.’ ; c++ );
cout  << c;
getch();
return 0 ;
}
```

مثال

عدد صحیح ومثبت n را گرفته وفاکتوریل آنرا محاسبه و چاپ کند.

```
#include<iostream.h>
int  main()
{
int  n,I;
long int   f=1;
cout << “yek adad sahih mosbat vared konid:”;
cin >> n;
for  (i=1;i<=n;i++)
f*=i;
cout << f;
return 0;
}
```

مثال

- مجموع اعداد صحیح و متوالی را بین ۱ تا n محاسبه و چاپ کند.

```
#include<iostream.h>
int  main()
{
int  n,i;
long int  s=0;
cout << “yek adad vared konid:”;
cin >> n;
for (i=1 ; i<=n ; i++)
s+=i;
cout  << s;
return 0;
}
```

حلقه‌های تکرار تو در تو با for

- وقتی حلقه‌ی تکراری در داخل حلقه‌ی تکرار دیگری قرار داشته باشد، می‌گوییم که حلقه‌های تکرار تودرتو ایجاد شده‌اند.
- به ازای هر بار تکرار حلقه‌ی تکرار خارجی، حلقه‌ی تکرار داخلی به طور کامل انجام می‌شود.
- انتهای حلقه‌ی تکرار داخلی، زودتر از حلقه‌ی تکرار خارجی مشخص می‌شود.

برنامه‌ای که جدول ضرب اعداد ۱ تا ۱۰ را تولید کرده و در خروجی نمایش دهد.

```
#include <iostream.h>
#include <conio.h>
int main()
{
int I,j;
clrscr();
for(i=1;i<=10;i++) {
  for (j=1; j<=10 ; j++)
    cout << i*j  << “\t”;
  cout << endl;
  }
getch();
return 0;
}
```

دستور تکرار while:

نحوه استفاده از این ساختار بصورتی است که تا زمان برقراری یک شرط خاص، مجموعه دستورات مشخص شده توسط آن اجرا شده و به دو صورت مورد استفاده قرار می‌گیرد.

1. while(شرط)
   دستور;

2. while(شرط)
   {
   دستور۱;
   .
   .
   دستورn;
   }

دستورتکرار do...while :

این دستور نیز به دو صورت مورد استفاده قرارمی گیرد.

۱.روش اول :

```
do    دستور ;
while    (شرط )
```

۲.روش دوم :

```
do {
دستور۱

دستور۲

دستورn
}
while    (شرط);
```

برنامه‌ای که جمله‌ای را از ورودی خوانده و تعداد حروف آن را شمارش نماید. انتهای جمله به نقطه ختم می‌شود.

```
#include <iostream.h>
#include <conio.h>
int main()
{
int c=0;
clrscr();
cout << “yek jomleh vared konid ke be noghteh khatm
    mishavad:” << endl;
while (cin.get() !=‘.’)
   cout++;
cout << c;
getch();
return 0 ;
}
```

برنامه‌ای که تعدادی عدد را خوانده و مجموع مربعات آنها را محاسبه نموده، به همراه تعداد اعداد در خروجی چاپ نماید.

```
#include <iostream.h>
#include <conio.h>
int main()
{
int x, sum=0, n=0;
char ans='y';
clrscr();
while (ans == 'y'){
  cout << "yek adad vared konid: ";
  cin >>x;
  sum += x*x;
  n++;
  cout  << "aya mikhahid edameh dahid? (Y/N): ";
  cin  >> ans;
  } //end of while
```

- cout <<“tedad adad hast:” << n;
- cout << sum;
- getch();
- return 0;
- }

برنامه ای که تعدادی عدد را از ورودی خوانده و وارون رقم‌های آنها را در خروجی چاپ نماید.

```
#include<iostream.h>
#include<conio.h>
int main()
{
  int num,digit;
  clrscr();
  while(1){
   cout<<“\n Enter a Number:”;
   cin>>num;
  cout<<“inverse=“;
```

ادامه برنامه

```
do {
      digit=num%10;
       cout<<digit;
       num/=10;
       } while(num!=0);
    }//end of whil(1)
   // return 0;
}
```

خروجی :

Enter a Number:26

Inverse=62

چون برنامه با کلید ctrl+Break
خاتمه می‌یابد نیازی به دستور
Return 0;نیست

مثال

- اعداد ۱ تا ۹ را در ۹ خط چاپ کند.

```
#include <iostream.h>
int main()
{
int c=0;
do
cout<< c++<<endl;
while(c<=9);
return 0;
}
```

ساختارهای تصمیم :

ساختار تصمیم if :

```
if (شرط){
  دستور۱;
  دستور۲;
   ...
  دستور n;
 }
else{
 دستور۱;
 دستور۲;
  ...
 دستور n;
 }
```

```
if (شرط)
  دستور;
else
 دستور;
```

```
if (شرط)
  دستور;
```

مثال

یک عدد اعشاری را گرفته و جذر آنرا چاپ کند.

```
#include <iostream.h>
#include <math.h>
int  main()
{
float  x,s;
cout << “yek adad vared konid:”;
cin >> x;
if  (x<0)
cout <<“ x manfi hast.”;
else{
s=sqrt(x);
cout << s;
}
return 0;
}
```

برنامه‌ای که نمره عددی دانشجویی را خوانده و معادل حرفی آن را در خروجی چاپ نماید. هر وقت نمره صفر وارد شد برنامه خاتمه یابد

```
#include <iostream.h>
#include <conio>
int main()
{
int nomreh;
clrscr();
cout <<“\n yek nomreh vared konid:”;
cin >> nomreh;
```

ادامه برنامه

```
while (nomreh){
 if (nomreh >= 17  &&  nomreh <= 20)
    cout<<“nomreh = ”<<nomreh << “ score=”<<‘A’;
 else  if (nomreh >= 15  &&  nomreh < 17)
    cout <<“nomreh = ”<< nomreh << “ score=”<<‘B’;
 else  if ( nomreh >= 12 &&  nomreh <15)
    cout <<“nomreh = ”<< nomreh << “ score=”<<‘C’;
 else if (nomreh <12)
    cout <<“ nomreh = ”<< nomreh << “ score=”<<‘D’;
 cout <<“\n yek nomreh vared konid”;
 cin >> nomreh;
 }// end of while
return 0;
}
```

مثال

برنامه‌ای که طول سه پاره‌خط را از ورودی گرفته مشخص نماید که آیا تشکیل یک مثلث می‌دهد یا نه؟

```
#include <iostream.h>
int main()
{
float  a,b,c;
cout <<“ se adad vared konid:”;
cin >> a >> b >> c;
if ((a<=b+c) && (b<=a+c) && (c<=a+b))
cout << “areh”;
else
cout << “na”;
return 0;
}
```

دستورهای break و continue:

- دو دستور فوق مسیر جریان کنترل را تغییر می‌دهند.
- break: این دستور موجب خروج از حلقه‌های تکرار می‌شود.
- continue: این دستور در حلقه تکرار موجب انتقال کنترل به ابتدای حلقه می‌شود.

مثال

- برنامه‌ای که تعدادی عدد صحیح از ورودی خوانده مجموع آنها را محاسبه و چاپ کند. آخرین عدد ورودی منفی است.

```
#include <iostream.h>
int main()
{
int  x , s=0;
While (1) {
cout << “yek adad vared konid:”;
cin >> x;
if (x<0){
```

ادامه برنامه

```
cout << “adad manfi ast.”;
break;
}
s+=x;
}
return 0;
}
```

مثال

برنامه‌ای که تعدادی عدد صحیح از ورودی خوانده و اعداد نامنفی را باهم جمع کند و این کار را تا زمانی که عدد کوچکتر یا مساوی ۱۰۰۰ است انجام دهد.

```
#include <iostream.h>
int main()
{
int  x ,s=0;
do{
cin >> x;
if (x<o)
continue;
s+=x;
```

ادامه برنامه

```
} while (x<=1000);
cout << s;
return 0;
}
```

مثال

برنامه‌ای که n عدد صحیح را از ورودی خوانده و اعداد نامنفی را جمع و میانگین ان را محاسبه کند.

```
#include <iostream.h>
int  main()
{
int  x  ,i , n , s=0;
float  a;
cin >> n;
for (i=1 ; i<=n , i++){
```

ادامه برنامه

```
cin >> x;
if (x<0)
continue;
s+=x;
}
a=s/n;
cout << a;
return 0;
}
```

ساختار تصمیم گیری switch :

درمواردی که تعداد انتخابهای یک وضعیت زیاد شود بهتر است از دستور **switch** استفاده کنیم .

```
Switch( عبارت)
{
    case  < مقدار ۱ >:
          <دستورات ۱>;
          break;
    case  < مقدار ۲ >:
          <دستورات ۲>;
          break;
             .
             .
   defult:
          < N دستورات >;
}
```

برنامه‌اي كه عددي از 1 تا 7 را خوانده، روزي از هفته را كه معادل با آن است را در خروجي چاپ كند.

```
int main()
{
int  date;
cin<<date;
switch(date)
 {
     case 1:cout<<" شنبه";   break;
     case 2:cout<<"  يكشنبه";  break;
     case 3:cout<<"  دوشنبه";  break;
     case 4:cout<<"  سه شنبه";  break;
     case 5:cout<<" چهارشنبه ";  break;
     case 6:cout<<"  پنج شنبه";  break;
     case 7:cout<<" جمعه ";  break;
     default:cout<<"Not  Valid";
 }
return 0;
}
```

مثال

```
#include <iostream>
int  main()
int  n;
cin >> n;
switch (n){
case  0:
        cout << “sefr”;
         break;
case  1:
        cout << “yek”;
         break;
case  2:
        cout << “do”;
         break;
}
return 0;
}
```

مثال

```
#include <iostream>
int  main()
int  n;
cin >> n;
switch (n){
case  0:
Case  1:
Case  2:
        cout <<  “kamtar az se”;
         break;
```

```
case  3:
        cout << “mosavi se”;
         break;
default:
        cout << “bozorgtar az se”;
}
return 0;
}
```

فصل سوم

توابع

توابع :

به مجموعه ای ازدستورات که کار خاصی را انجام می‌دهند تابع گویند.

با استفاده از توابع می‌توان برنامه‌های ساخت یافته نوشت. وظایف این نوع برنامه‌ها توسط بخش‌های مستقلی که تشکیل‌دهنده‌ی برنامه‌اند انجام می‌شود. این بخش‌های مستقل همان توابع هستند.

امتیازات برنامه نویسی ساخت یافته

۱- نوشتن برنامه‌های ساخت یافته آسان است. بعلت تقسیم برنامه به بخش‌های کوچک‌تر.

۲- همکاری بین افراد را فراهم می‌کند.

۳- اشکال زدایی برنامه‌های ساخت‌یافته ساده‌تر است.

۴- برنامه نویسی ساخت یافته باعث صرفه‌جویی در وقت می‌شود. بعلت استفاده از توابع نوشته شده در برنامه‌های دیگر.

توابع

استفلده از توابع در برنامه‌ها به برنامه نویس این امکان را می‌دهد که بتواند برنامه های خود رابه صورت قطعه قطعه برنامه بنویسد. تاکنون برنامه هایی که نوشتیم فقط از تابع main() استفاده کردیم.

- توابع سه جنبه دارند: جنبه تعریف، جنبه فراخوانی و جنبه اعلان.

جنبه‌ی تعریف تابع: مجموعه‌ای از دستورات که عملکرد تابع را مشخص می‌کنند.

جنبه‌ی فراخوانی تابع: دستوری که تابع را فراخوانی می‌کند. (فراخوانی تابع با نام آن انجام می‌شود.)

جنبه‌ی اعلان تابع: الگوی تابع، قبل از تابع main بصورت زیر تعریف می‌شود:

<نوع تابع> نام تابع (لیست پارامترها);

در استفاده از توابع در C++ موارد زیر را رعایت نمایید

۱- الگوی تمام توابع را قبل از تابع main() اعلان کنید.

۲- نوع تابع را تعیین نمایید.

۳- برای اجرای توابع، آنها را با نامشان فراخوانی نماید.

۴- متغیرهای مورد نیاز توابع را در داخل توابع تعریف کنید. (هیچ تابعی نمی‌تواند از متغیرهای توابع دیگر استفاده نماید. مگر اینکه از طریق پارامترها منتقل شوند)

۵- تعریف توابع در داخل تابع دیگر امکان‌پذیر نیست.

۶- هنگام تعریف توابع، دقت داشته باشید که تعداد و نوع پارامترها و آرگومان‌ها یکسان باشد.

۷- هنگام اعلان الگوی توابع، نیاز به ذکر اسامی پارامترها نیست.

۸- اگر تابعی فاقد آرگومان است، به جای لیست آرگومان‌ها، کلمه‌ی void را قرار دهید.

روشهای ارسال پارامترها به تابع

۱- روش فراخوانی با مقدار (Call by value)

(مقادیر آرگومان‌ها در پارامترها کپی می‌شوند.)

1. توابعی که هیچ مقداری بر نمی‌گردانند (void)
2. توابعی که یک مقدار بر می‌گردانند

۲- روش فراخوانی با ارجاع (Call by referende)

(آدرس آرگومان‌ها به پارامترها منتقل می‌شود.)

1. توابعی که چندین مقدار را باز می‌گردانند

مثال

برنامه ای که ضرایب معادله درجه دومی را از ورودی خوانده آنها را به تابعی ارسال کند. تابع معادله را حل کرده جوابهای آنرا به خروجی ببرد.

```
#include <iostream.h>
#include <math.h>
void  root(int, int, int);
int  main()
{
int a,b,c;
cout << “se adad vared konid:”;
cin >> a>> b>> c;
```

ادامه برنامه

```
root(a,b,c);
return 0;
}
void root(int a,int b,int c)
{
int  delta;
float  x1,x2;
delta=b*b-4*a*c;
if (delta==0)
{
x1=(-b)/(2*a);
x2=x1;
```

ادامه برنامه

```
cout << do rishe mozaaf darad:<<x1;
}
else if (delta >0)
{
x1=(-b+sqrt(delta))/(2*a);
x1=(-b-sqrt(delta))/(2*a);
cout << do rishe darad:<<x1<<x2;
}
else
cout << rishe nadarad.;
}
```

مثال

برنامه‌ای که دو مقدار صحیح را از ورودی گرفته به تابعی ارسال کند.
تابع max انها رامحاسبه و به برنامه ارسال کند.

```
#include <iostream.h>
int  max(int,int);
int main()
{
int  a,b;
cin >> a>>b;
cout << max(a,b);
return o;
}
```

ادامه برنامه

```
int max(int a,int b)
{
int  c;
c=(a>b) ? a : b;
return c;
}
```

مثال

برنامه ای که دو عدد اعشاری را از ورودی خوانده به تابعی ارسال کند. تابع حاصلجمع آنها را محاسبه کرده به برنامه ارسال کند.

```
#include <iostream.h>
float  m( float,float);
int main()
{
float  a,b;
cin >> a>>b;
cout <<m(a,b);
return 0;
}
```

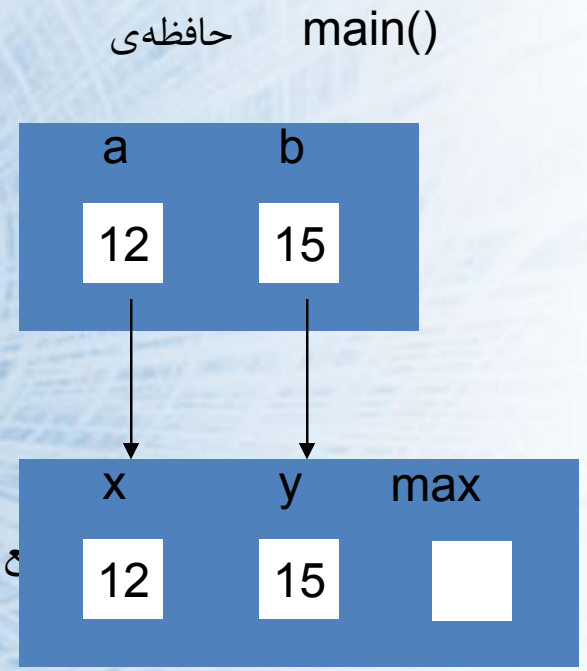
ادامه برنامه

```
float  m(float a,float b)
{
float c;
c=a+b;
return  c;
}
```

برنامه‌ای که بااستفاده ازتابع دوعددرا به عنوان ورودی دریافت کرده ومقدار بزرگتررا برگرداند.

```
#include<iostream.h>
#include<conio.h>
int fmax(int,int);
void main(void)
{
  int a,b;
  cout<<“enter a:”;
  cin>>a;
  cout<“enter b:”;
  cin>> b;
  cout<<“maximum:”<<fmax(a,b);
  getch();
}

int fmax(int x,int y)
{
    int max;
    max = x>y ? x:y;
    return(max);
}
```



خروجی برنامه :

enter a: 12

enter b: 15

maximum: 15

برنامه‌ای که در فراخوانی با مقدار، چگونگی تغییر در پارامترها و عدم تأثیر آنها را در آرگومانها نشان می‌دهد.

```
#include <iostream.h>
#include <conio.h>
void f1(int,int);
int main(){
  int x,y;
  clrscr();
  cout<< “Enter two integer numbers:”;
  cin >> x >> y;
  cout  << “you entered : x=“ << x << “, y=“ << y;
  f1(x,y);
  cout << “\nAfter return from f1: x=“ << x <<“, y=“ << y;
  getch();
  return 0;
}
```

ادامه‌ی برنامه

```
void f1(int x,int y)
 {
  cout <<“\nf1 recives : x= “<<x<<“, y=“<<y;
  x++;
  y++;
  cout<<“\nNew values in f1 :x= “<<x<<“, y=”<<y;
}
```

- خروجی

```
Enter two integer numbers: 10  15
You entered: x= 10 , y = 15
f1 recives : x = 10 , y = 15
New values in f1 : x = 11 , y = 16
After return from f1 : x = 10 , y = 15
```

متغیرهای محلی و عمومی

- متغیرهایی که در داخل تابعی تعریف می‌شوند، متغیر محلی نامیده می‌شوند و فقط در همان تابع قابل استفاده هستند.
- اگر متغیرها در خارج از توابع و در بالای تابع main() تعریف شوند، در تمام توابع موجود در برنامه قابل استفاده‌اند و متغیر عمومی نام دارند.
- یکی از تفاوت‌های متغیرهای محلی و عمومی در این است که متغیرهای محلی تا زمانی که مقدار نگرفته‌اند، مقادیر آنها معتبر نیست ولی متغیرهایی عمومی دارای مقدار اولیه صفر هستند.
- اگر در تابعی، متغیر محلی همنام با متغیر عمومی تعریف شود، در آن تابع، آن متغیر عمومی قابل استفاده نیست، بلکه از متغیر محلی استفاده می‌شود.

توابع بازگشتی

بازگشتی به مفهومی گفته می شود که در آن تابعی خودش را فراخوانی می کند.

برنامه ای که عددی مثل n را از ورودی خوانده به کمک تابع بازگشتی فاکتوریل آنرا محاسبه کند.

```
#include <ostream.h>
unsigned long fact(int);
int main()
{
int m;
cin >> m;
cout << “fact=“<< fact(m);
return 0;
}
```

ادامه برنامه

```
unsigned long fact(int x)
{
if (x !=0)
return(x* fact(x-1));
return 1;
}
```

ردیابی تابع محاسبه فاکتوریل

فراخوانی اول: چون 4!=0 است تابع به ازای x=3 فراخوانی می شود.

```
unsigned long fact(4)
{
if (4 !=0)
return(4* fact(4-1));
return 1;
}
```

فراخوانی دوم: چون 3!=0 است تابع به ازای x=2 فراخوانی می شود.

```
unsigned long fact(3)
{
if (3 !=0)
return(3* fact(3-1));
return 1;
}
```

فراخوانی سوم: چون 2!=0 است تابع به ازای x=1 فراخوانی می شود.

```
unsigned long fact(2)
{
if (2 !=0)
return(2* fact(2-1));
return 1;
}
```

فراخوانی چهارم: چون 1!=0 است تابع به ازای x=0 فراخوانی می شود.

```
unsigned long fact(1)
{
if (1 !=0)
return(1* fact(1-1));
return 1;
}
```

فراخوانی پنجم: چون 0=0 است. دستور return 1 اجرا می شود.

```
unsigned long fact(0)
{
if (0 !=0)
return(1* fact(0-1));
return 1;
}
```

توابع بازگشتی :

در برنامه نویسی ممکن است نیاز پیدا کنیم که تابعی خودش را به صورت مستقیم یا غیر مستقیم فراخوانی کند. به چنین توابعی، توابع بازگشتی گفته می شود .

مثال:

برنامهای بنویسیدکه با استفاده از تابع بازگشتی ۲۰ جمله اول دنباله زیر را در خروجی نمایش دهد.

2 5 11 23 47

```
#include <iostream.h>
long int d(long int);
int main( )
{
  for (int i=1;i<=20;i++)
   {
    cout<<d(i)<<"\t";
    if (i%5==0) cout<<endl;
   }
  return 0;
}
long int d(long int n)
{
  if (n == 1)
    return 2;
  else
    return 2*d(n-1)+1;
}
```

خروجی برنامه

2 5 11 23 47

95 191 383 767 1535

3071 6143 12287 24575 49151

98303 196607 393215 786431 1572863

کلاس‌های حافظه و حوزه متغیرها

- کلاس حافظه، ویژگیی از متغیر است که دو چیز را در مورد متغیر مشخص می‌کند:

۱- حوزه متغیر (متغیر در چه جاهایی از برنامه قابل دستیابی است.)

۲- طول عمر متغیر (متغیر کی به وجود می‌آید و کی از بین می‌رود.)

کلاس حافظه در C++

- چهار نوع کلاس حافظه در C++ قابل استفاده است:

۱- کلاس حافظه اتوماتیک

۲- کلاس حافظه ثبات

۳- کلاس حافظه استاتیک

۴- کلاس حافظه خارجی

- برای تعیین کلاس حافظه برای متغیرها، به صورت زیر عمل می‌شود:

<کلاس حافظه> <نوع متغیر> نام متغیر;

کلاس حافظه اتوماتیک

- متغیرهایی که در داخل تابعی تعریف می‌شوند (متغیرهای محلی)، با فراخوانی تابع ایجاد می‌شوند و با خاتمه اجرای تابع از بین می‌رود. کلاس حافظه این متغیرها **اتوماتیک** است.

- auto float m;

کلاس حافظه ثبات

- کلاس حافظه ثبات به کامپایلر پیشنهاد می‌کند که متغیر اتوماتیک را در ثبات پردازنده قرار دهد. بنابراین، حوزه و طول عمر متغیرهای کلاس حافظه ثبات مثل اتوماتیک است.

- register int p;

- محدودیت‌های کلاس حافظه ثبات:

- فقط برای متغیر محلی قابل استفاده است.
- انواع کاراکتر، صحیح و اشاره‌گر می‌تواند با کلاس حافظه ثبات تعریف شوند.
- تعداد اندکی از متغیرها را می‌توان با این کلاس تعریف نمود.

کلاس حافظه استاتیک

۱- متغیر استاتیک محلی (در داخل تابع تعریف می‌شود)

۲- متغیر استاتیک عمومی (در خارج تابع تعریف می‌شود)

- مقدار اولیه متغیرهای استاتیک محلی و استاتیک عمومی صفر است.

- ویژگی‌های متغیرهای استاتیک محلی

۱- فقط در همان تابعی که تعریف می‌شوند قابل استفاده‌اند.

۲- هنگام فراخوانی تابع ایجاد می‌شوند و هنگام خروج از تابع، آخرید مقدار خود را حفظ می‌کنند.

۳- فقط یک‌بار مقدار اولیه می‌گیرند.

مثال: متغیر i **در برنامه اصلی با کلاس حافظه ثبات و در تابع** test**، متغیر** x **با کلاس حافظه اتوماتیک، متغیر** y **با کلاس حافظه استاتیک تعریف شده‌اند.**

```
#include <iostream.h>
#include <conio.h>
void test(void);
int main()
{
  register int I;
  clrscr();
  for(i=0; i<5 ; i++)
    test();
  getch();
  return 0;
}
```

```
void test(void)
{
  int x=0; //automatic variable
  static int y = 0;
  cout << "\n auto x="<<x;
  cout << ", static y=" <<y;
  x++;
  y++;
}
```

خروجی

auto x = 0 , static y = 0

auto x = 0 , static y = 1

auto x = 0 , static y = 2

auto x = 0 , static y = 3

auto x = 0 , static y = 4

متغیرهای استاتیک عمومی

- در خارج از توابع تعریف می‌شوند. این متغیرها در توابعی که بعد از آنها تعریف می‌شوند قابل استفاده‌اند.

```
void  f1(void);
void  f2(void);
#include  <stdio.h>
int  main ()  {
  …
  f1();
  …
  f2();
  …
  return 0;
}
```

```
static int x , y;
void  f1(void)
{
  …
}
void  f2(void)
{
  …
}
```

کلاس حافظه خارجی

- متغیرهایی که در خارج از توابع تعریف می‌شوند دارای کلاس حافظه خارجی‌اند.

۱- با شروع برنامه ایجاد می‌شوند و تا پایان اجرای برنامه حضور دارند.

۲- در سرتاسر برنامه قابل استفاده‌اند.

- توابع همنام (لیست پارامترها، تعداد یا نوع آنها متفاوت است)

```
int myfunction(int,int);
int myfunction(long,long);
```

- آرگومان‌های فرضی

```
long myfunction (int = 50);
```

- قالب‌های تابع (انواع پارامترها و نوع تابع با T مشخص شده‌اند که T در حین اجرای برنامه تعیین می‌گردد)

```
template <class T>
T  maximum(T  p1,T  p2,T  p3)
{
 ....
 }
```

چند تابع ریاضی (الگوی توابع در فایل math.h)

- تابع abs(): محاسبه‌ی قدر مطلق اعداد صحیح
- int abs(int num)
- توابع acos()،atan()،asin(): محاسبه‌ی آرک سینوس یک عدد
- double asin(double arg)
- توابع cos()،tan()،sin(): محاسبه‌ی سینوس یک زاویه
- double sin(double arg)
- تابع sqrt(): جذر یک عدد مثبت
- double sqrt(double num)

int printf (const char * format, ...);

```
/* printf example */
#include <stdio.h>
 int  main() {
printf ("Characters: %c %c \n", 'a', 65);
printf ("Decimals: %d %ld\n", 1977, 650000L);
printf ("Preceding with blanks: %10d \n", 1977);
printf ("Preceding with zeros: %010d \n", 1977);
printf ("Some different radixes: %d %x %o %#x \n“,100,100,100,100,100);
printf ("floats: %4.2f %+.0e %E \n", 3.1416, 3.1416, 3.1416);
printf ("Width trick: %*d \n", 5, 10);
 printf ("%s \n", "A string");
return 0; }
```

خروجی برنامه

Characters: a A

Decimals: 1977 650000

Preceding with blanks: 1977

Preceding with zeros: 0000001977

Some different radixes: 100 64 144 0x64

floats: 3.14 +3e+00 3.141600E+00

Width trick: 10

A string

int scanf (const char * format, ...);

```
/* scanf example */
#include <stdio.h>
int  main () {
char  str [80];
int  i;
printf ("Enter your family name: ");
scanf ("%s",str);
printf ("Enter your age: ");
scanf ("%d",&i);
printf ("Mr. %s , %d years old.\n",str,i);
printf ("Enter a hexadecimal number: ");
scanf ("%x",&i);
printf ("You have entered %#x (%d).\n",i,i);
return 0; }
```

نتیجه‌ی اجرای برنامه

Enter your family name: Soulie
Enter your age: 29
Mr. Soulie , 29 years old.
Enter a hexadecimal number: ff
You have entered 0xff (255).

آرایه ها :

یک آرایه مجموعه ای از خانه ها متوالی حافظه می باشد که دارای یک نام و یک نوع می باشند. به هر یک از این خانه ها یک عنصر آرایه گفته می شود. برای دستیابی به یک عنصر آرایه، باید نام آرایه و شمارنده آن خانه را مشخص کنیم. لذا عناصر آرایه توسط متغیری به نام اندیس مشخص می شوند به همین دلیل، آرایه ها را

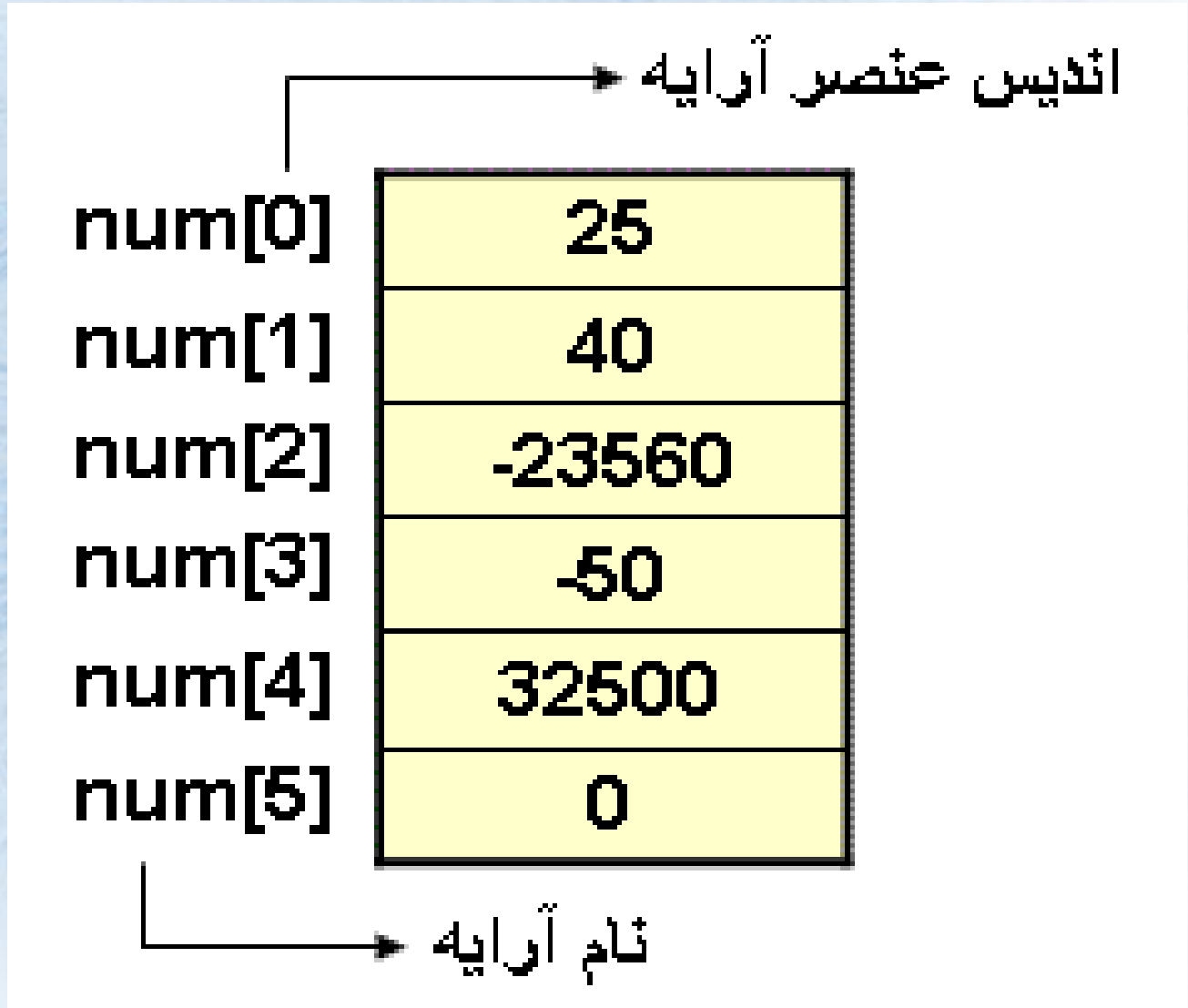
متغیرهای اندیس دار نیز می گویند نوع ارایه نیزیکی ازانواع داده ذکرشده درمبحث مفاهیم حافظه و انواع داده ای می باشد .

نحوه تعریف آرایه های یک بعدی :

نام آرایه نوع آرایه [طول آرایه];

- به عنوان مثال دستور زیر آرایه ای به طول ۶، با نام num را از نوع int ایجاد می کند.

```
int num [6];
```



آرایه های یک بعدی را می توان بعنوان آرگومان تابع ارسال کرد که در این صورت نام آرایه، به عنوان آرگومان ذکر می شود. پارامتر معادل آن هم می تواند به ۳ صورت زیر باشد :

۱- آرایه ای با طول مشخص

۲- آرایه ای با طول نامشخص

۳- اشاره گر

مثال : برنامه ای بنویسید که توسط آرایه ، نمودار میله ای افقی برای اعداد {۱ و۱۷ و۵ و۱۳ و۹ و۱۱ و۷ و۱۵ و۳ و۱۹ } رسم کند .

```
#include <iostream.h>
int main()
{
const int arraySize = 10;
int n[ arraySize ] = { 19, 3, 15, 7, 11, 9, 13, 5, 17, 1 };
 cout << "Element" << " Value" << endl;
 for ( int i = 0; i < arraySize; i++ )
 {
cout << i << "\t " << n[ i ] << "\t";
   for ( int j = 0; j < n[ i ]; j++ )
   cout << '*';     cout << endl;
  } return 0;
}
```

خروجی برنامه به صورت زیر می باشد :

```
Element  Value
0    19   ********************
1     3   ***
2    15   ***************
3     7   *******
4    11   ***********
5     9   *********
6    13   *************
7     5   *****
8    17   *****************
9     1   *
```

مرتب سازی آرایه ها

- مرتب سازی صعودی
- X[0] < x[1] < x[2] < ... < x[n]
- مرتب سازی نزولی
- X[0] > x[1] > x[2] > ... > x[n]

Exchange sort(مرتب سازی تعویضی)
- پیاده سازی آسان
- کارایی کم

مثال: مرتب سازی صعودی آرایه

- ۳ ۹ ۱ ۴

- مرحله اول مقایسه ها:
  - X[0] با x[1] یعنی (۳ با ۹) جابجایی انجام نمی شود. ۳ ۹ ۱ ۴
  - X[0] با x[2] یعنی (۳ با ۱) جابجایی انجام می شود. ۱ ۹ ۳ ۴
  - X[0] با x[3] یعنی (۱ با ۴) جابجایی انجام نمی شود. ۱ ۹ ۳ ۴
- مرحله دوم مقایسه ها:
  - X[1] با x[2] یعنی (۹ با ۳) جابجایی انجام می شود. ۱ ۳ ۹ ۴
  - X[1] با x[3] یعنی (۳ با ۴) جابجایی انجام نمی شود. ۱ ۳ ۹ ۴
- مرحله سوم مقایسه ها:
  - X[2] با x[3] یعنی (۹ با ۴) جابجایی انجام می شود. ۱ ۳ ۴ ۹

برنامه ای که تعدادی عدد از ورودی خوانده، آنها را به روش تعویضی مرتب نماید و نتیجه را در خروجی نمایش دهد.

```
#include <iostream.h>
#include <conio.h>
void ginput (int [], int);
void exchange(int [], int);
void goutput (int [], int);
int main()
{
const int k=7;
int temp[7];
clrscr();
ginput(temp, k);
exchange(temp, k);
cout<< “The sorted data are:\n”;
goutput(temp,k);
getch();
return 0 ;
}
```

ادامه برنامه

```
void ginput(int temp[], int len)
{
int i;
for(i=0;i<len;i++)
 {
   cout << “Enter number “ << (i+1) << “:”;
   cin >> temp[i];
  }
}

void exchange(int temp[],int len)
{
 int i,j,item;
 for (i=0;i<len;i++)
  for (j=i;j<=len; j++)
    if (temp[i]<temp[j])  {
        item=temp[j];
        temp[j]=temp[i];
        temp[i]=item;
       }
}
```

ادامه برنامه

```
void output(int temp[],int len)
{
  int i;
  for (i=0;i<len;i++)
   cout << temp[i]  << " ";
}
```

جستجو در آرایه

- جست و جوی ترتیبی:
  - در این روش، عنصر مورد جستجو با هر یک از عناصر آرایه مقایسه می‌گردد.
- جستجوی دودویی:
  - این جستجو در آرایه مرتب شده انجام می‌شود. در این روش، عنصر مورد نظر با عنصر وسط آرایه مقایسه شده، اگر با این عنصر برابر بود، جستجو خاتمه می‌یابد در غیر اینصورت آرایه به دو بخش تقسیم می‌گردد. اگر عنصر مورد جستجو از عنصر وسط بزرگتر بود، جستجو در بخش بالایی آرایه وگرنه در بخش پایینی انجام می‌شود. این روند تا یافتن عنصر مورد نظر یا بررسی کل آرایه ادامه می‌یابد.

برنامه‌ای که شماره دانشجویانی را از ورودی خوانده در آرایه‌ای قرار دهد، سپس شماره یک دانشجو را خوانده آنرا به دو روش ترتیبی و دودویی جست‌وجو نماید.

- در این برنامه از چهار تابع استفاده می‌گردد.
- تابع ginput() برای ورود عناصر آرایه
- تابع bsearch() برای جست‌وجوی دودویی
- تابع isearch() برای جست‌وجوی ترتیبی
- تابع exchange() برای مرتب سازی آرایه

```
#include <iostream.h>
#include <conio.h>
void ginput (int [], int);
void exchange(int [], int);
int bsearch(int [], int);
int isearch(int [], int);
int main()
{
const int k=7;
int st[k], no;
clrscr();
ginput(st, k);
cout<< “Enter a student # to search:”;
cin >> no;
cout<< “Result of linear search:”;
if (isearch(st,sk,no)==-1)
  cout<<“Numer “<<no<<“not exist in list.”;
else
  cout << “Number “ <<no << “exist in list.”;
exchange(st, k);
cout<< “Result of binarry search:”;
if (bsearch(st,sk,no)==-1)
  cout<<“Numer “<<no<<“not exist in list.”;
else
  cout << “Number “ <<no << “exist in list.”;
getch();
return 0 ;
}
```

ادامه برنامه

```
void ginput(int temp[], int len)
{
int i;
for(i=0;i<len;i++) {
  cout << “Enter number “ << (i+1) << “:”;
  cin >> temp[i];  }
}
/**********************/
int bsearch(int st[], int len, int no)
{
int mid,low,high=len-1;
while (low <= high){
 mid= (low+high)/2;
 if (no<st[mid])
   high=mid-1;
  else if (no>st[mid])
     low= mid +1;
  else return mid  }
return -1;
}
```

```
void exchange(int temp[],int len)
{
 int i,j,item;
 for (i=0;i<len;i++)
  for (j=i;j<=len; j++)
    if (temp[i]<temp[j])  {
       item=temp[j];
       temp[j]=temp[i];
       temp[i]=item;
      }
}
/**********************/
int isearch(int st[],int len,int no)
{
 int I;
 for(i=0; i<len; i++)
   if(st[i] == no)
     retuen I;
 return -1;
}
```

نحوه تعریف آرایه دو بعدی در c++:

نوع آرایه نام آرایه [بعد ۱] [بعد ۲];

مثال: Int y[3] [4];

مثال : در برنامه زیر آرایه ۲ بعدی ۱۰ در ۱۰ را با مقادیر جدول ضرب ، مقدار دهی می کنیم و سپس آن را بر روی صفحه نمایش چاپ می کنیم .

```
#include <iostream.h>
void main( )
{
  int a[10][10],i,j;
  for (i=0;i<10;i++)
    for (j=0;j<10;j++)
      a[i][j]=(i+1)*(j+1);
  for (i=0;i<10;i++){
    for (j=0;j<10;j++)
      cout <<a[i][j]<<"\t";
    cout<<endl;
  }
}
```

خروجی برنامه به صورت زیر می باشد :

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

پردازش رشته ها در c++:

رشته نوع جدیدی نیست بلکه آرایه ای از کارکترها است. رشته ها برای ذخیره و بازیابی و دستکاری متن ها در C++ بکار می روند برای تعیین انتهای رشته از کارکتر '\0' استفاده می شود. طول رشته همیشه یک واحد بیشتر از تعداد کارکترها تعریف می شود زیرا کارکتر آخر هر رشته '\0' است.

مثال : برنامه ای که طول یک رشته را به عنوان خروجی بر می گرداند.

```
#include <iostream.h>
#include <string.h>
void main()
{
char *string1 = "abcdefghijklmnopqrstuvwxyz";
char *string2 = "four";
char *string3 = "Boston";
cout << "The length of \"" << string1
    << "\" is " << strlen( string1 )
    << "\nThe length of \"" << string2
    << "\" is " << strlen( string2 )
    << "\nThe length of \"" << string3
    << "\" is " << strlen( string3 ) << endl;
}
```

خروجی برنامه :

The length of "abcdefghijklmnopqrstuvwxyz" is 26

The length of "four" is 4

The length of "Boston" is 6