فلوچارت

در گذشته برای درک بهتر الگوریتم ها و سهولت دنبال کردن دستورالعمل های آن از یکسری اشکال خاص برای نشان دادن روند الکوریتم ها استفاده می کردند که به آن فلوچارت گفته می شود. در واقع فلوچارت مجموعه ای از علائم ساده است که الگوریتم را به صورت نمادهای تصویری نمایش می دهد.

علائم فلوچارت

- **علامت شروع و پایان در فلوچارت**

از شکل بیضی برای نمایش شروع و پایان الگوریتم استفاده می کنیم

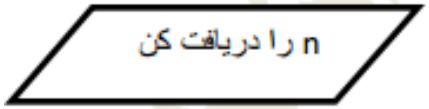
- **علامت جایگزینی و انتساب در فلوچارت**

برای انجام عمل جایگزینی و یا انتساب و یا عمل محاسباتی از مستطیل استفاده می شود

علائم فلوچارت

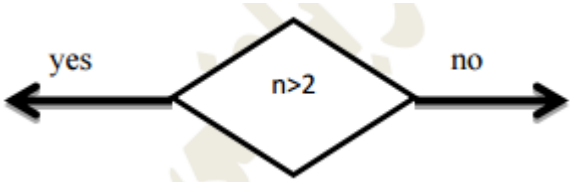
- **علامت ورودی در فلوچارت**

از علامت متوازی الاضلاع برای گرفتن ورودی ها استفاده می شود



- **علامت شرط در فلوچارت**

از علامت لوزی برای شرط استفاده می شود



علائم فلوچارت

- **علامت چاپ در فلوچارت**

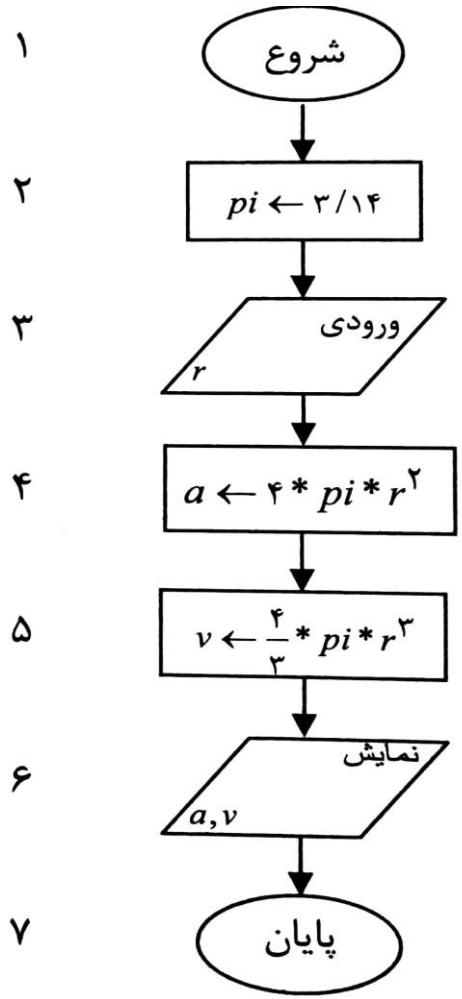
از علائم زیر برای چاپ استفاده می شود

چاپ روی کاغذ

چاپ روی صفحه نمایشگر

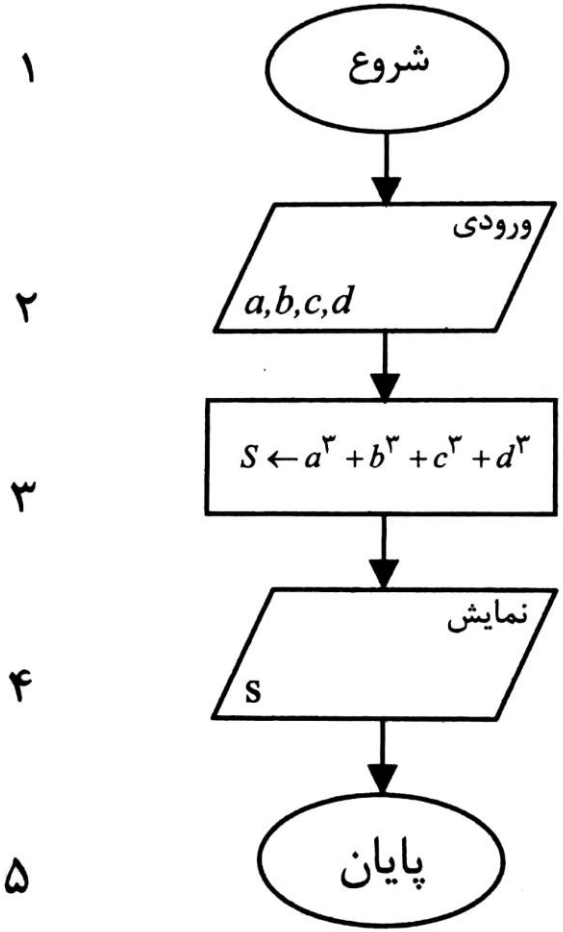
فلوچارت

فلوچارتی بنویسید که شعاع کره ای را بعنوان ورودی دریافت کند،مساخت و حجم کره را محاسبه کرده و نمایش دهد.



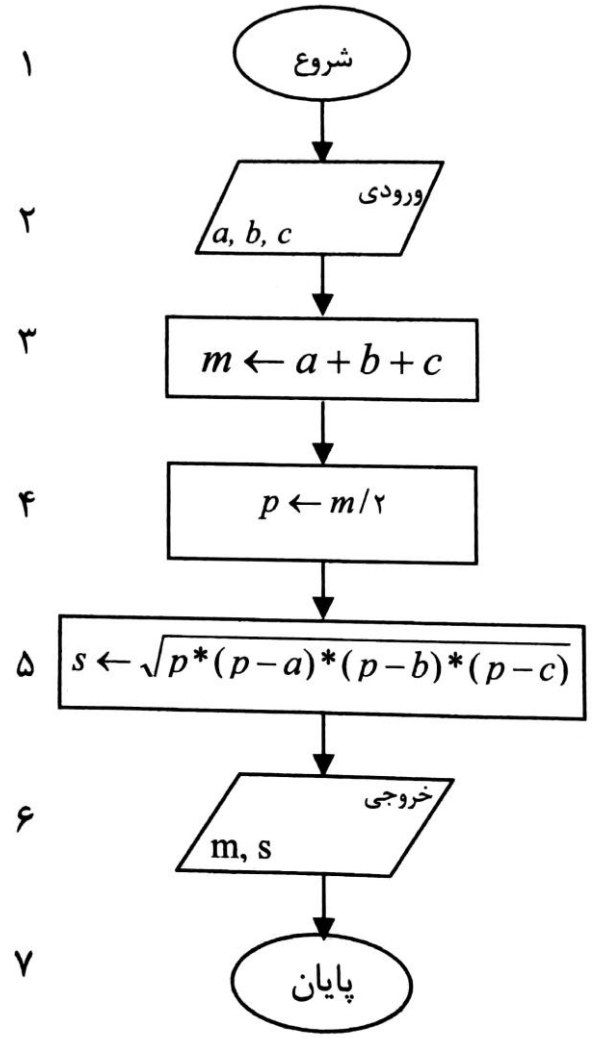
فلوچارت

فلوچارتی بنویسید که مقادیر چهار متغیر a,b,c و d را خوانده، مجموع مکعبات آنها را محاسبه کرده و نمایش دهد.



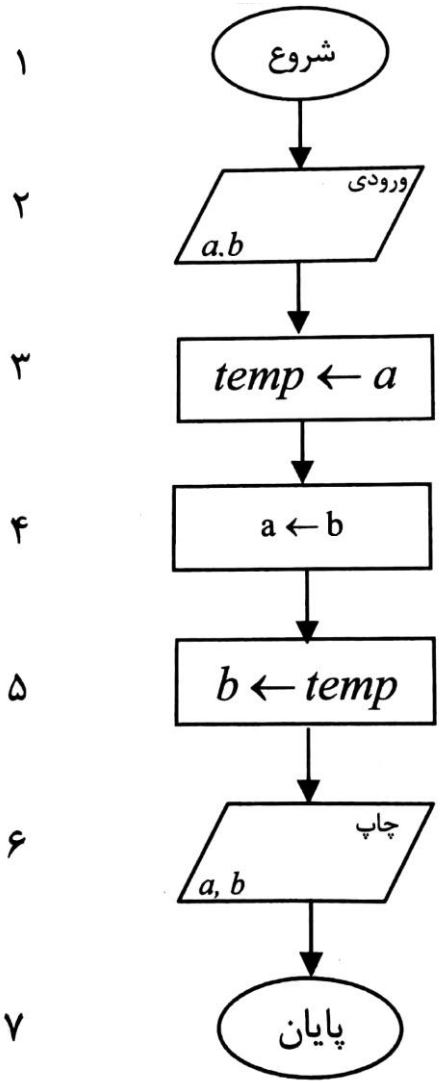
فلوچارت

فلوچارتی بنویسید که طول اضلاع یک مثلث را خوانده، محیط و مساحت آنرا محاسبه کرده و نمایش دهد.



فلوچارت

فلوچارتی بنویسید که مقادیر دو متغیر را از ورودی خوانده، سپس مقادیر آنها را با یکدیگر تعویض نماید و نتیجه را چاپ کند.



آشنایی با c++

C++ یک زبان portable است. زبان های قابل حمل یا مستقل از متن زبان هایی هستند که به هیچ ماشین خاصی وابسته نیستند، برنامه های نوشته شده با این زبان ها تا **حد زیادی** قابل حمل می باشند. در مقابل زبان های قابل حمل زبان های غیر قابل حمل وجود دارند که این زبان ها وابسته به سخت افزاری که برروی آن نوشته شده اند می باشند که از جمله این زبان ها می توان به زبان ماشین و اسمبلی اشاره کرد.

همانطور که گفتیم C ++توسعه یافته زبان Cاست که علاوه بر ویژگی های زبان Cدارای خاصیت شی گرایی نیز هست.

نحوه اجرای یک برنامه در C++

همانطور که گفتیم برای اینکه الگوریتمی که برای حل یک مسئله طرح کردیم برای اینکه توسط یک سیستم کامپیوتری اجرا شود نیاز دارد آن را ابتدا با استفاده از یک زبان برنامه سازی که اکثرا از زبانهای سطح بالا استفاده می شود نوشته شود که به این برنامه در این مرحله کد اصلی گفته می شود و پس از اینکه این برنامه به زبان ماشین برگردانده شد توسط کامپیوتر قابل فهم و اجرا است.

اجرای یک برنامه به زبان C ++شامل ۶ مرحله است:

۱. مرحله تولید برنامه:

در این مرحله برنامه توسط یکی از برنامه های ادیتوری به زبان C ++نوشته می شود و بعد از این مرحله برنامه با پسوند های .cc ،.cxx ،.cpp یا Cدر یکی از مکان های ذخیره سازی اطلاعات ذخیره می شود.

۲. پیش پردازش

پیش پردازنده ها شامل دستوراتی هستند که توسط کامپایلر قبل از شروع به کامپایل انجام می شوند. هر دستوری در C ++که با علامت # شروع شود جزو دستورات پیش پردازنده هستند و زمانی که بخواهیم از کتابخانه خود C ++استفاده کنیم از پیش پردازنده ها استفاده می کنیم،

۳. کامپایل:

در این مرحله کامپایلر دستورات کد اصلی را به زبان ماشین ترجمه می کند.

۴. پیوند دهنده :

هنگام برنامه نویسی در C ++ این امکان وجود دارد که از توابع کمکی نوشته شده در برنامه های دیگر استفاده کرد که این توابع باید به برنامه کد اصلی پیوند زده شود که با استفاده از پیونددهنده این امکان فراهم می شود.

۵. بارگذاری:

برای اینکه یک برنامه در کامپیوتر قابل اجرا باشد باید ابتدا در حافظه اصلی بارگذاری شود که اینکار با استفاده از بارگذار کننده صورت می گیرد.

۶. اجرا:

در مرحله آخر کامپیوتر با استفاده از CPU دستورات برنامه نوشته شده را اجرا می کند.

کلیه مراحل گفته شده در بالا در یک IDEبطور کامل پس از اجرای برنامه انجام می گیرد. علاوه براین، محیط ها IDEمعمولا دارای امکانات اشکال زدایی برنامه (Debug) نیز می باشند که شامل مواردی همچون اجرای خط به خط برنامه و یا دیدن محتویات متغیرها در زمان اجرا است

مراحل گفته شده در بالا برای اجرای یک برنامه است ولی اکثر مواقع اجرای برنامه ها برای بار اول دارای خطاهای اجتناب ناپذیری هستند خطاها از لحاظ تاثیری که بر اجرای برنامه ها می گذارند، متفاوتند. گروهی ممکن است باعث شوند که از همان ابتدا برنامه اصلا کامپایل نشود و گروه دیگر ممکن است پس از گذشت مدتها و در اثر دادن یک ورودی خاص به برنامه، باعث یک خروجی نامناسب و یا یک رفتار دور از انتظار (مانند قفل شدن برنامه) شوند.

بطور کلی خطاها به دو دسته تقسیم می شوند:

۱. خطاهای نحوی (خطاهای زمان کامپایل):

این خطاها در اثر رعایت نکردن قواعد دستورات زبان C ++و یا تایپ اشتباه یک دستور بوجود می آیند و در همان ابتدا توسط کامپایلر به برنامه نویس اعلام می گردد. برنامه نویس باید این خطا را رفع کرده و سپس برنامه را مجددا کامپایل نماید. لذا معمولا این قبیل خطاها خطر کمتری را در بردارند.

۲. خطاهای منطقی (خطاهای زمان اجرا):

این دسته خطاها در اثر اشتباه برنامه نویس در طراحی الگوریتم درست برای برنامه و یا گاهی در اثر درنظر نگرفتن بعضی شرایط خاص در برنامه ایجاد می شوند. متاسفانه این دسته خطاها در زمان کامپایل اعلام نمی شوند و در زمان اجرای برنامه خود را نشان می دهند. بنابراین، این خود برنامه نویس است که پس از نوشتن برنامه باید آن را تست کرده و خطاهای منطقی آن را پیدا کرده و رفع نماید. متاسفانه ممکن است یک برنامه نویس خطای منطقی برنامه خود را تشخیص ندهد و این خطا پس از مدتها و تحت یک شرایط خاص توسط کاربر برنامه کشف شود. به همین دلیل این دسته از خطاها خطرناکتر هستند.

خود این خطاها به دو دسته تقسیم می گردند

a) خطاهای مهلک: در این دسته خطاها کامپیوتر بلافاصله اجرای برنامه را متوقف کرده و خطا را به کاربر گزارش می کند. مثال معروف این خطاها، خطای تقسیم بر صفر می باشد.

a) خطاهای غیرمهلک: در این دسته خطا، اجرای برنامه ادامه می یابد ولی برنامه نتایج اشتباه تولید می نماید. به عنوان مثال ممکن است دراثر وجود یک خطای منطقی در یک برنامه حقوق و دستمزد، حقوق کارمندان اشتباه محاسبه شود و تا مدتها نیز کسی متوجه این خطا نشود!

یک نمونه برنامه در++c

در این قسمت برای آشنایی با زبان برنامه نویسی C ++یک نمونه برنامه ساده آن را قرار داده ایم که با استفاده از آن چندین نکته مهم در رابطه با برنامه نویسی به زبان C ++را توضیح می دهیم.

```
// This is a program in C++
#include <iostream>
/* allows program to
 output data to the screen */

using namespace std;

// function main begins program execution
 Void  main()
 {
        cout << "Welcome to C++!\n ";        // display message

  } // end function main
```

خروجی برنامه:

```
Welcome to C++!
```

یک نمونه برنامه در++c

همانطور که در خروجی مثال بالا می بینید که نماد \n که در انتهای رشته قرار گرفته است در خروجی نمایش داده نشده است. وقتی (\) Backslashدر انتهای یک رشته قرار گیرد و کاراکتری که بعد آن قرار بگیرد یک کاراکتر خاص است.

```
#include <iostream> // allows program to output data to the screen
using namespace std;
// function main begins program execution
int main()
{
    cout << "Welcome\n to\n\n C++!\n";
} // end function main
```

خروجی برنامه:

```
Welcome
to

C++!
```

یک نمونه برنامه در++c

بعضی کاراکتر ها هستند که به همراه \ عملیات خاصی را انجام می دهند که در جدول زیر آنها را توضیح می دهیم.

نماد	توضیحات
\n	باعث می شود کرسر به خط جدید برود
\t	باعث می شود موقعیت کرسر به اندازه یک tab جلوتر رود
\a	بوق کامپیوتر
\\	کاراکتر \ را چاپ می کند
\'	کاراکتر ' را چاپ می کند
\"	کاراکتر " را چاپ می کند

مفاهیم اولیه در++c

متغیر: نامی است که به یک قسمت از برنامه مانند متغیر، تابع، ثابت و یا ... داده می شود. در زبان ++ Cبرای انتخاب شناسه ها فقط می توان از علائم زیر استفاده کرد:

- حروف انگلیسی کوچک و بزرگ (A...Z a...z)
- ارقام (0,1,...,9)
- علامت خط پایین یا_

البته یک متغیر نمی تواند با یک رقم شروع شود. مثال شناسه های number_one، number1 ، numberمجاز هستند اما شناسه های number one، 1numberمجاز نیستند. البته در برنامه نویسی امروزی پیشنهاد می شود بجای متغیرهایی همانند number_one از numberOneاستفاده گردد. یعنی بجای اینکه در شناسه ها از _ برای جداکننده استفاده کنیم، اولین حرف هر قسمت را بصورت بزرگ بنویسیم.

مفاهیم اولیه در++c

این مسئله معمولا درهنگام برنامه نویسی باعث ایجاد بعضی خطاها می شود.

نکته: زبان C++ هیچ محدودیتی در طول شناسه ها قرار نداده است

نکته: بهتر است از شناسه های با معنی در برنامه هایمان استفاده کنیم

نکته: بهتر است از اختصار استفاده نکنیم تا درک عملکرد برنامه برای دیگران قابل فهم باشد.

نکته: در هنگام انتخاب شناسه نمی توانید از کلمات کلیدی که برای منظورهای خاص در زبان C++ رزرو شده اند استفاده کنید.

یک نمونه برنامه در++c

auto	break	case	char	const
continue	default	do	double	else
enum	extern	float	for	goto
if	int	long	register	return
short	signed	sizeof	static	struct
switch	typedef	union	unsigned	void
volatile	while	and	and_eq asm	bitand bitor
bool	catch	class	compl	const_cast
delete	dynamic_cast	explicit	export	false
friend	inline	mutable	namespace	new
not	not_eq	operator	or	or_eq
private	protected	public	reinterpret_cast	static_cast
template	this	throw	true	try
typeid	typename	using	virtual	wchar_t
xor	xor_eq			