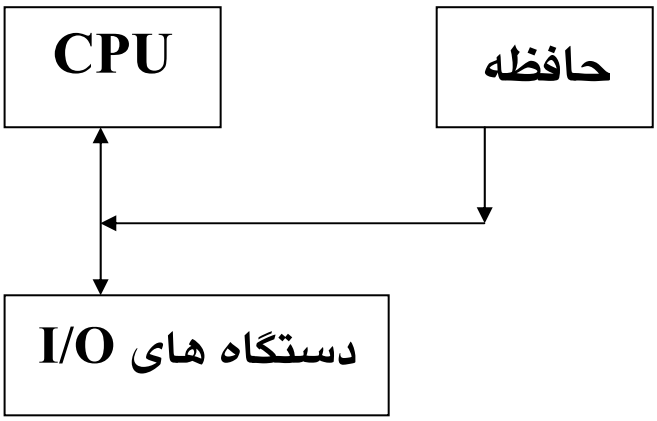
Gpc(General Purpose Computer) : کامپیوترهایی هستند که برای منظورهای گوناگون بکار می روند.

Spc(Special Purpose Computer) : کامپیوترهایی که برای مقاصد ویژه ای طراحی و ساخته می شوند.

کامپیوترهای Gpc شامل CPU و دستگاه های I/O و حافظه (اصلی) می باشند.



بخشی از دستگاه I/O را که شامل دیسک هاست را **حافظه ی جانبی** گویند.

I/O:
- Disks & Tape
- Keyboard

حافظه در Gps:
- **ROM** :**R**ead **O**nly **M**emory.محتویات با خاموش شدن از بین نمی روند
- **RWM** :**R**ead **W**rite **M**emory.محتویات با خاموش شدن از بین می روند

حافظه ی دسترسی اتفاقی RAM: Random Access Memory

هرکدام از بایت های حافظه ی RAM را بخواهیم به طور اتفاقی پس از یک T_A محتویات آن حافظه در اختیار ما قرار خواهد گرفت. Time Of Access = T_A

TA = مقدار ثابت → Random Access Memory

نقطه مقابل RAM ← SAM

حافظه ی دسترسی ترتیبی SAM = Sequential Access Memory

تمام سی دی ها و هاردسک ها و فلاپی ها جزء SAM هستند.

$T_A=F_{(A)}$

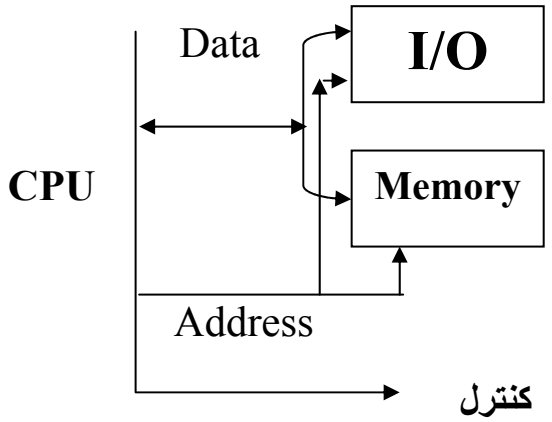
RAMوRom مخالف هم نیستند.

در مقابل Rom ، RWM قرار دارد.

RWM(Read Write Memory) ---------- حافظـــه خوانـــدنیROM هم خواندنی و هم نوشتنی

مربوط به قسمت
Address

0	1	2	3
00111111	11111111	00000000	
4	5	6	7



آدرس: CPU با مقداردهی به خطوط آدرس می گوید که با چه خانه ای از حافظه یا دستگاه های I/O کار دارد.

داده : cpu با کمک این خطوط داده را دریافت کرده یا تحول میدهد.

کنترل : اعلام می کنند که cpu می خواهد بخواند یا بنویسد.

تفاوت I/O با حافظه : در حافظه data و برنامه داریم ولی در I/o فقط data داریم.

برنامه برای اجرا شدنباید حتماً داخل حافظه باشد(زیرا cpu فقط از داخل حافظه می تواند برنامه را دسترسی کند)

I/o برای تبادل داده است(کارش فقط ورود و خروج داده است.)

برنامه عبارت است از تقسیم شده ی یک عمل پیچیده به بخش های کوچک که هر یک از این بخش ها به تنهایی توسط کامپیوتر شناخته شده است.

شناخته شده یا ناشناس بودن(انجام پذیر یا انجام ناپذیر بودن) یک عمل برای کامپیوتر بستگی به سیستم سخت افزاری و نرم افزاری ما دارد.

ریزشده و تقسیم بندی شده ی یک کاری می توان از آن براحتی برنامه ای را استخراج کرد «**الگوریتم**» نام دارد.

برنامه برای اجرا شدن باید در حافظه ی کامپیوتر جای گیرد.

داده های مورد پردازش توسط برنامه نیز باید در حافظه جای گیرند.

دستور: یک عبارت دارای معنی صریح که توسط برنامه نویس و کامپیوتر فهمیده می شود.

داده : عبارت است از اعداد – حروف – نشانه ها و بطورکلی کمیت ها.

۱- هر کدام از خانه ها را یک Word(کلمه) گویند.
۲- به یک ها و صفرها ، bit گفته میشود.
۳- به ۸ بیت یک بایت گفته می شود.
٤- به روشی که براساس آن داده ها تبدیل به یک و صفرها مـی کنـیم کدینگ می گوییم(coding)

یکی از کدها برای کدینگ حروف الفبا
American Standard Code for Information)ASCII
(Interchange
EBCDIC

برنامه برای اجراشدن باید در حافظه ی کامپیوتر جای گرد برا این کار ما مراحل زیر را باید طی کنیم:

کدماشین → اسمبلر → اسمبلی → کامپایلر → برنامه(زبان برنامه نویسی) → کاربر

برنامه با کامپایلر ترجمه می شود (معمولاً یک برنامه ی کوچک تبدیل به یک برنامه بزرگ به زبان اسمبلی می شود اگرچه حجم زیاد می شود ولی بیان ساده تر می شود)

روشهای کدینگ اعداد:

۱- تبدیل به عدد صحیح دودویی
۲- تبدیل به فرمت های اعشاری

بوت(boot strap) ، سیستم عامل ، برنامه کاربردی

بوت شدن: عبارت است از اجرای برنامه برنامه های اولیه ایی که میزبان برنامه های دیگر را به عهده دارد و کامپیوتر را برای دریافت دستورهای کاربر آماده می کند.

۱- پردازنده به آدرس از پیش دانسته شده ایی از حافظه رجوع کرده و روتین های اولیه را از آنجا اجرا می کند. آدرس از پیش دانسته شده (قراردادی بین سازنده پردازنده و سازنده کامپیوتر است و کامپیوتر از همان ابتدا با این آدرس ها آشنایی دارد.)

Rom BIOS → Basic Input Output System

بعنوان مثال پردازنده Z80 آدرس 0 را شروع میکند.
و پردازنده های 8088 و80286 و 386 و 486 آدرس FFFF0 را شروع میکنند.

۲- روتین POST انجام میشود و کامپیوتر اجزا گوناگون خود را آزمایش میکند تا از درستی آنها مطمئن شود.
(آزمون از خود هنگام روشن شدن)Power On Self Test

۳-روتین های سرویس دهی به سیستم عامل از درون ROM اجرا شده یا اگر لازم باشد در جاهای مناسب از RAM قرار داده میشود.

٤-با بکارگیری Boot Device ها به دنبال سیستم عامل برای Load شدن در حافظه می گردد و در صورت یافتن آن ، آن را به حافظه می آورد.

۵- پس از بار شدن (load) سیستم عامل(OS) در حافظه کنترل کامپیوتر بدست OS داده میشود.

DOS → 14 K byte
WINDOWS XP → 300-400 MB

Operation System(OS) : برنامه ایست که تمام امکانات و توانایی های دستگاه رایانه را در اختیار دارد و وظیفه سرویس دهی به کاربران را برعهده دارد. همچنین OS های کامپیوترهای GPC باید امکاناتی فراهم کنند که برنامه های کاربران با بکارگیری این امکانات اجرا میشوند.

روش های coding

coding
- Numerical
  - Integer اعداد درست
  - float اعداد اعشاری
- Alphabetical(نشانه ها - حرفی)

روش کدگذاری عددی درست:

۱- روش مقدار absolute (روش بی علامتunsigned)

در این روش ارزش دودویی (باینری)عدد در تعدادی بیت که تشکیل یک خانه حافظه میدهند ذخیره میشود.

نمونه: ذخیره ی عدد ۵۷ در ۸ بیت؟

0	0	1	1	1	0	0	1

111001

$$\begin{array}{l|l} 57 & 2 \\ \hline 1 & 28 \end{array} \quad \begin{array}{l|l} 28 & 2 \\ \hline 0 & 14 \end{array} \quad \begin{array}{l|l} 14 & 2 \\ \hline 0 & 7 \end{array} \quad \begin{array}{l|l} 7 & 2 \\ \hline 1 & 3 \end{array} \quad \begin{array}{l|l} 3 & 2 \\ \hline 1 & 1 \end{array}$$

اگر خانه ها زیاد آمد اضافه ها را صفر میگذاریم اگر خانه ها کم آمد نمی توانیم ذخیره ی عدد را نمایش دهیم و باید تعداد بیتها را افزایش دهیم.

اعداد را از راست به چپ در بیت ها می گذاریم.

نمونه: برگردان عدد 111001 به پایه 10 ؟ (با استفاده از جدول ارزش مکانی)

32	16	8	4	2	1
1	1	1	0	0	1

(1*32)+ (1*16)+ (1*8)+ (0*4)+ (0*2)+ (1*1)=57

۲-روش مقدار و علامت

Sign-value

Sign-magnitude

+57	0	0	1	1	1	0	0	1
57 -	1	0	1	1	1	0	0	1

۳-روش مکمل ۲ two's Complement

عدد + بصورت همان روشهای گذشته نمایش داده میشود.

عدد – بصورت مکمل ۲ عدد نمایش داده میشود.

عدد – ۲ (تعداد بیت) = مکمل ۲ عدد

نمونه : مکمل ۲ عدد ۵۷ را بدست آورید.

توضیح: تا اولین ۱ که رسیدیم دست نخورده می ماند و بعد از آن همه ی بیت ها برعکس می شوند (یک ها را به صفر و صفر ها را به یک تبدیل می کنیم.)

111001 → 000111

2^6-57=64-57=7

نمونه:

011100110→ 100011010

011100110← 100011010

نکته : چنانچه مکمل ۲ یک عدد را مجدداً مکمل ۲ کنیم همان عدد ابتدایی بدست می آید زیرا:

r^n-(r^n-عدد)= r^n – r^n + عدد= عدد

تفریق مکمل ۲ :

33-14=33-(مکمل ۲ عدد ۱٤)

بیشینه(max) حالت های قابل نمایش در n بیت برابر 2^n است.

بنابراین در روش absolute می توان در n بیت اعداد 0 تا 2^n-1 را نمایش داد.

در روش های علامت دار می توان 0 و 2^{n-1} -1 عدد مثبت و 2^{n-1} عدد منفی را نمایش داد.

نمونــه: در ۸ بیــت چــه اعــداد مثبــت و چــه اعــداد منفــی و چــه اعــداد unsigned را می توان نمایش داد.

127 تا 1 و 128- تا 1,0- مکمل ۲ , $2^8-1=255$ 0-255 unsigned

٤-روش power(نما) و mantice(پایه) در نمایش اعداد اعشاری

در این روش ابتدا عدد ممیز دار را به اصطلاح نرمال می کنیم یعنی بـه گونه ایی در توانی از ۲ ضرب می کنیم که همه رقـم هـای پـشت ممیـز (سمت چپ) صفر باشند و اولین رقم جلوی ممیز (سمت راسـت) حتمـا ۱ باشد.

111101001101 → 01110101101

ایـن عـدد را در 2^{-6} ضـرب کـرده ایـم اگـر در 2^{-7} ضـرب کنـیم عـدد 011110101101 بدست می آید که نرمال نیست.

نمونه: در یک ذخیره سازی ممیـزدار ۳۰ بیـت(یـک بیـت بـرای علامـت است) برای پایه و ۱۰ بیت(یک بیت برای علامت اسـت) بـرای تـوان در نظر گرفته ایم کوچکترین و بزرگترین عددهای مثبتی که مـی تـوان بـه این روش نمایش داد کدام است؟(مبنا ۲ است.)

توان: 2^9-1 تا -2^9+1 -511 تا 511

مانتیس(پایه): 0.1 تا 0.111111.....1

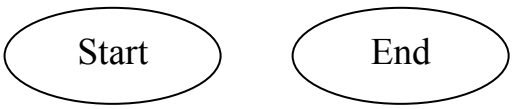
کوچکترین: $0.1 * 2^{-511}$

بزرگترین: $0.1111....1 * 2^{511}$

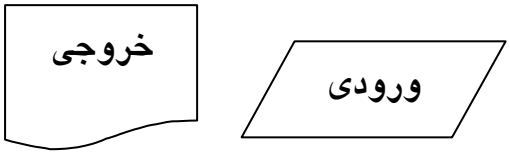
$11111....1 * 2^{482}$

فلوچارت: عبارت است از نمایش یک الگوریتم بصورت علایم که به ما کمک می کند که از درست بودن و مناسب بودن یک الگوریتم مطمئن شویم.

علامت های زیر در فلوچارت به کار می روند:

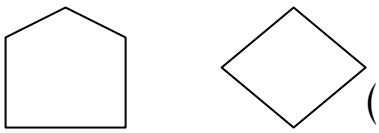


۱- نشانه های آغاز و پایان(بصورت بیضی)



۲- نشانه های ورودی و خروجی

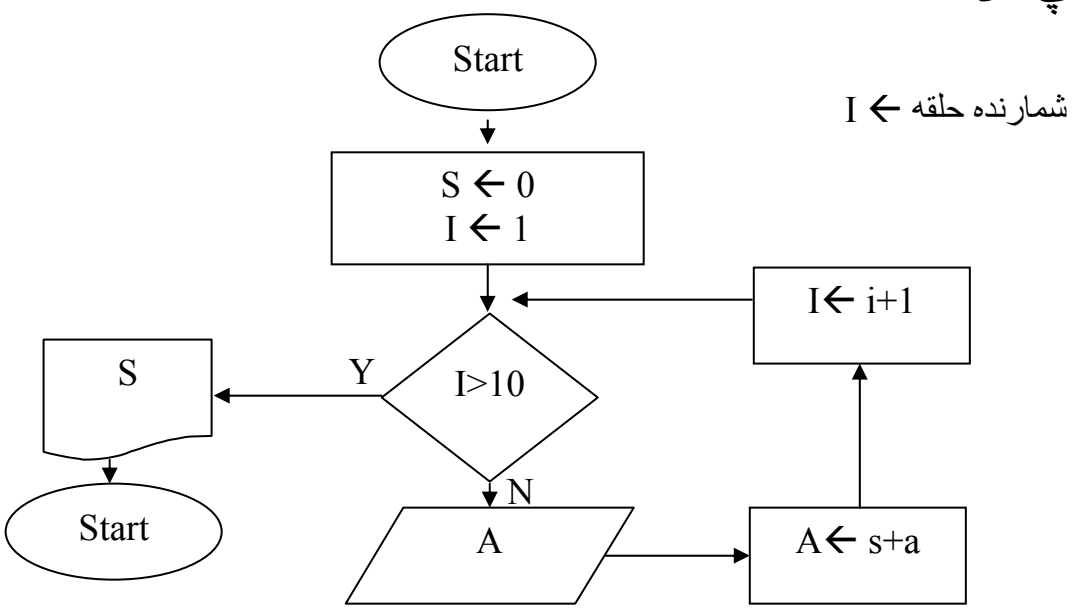
۳- نشانه عملیات



٤- نشانه تصمیم گیری(لوزی و n ضلعی)

۵-خطوط و فلش ها: که نشان دهنده مسیر دنبال کردن یک فلوچارت و جهت دنبال کردن آن .

نمونه: فلوچارتی رسم کنید که ۱۰ عدد از ورودی گرفته و جمع آنها را چاپ کند.



گونا(متغیر- variable) : یک جای خالی است که ما تنها نام یا آدرس آن را میدانیم ، مقدار آن در هنگام اجرای برنامه یا پیمایش فلوچارت مشخص میشود.

نسبت دهی(assignment): عبارت است از ریختن یک مقدار در یک متغیر (گونا) و با علامت روبرو نمایش داده میشود. مقدار ← متغیر

معنی آن اینست که مقدار نزدیک دم فلش در متغیر نزدیک نوک فلش ریخته می شود و مقدار قبلی متغیر با این مقدار جدید جایگزین میشود.

نکته: در رابطه با مقدار دهی عبارتهای زیر غلط هستند:
۲۴۵ ← a ۷ ← ۳

ثابت ها(اندازه ها - مقدارها): عبارتند از همه ی المان هایی که برای ما کاملاً شناخته شده هستند و کمیت یا چندی(مقدار) آنها دانسته شده است.

نکته : نمی توان گفت متغیری که ما مقدار آن را میدانیم ثابت است مگر اینکه مقدار آن هرگز تغییر نکند که در آن صورت متغیر نیست .

علامت های ریاضی: مانند + ، - ، * و ...

علامت های منطقی: علامتهایی هستند که میتوانند در ترکیب با متغیر ها ، ثابت ها و علامتهای ریاضی برای ما پاسخ «بله» یا «خیر» تولید کنند.
مانند > ، < ، = ، =< ، => ، < >

پیمایش یک فلوچارت(دنبال کردن یا trace): عبارت از اجرای گام به گام کارهای نمایش داده شده در فلوچارت است که معمولاً برای بررسی درستی یک فلوچارت انجام میشود.

بعنوان مثال نمونه فلوچارت قبل را trace می کنیم:

اعداد انتخابی(از ابتدا به انتها) ٤،١،٥،٢،٢،٢،١٣،١١،٩،٥

ابتدا جدولی از متغیرها کشیده و از نقطه آغاز فلوچارت شروع کرده و عدد میدهیم.

S	I	A
0	1	-
5	2	5
14	3	9
25	4	11
...	...	....
	11	4

نکته:

١- هنگام پیمایش تنها و تنها به دستورات فلوچارت عمل می کنیم.

٢- هنگام پیداشدن سردرگمی در فلوچارت باید فلوچارت را اصلاح کنیم.

٣- ترتیب دستورات فلوچارت را باید عمل کنیم.

٤- مقدارها و تصمیم گیری ها را با رجوع هرباره به جدول انجام دهیم و از تصمیم گیریهای ذهنی خودداری کنیم.

٥- معمولاً در فلوچارت حلقه هایی وجود دارد اولین و اخرین مقدار شمارنده حلقه بسیار مهم هستند و باید به آنها دقت کنیم.

نکات راجع به فلوچارت:

١- حتماً باید آغاز و پایان داشته باشد.

٢- بیضی آغاز باید یک خروجی داشته باشد و ورودی ندارد.

٣- بقیه جعبه ها بجز آغاز می توانند چندین ورودی داشته باشد و هیچیک بجز شرط نمی توانند دارای بیش از یک خروجی باشند.

٤- شرط و به طورکلی ساختار فلوچارت نباید به گونه ایی باشد که نتوان تصمیم گیری کرد که مسیر درست کدام است.

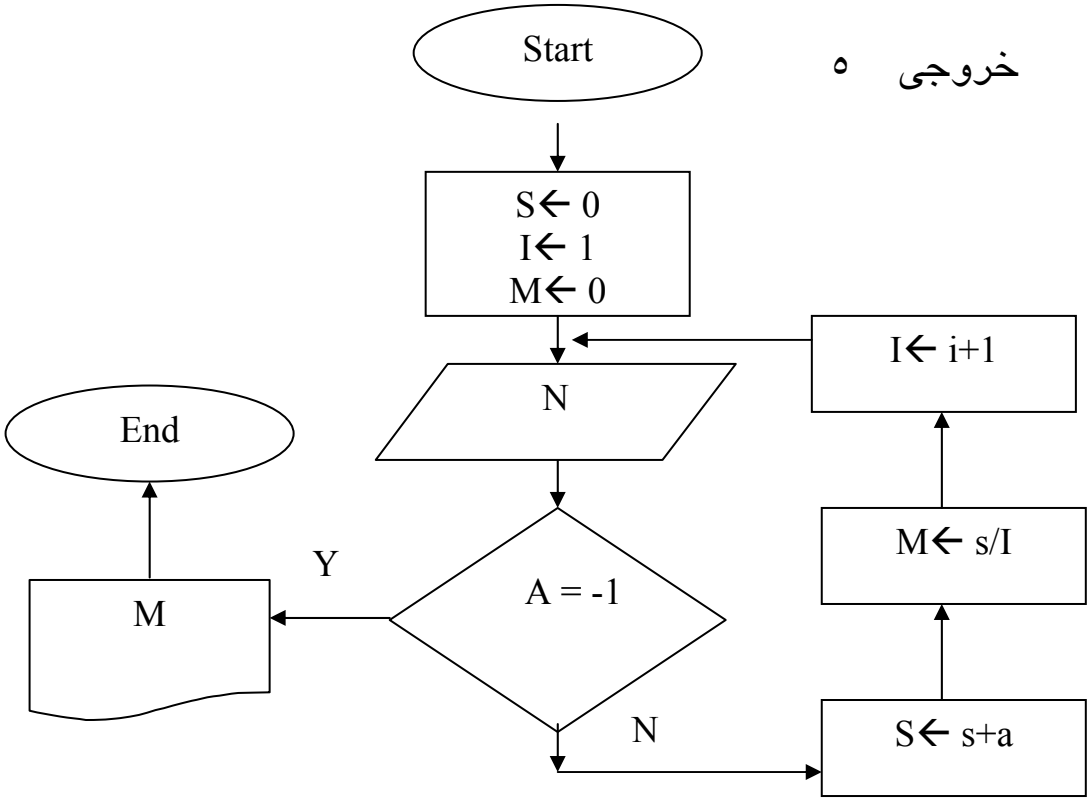
٥- بهتر است اعمال گفته شده در فلوچارت به گونه ایی باشند که به سادگی به دستورهای کامپیوتری تبدل شوند.

فلوچارتی رسم کنید که تعدادی عدد را از ورودی خوانده و میانگین آنها را چاپ کند(اخرین عدد 1- جزو عددها نیست)

اعداد انتخابی 2,4,6,8,-1

A	S	I	M
-	0	1	0
2	2	2	2
4	6	3	3
6	12	4	4
8	20	5	5
-1			

خروجی ٥



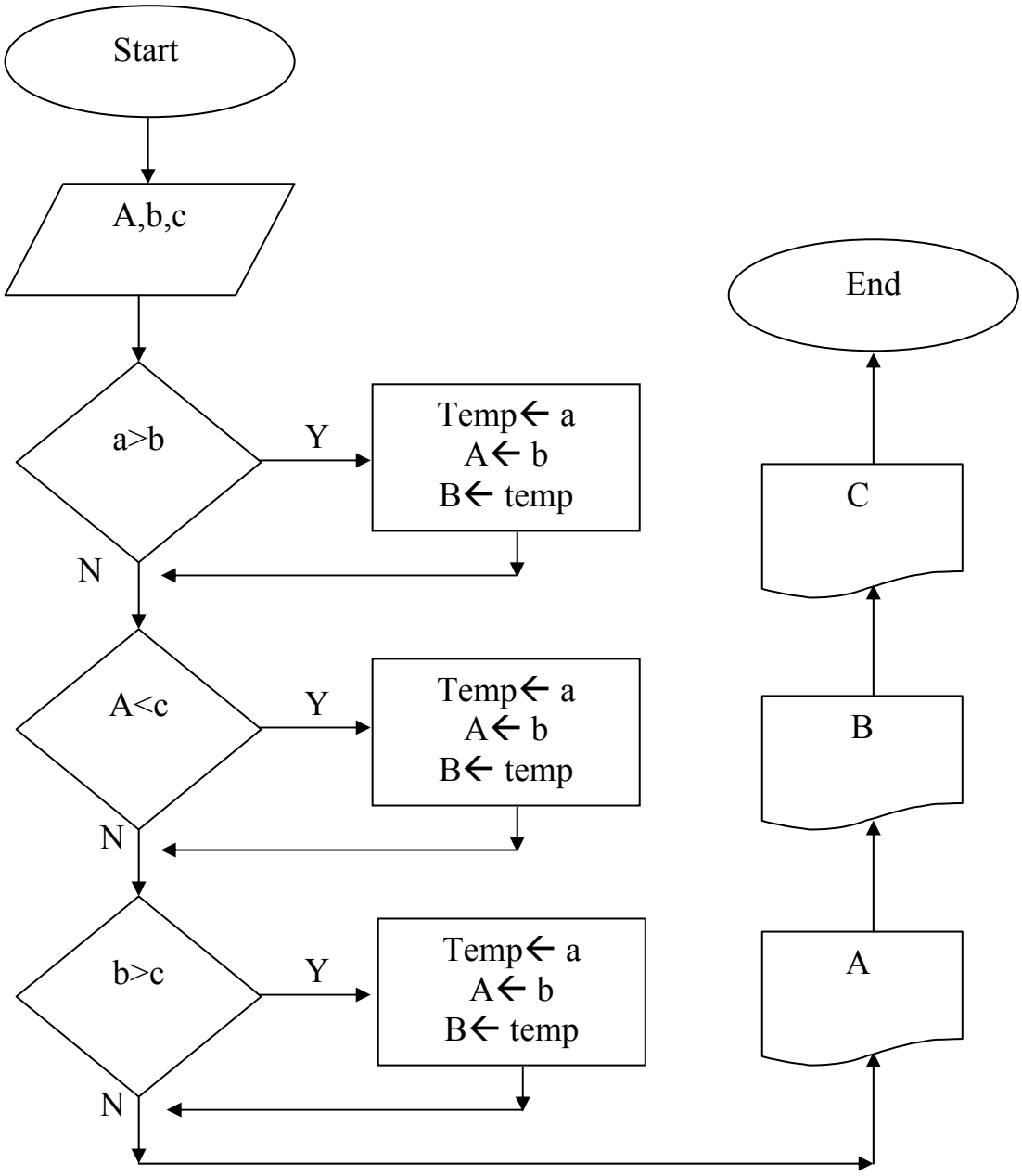
DESIGN TIME: مشکلاتی که در طراحی وجود دارند و در هنگام طراحی باعث سردر گمی طرح میشوند

RUN TIME: هنگامی پیش می آید که الگوریتم (فلو چارت)از نظر ساختار سالماست ولی مسئله خواسته شده را حل نمیکند.چک کردن مقادیر آخرحلقه (جاییکه داریم از حلقه خارج میشویم)برای چک کردن برای چک کردن شرط خروج است(معمولاً) چک کردن مقایر ابتدای حلقه برای آزمودن آغاز درست ومنطقی است .

فلو چارتی بکشید که سه عدد از ورودی گرفته و آنها را به ترتیب کوچک و بزرگ چاپ کند. سپس آن را Trace کنید .

اعداد انتخابی 7,9,5 a < = b < = c

a	b	c	temp
7	9	5	-
5	-	7	7
-	7	9	7
5	7	9	


Start
A,b,c
a>b
Y
Temp← a
A← b
B← temp
N
A<c
Y
Temp← a
A← b
B← temp
N
b>c
Y
Temp← a
A← b
B← temp
N
End
C
B
A

فرض کنیم ما n تا عدد داریم برای مرتب کردن (sortکردن) آنها ابتدا آنها را bobble sort می کنیم.

کوچک

۱۹	۷	۷	۷	۷	۲
۷	۱۹	۱٤	۱٤	۲	۷
۱٤	۱٤	۱۹	۲	۱٤	۱٤
۲	۲	۲	۱۹	۱۹	۱۹

بزرگ

آرایه: عبارت است از تعدادی متغیرهای از یک جنس و یک نوع که در کنار هم به یک نام خوانده می شوند و با اندیس دهی به آن میتوان به هر یک از متغیرها دسترسی جداگانه داشت.

A[1]	A[2]	A[3]	A[4]
19	14	7	19

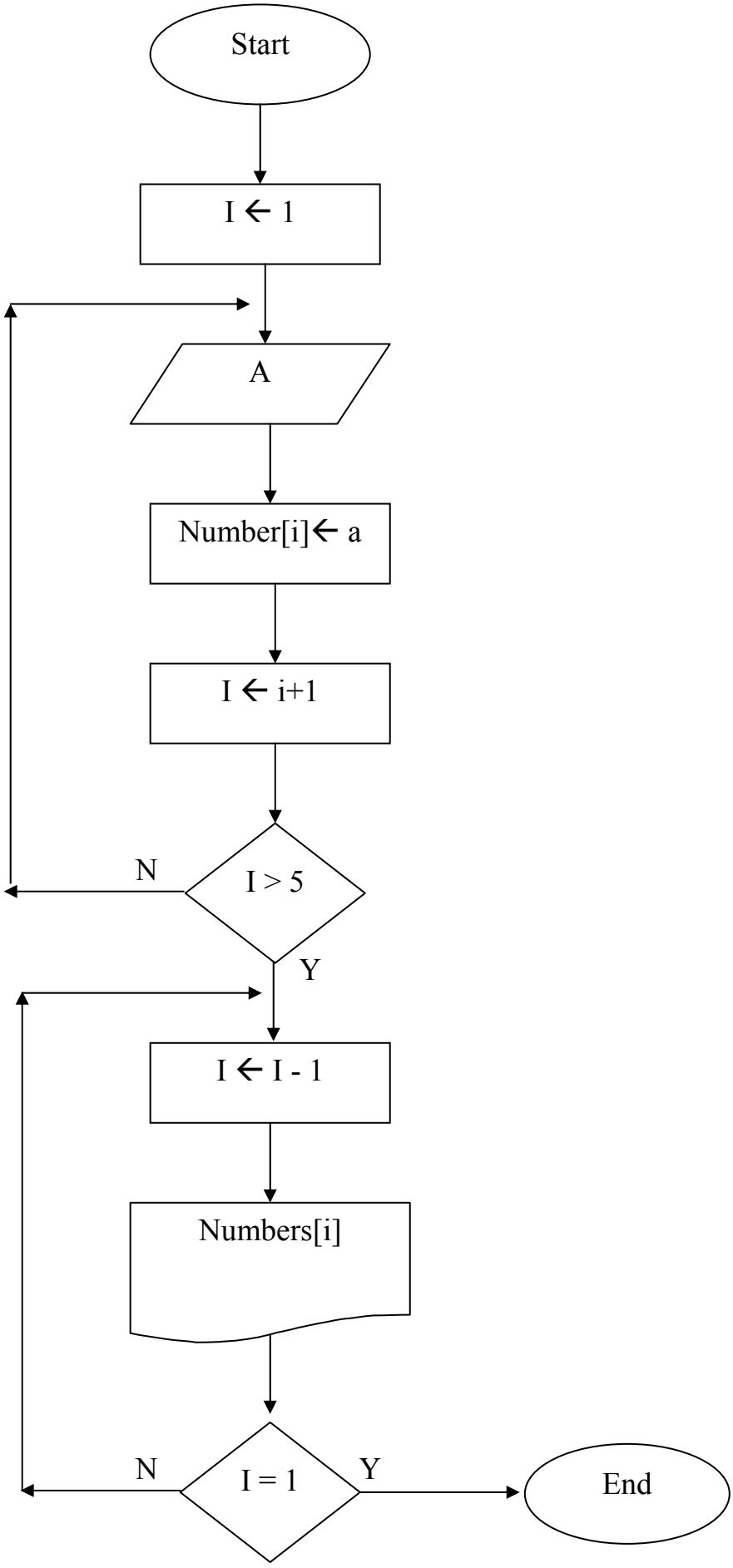
فلوچارتی بکشید که ٥ عدد را از ورودی گرفته و آنها را به ترتیب عکس آنچه از ورودی گرفته چاپ کند و آنرا trace کنید.

اعداد انتخابی1,7,3,9,17

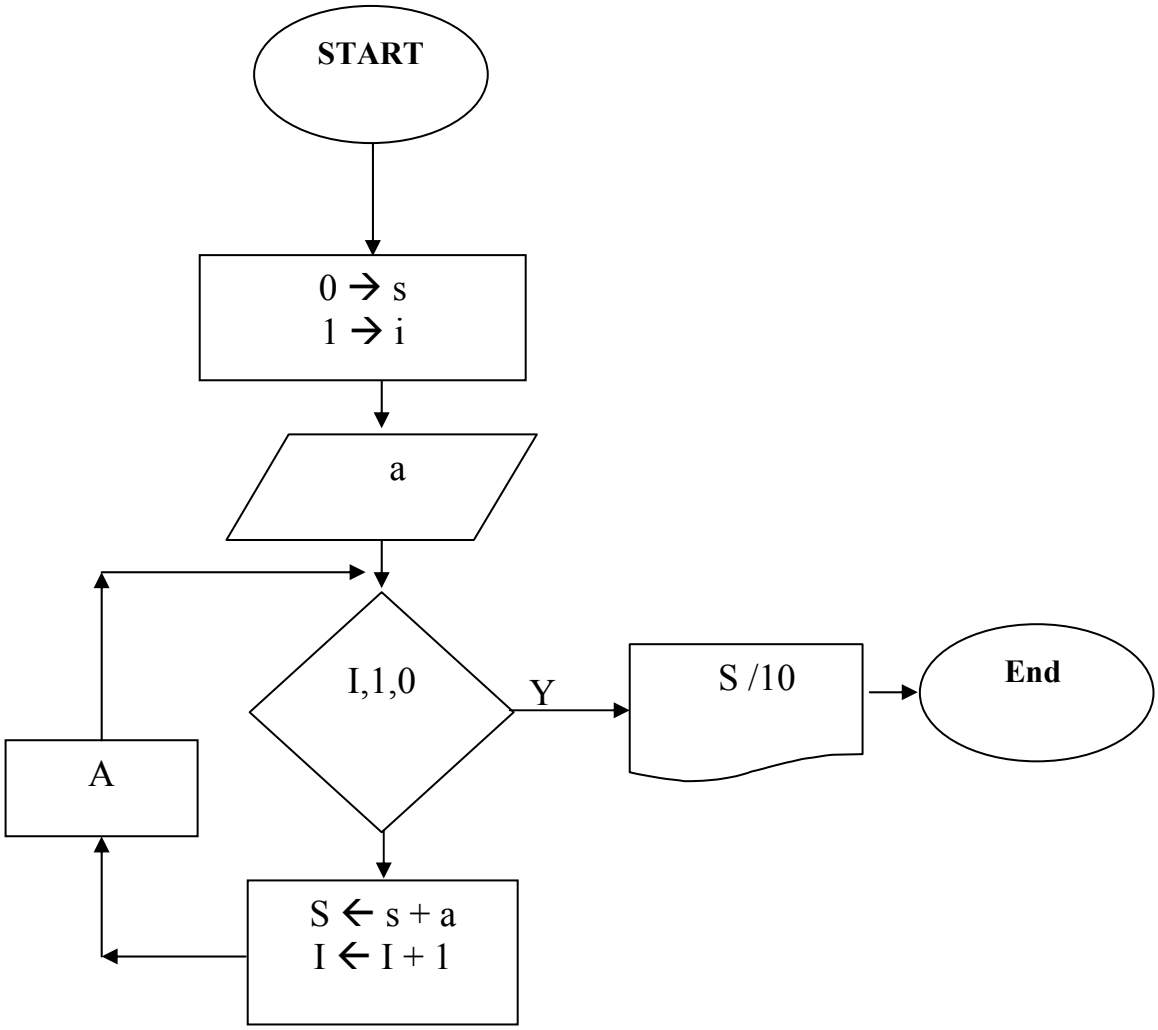
5	**4**	**3**	**2**	**1**
17	9	3	7	1

I	A
1	1
2	7
3	3
4	9
5	17
6	
5	17
4	9

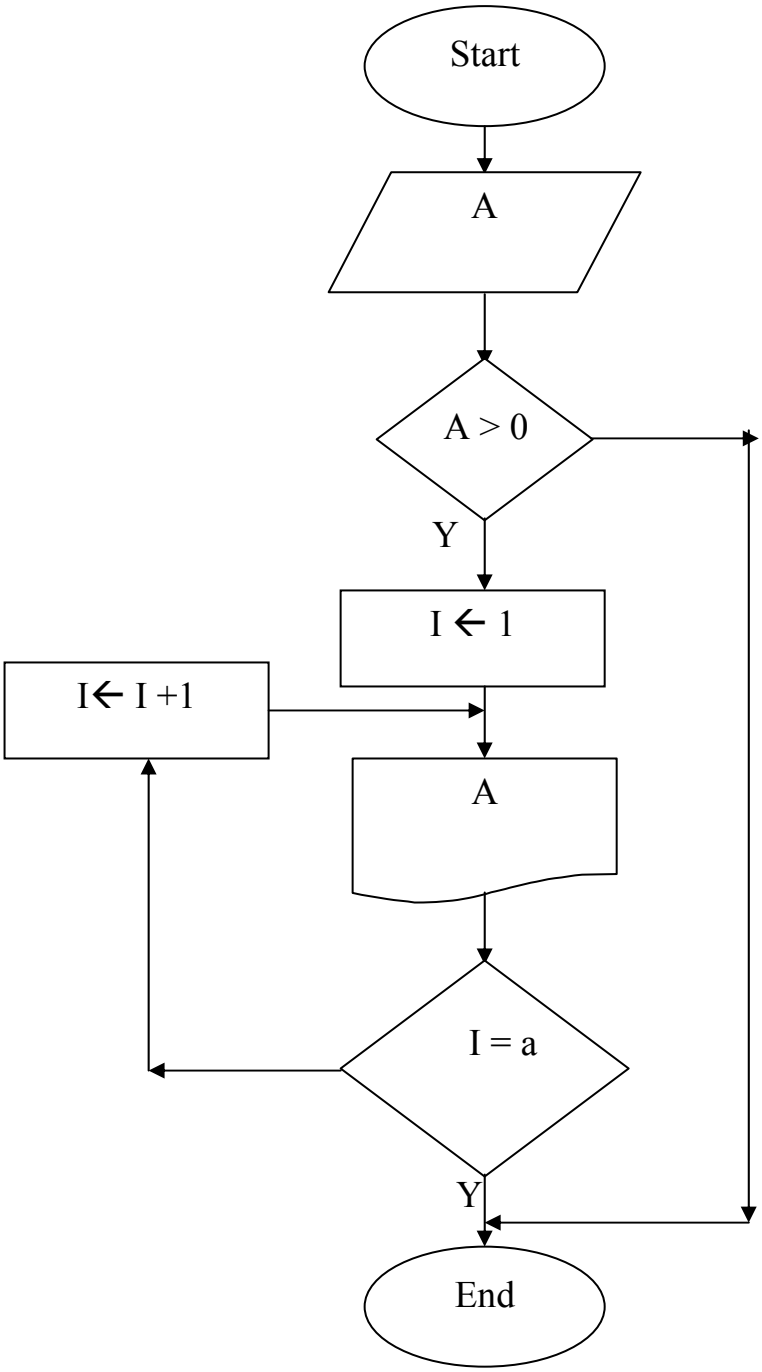
3	3
2	7
1	1



فلوچارتی بکشید که ۱۰ عدد را گرفته و میانگین آنها را چاپ کند.

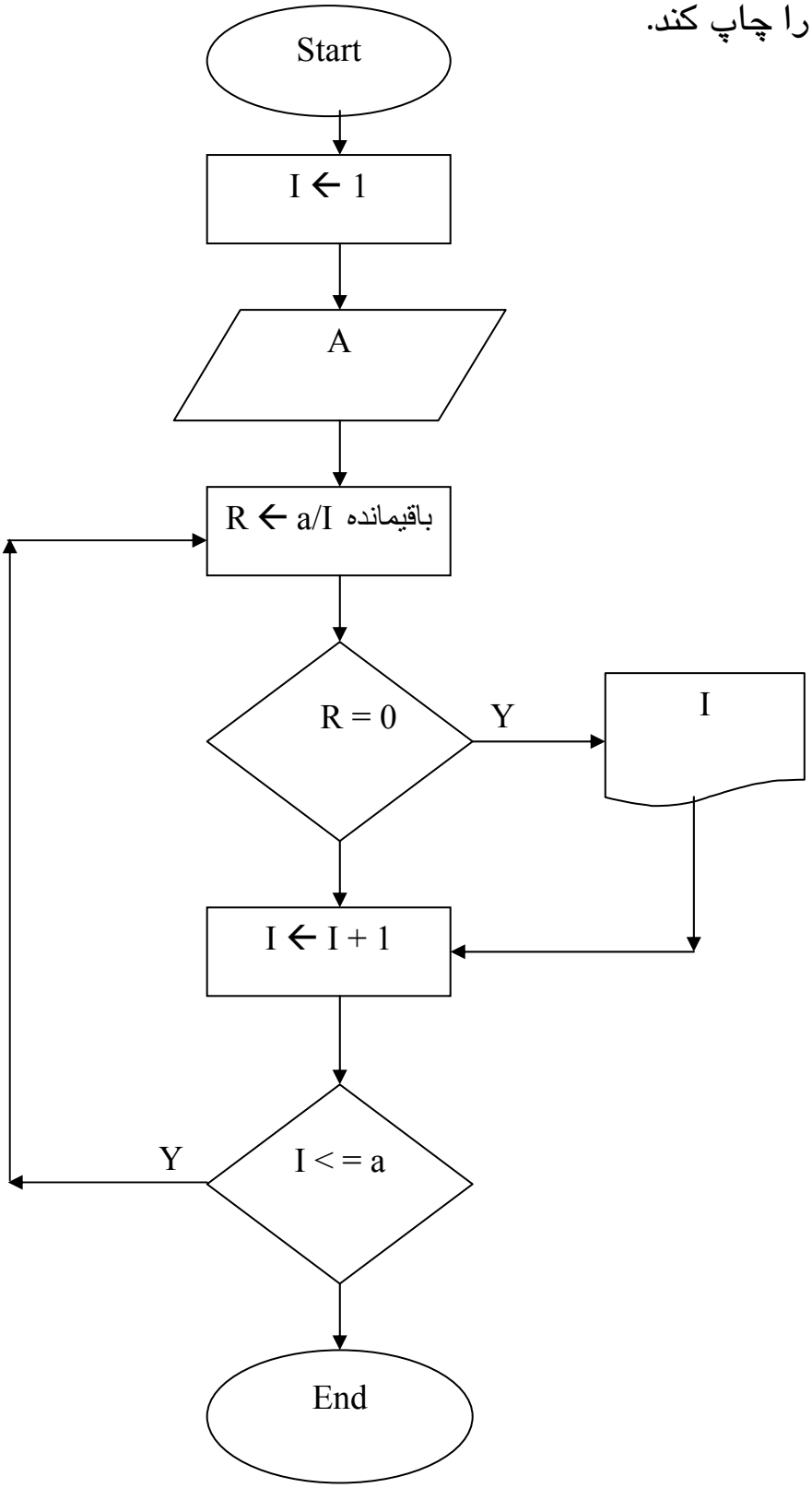


فلو چارتی می خواهیم که یک عدد درست را ازورودی خوانده و آنـرا بـه تعداد خودش چاپ کند.

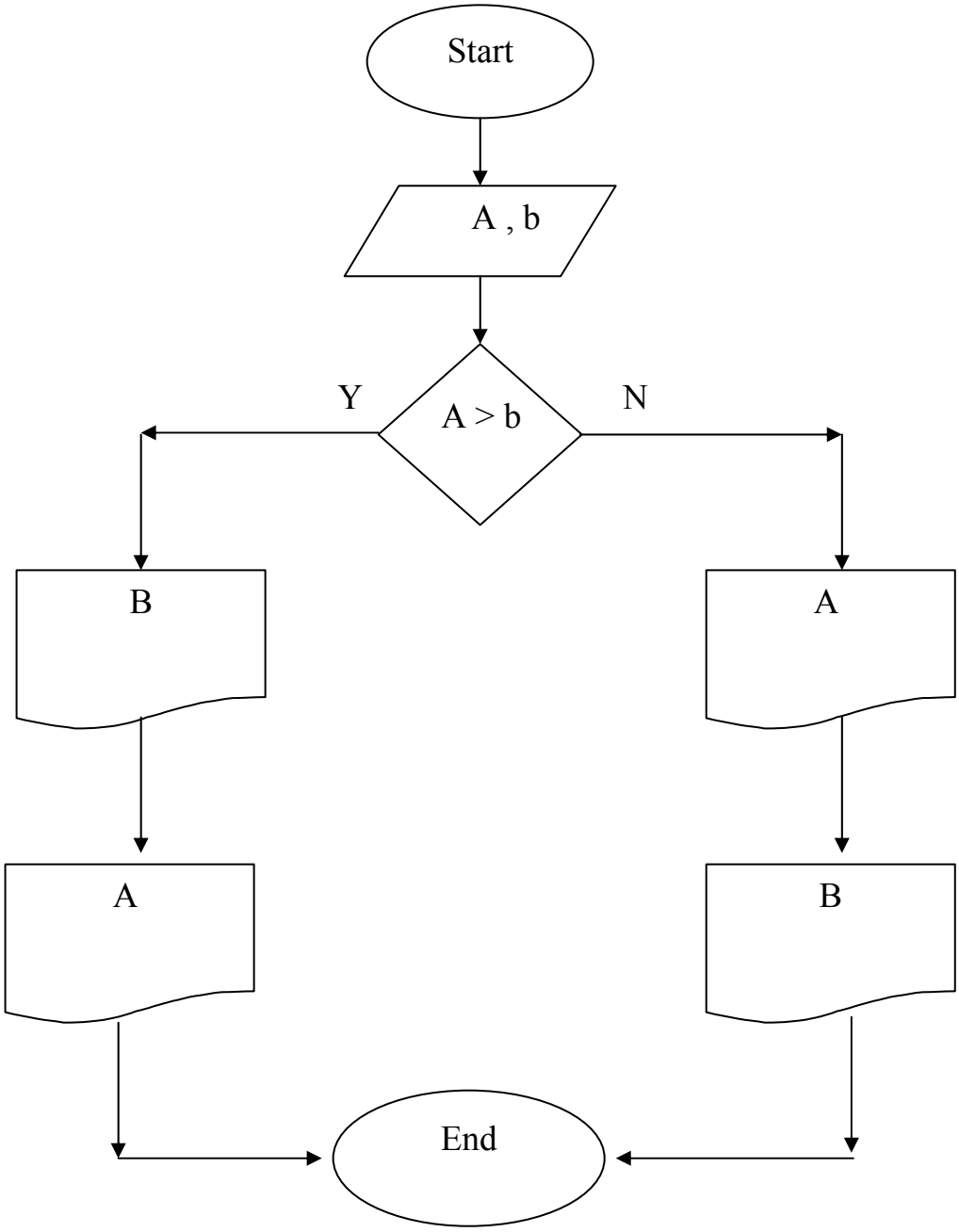


A	I
4	1
	2
	3
	4

فلوچارتی رسم کنید که یک عدد طبیعی را گرفته و مقسوم علیه های آن را چاپ کند.



فلوچارتی بکشید که دو عدد را گرفته و به ترتیب کوچک بـه بـزرگ چـاپ کند.

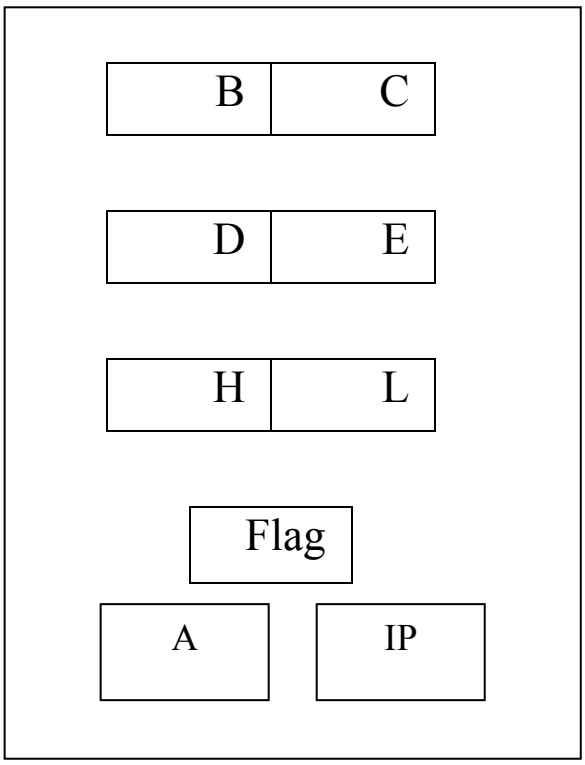


زبان اسمبلی:

زبانی است که هر دستور آن به یک دستور زبان ماشین ترجمه میشود که cpu آنرا میبیند و اجرا میکند. هر دستور زبان اسمبلی(هر خط از برنامه اسمبلی) دارای بخش های زیر است:

Operands(پارامتر-عملوند)	Menemonic(علامت اختصاری)	Label(برچسب آدرس)
داده هایی که قرار است عمل بر روی آنها انجام گردد.	علامتی است معمولاً مخفف کلمه ای است که دستور را بیان می کند.	آدرس حافظه که دستور در آن قرار دارد.

ساختار cpu :



رجیستر:محلی است برای ذخیره داده در cpu و انجام اعمال ریاضی – منطقی بر روی آن.

IP(Instraction Pointer): رجیستری است که آدرس دستور بعدی که باید اجرا شود در آن قرار دارد.

A(Accumubtor) : معمولاً در اعمال ریاضی – منطقی یک طرف عمل قرار دارد و معمولاً پاسخ در آن ذخیره میشود.

Flag: رجیستری است که هر بیت آن معنی ویژه ایی دارد و اعمال ریاضی و منطقی در دیگر رجیسترها بر روی این رجیستر اثر گذاشته و مقدار بیت های آن را تغییر می دهد.

....	c	z	s

Signbit(s) بیت علامت : هرگاه نتیجه عمل ریاضی قبلی منفی باشد این بیت ۱ میشود و اگر + یا صفر باشد این بیت صفر میشود.

Zero(z) بیت صفر: هرگاه نتیجه عمل ریاضی- منطقی قبلی صفر باشد بیت ۱ میشود و اگر غیرصفر باشد صفر مشود.

Carry(c) ۲ بر ۱ : هرگاه عمل ریاضی منجر به تولید ۲ بر ۱ از طبقه آخر شود این بیت ۱ میشود.

دستور Add : addition(جمع کردن) فرم کلی بصورت زیر است:

Label add a,b معادل→ a← b+a

دستور ld(load) : برای انتقال یک مقدار در یک رجیستر یا خانـه حافظـه بکار میرود.

Ld a,b a←b
Ld h,a h←a

نکته: همیشه حرکت از راست به چپ است.

دستور sub:

Sub a,b a← a-b

نکته : (bc) خانه ایی از حافظه است که آدرس آن رجیستر b و c است.

پرش مطلق: پرش jump

jp 1050
Ip←1050

یعنی دستور بعدی که اجـرا خواهـد شـد در آدرس ۱۰۵۰ اسـت (بـرو بـه آدرس ۱۰۵۰ و هر دستوری آنجا بود اجرا کن)

پرش شرطی:

Jc(jump carry) اگر flag ۲ بر یک ، مقدار یک دارد پرش کن

Jc(آدرس)

Jnc(jump no carry) : اگر carry flag=0 پرش کن

Jnc(آدرس)

Jz(jump zero) : اگر پرچم z برابر با یک است پرش به آدرس موردنظر انجام شود.

Jz(آدرس)

دستور nop(no operation) : هیچ کاری انجام نده.

دستور input : یعنی چیزی را از ورود بگیر و در پارامتر ذخیره کن.

Input(پارامتر)

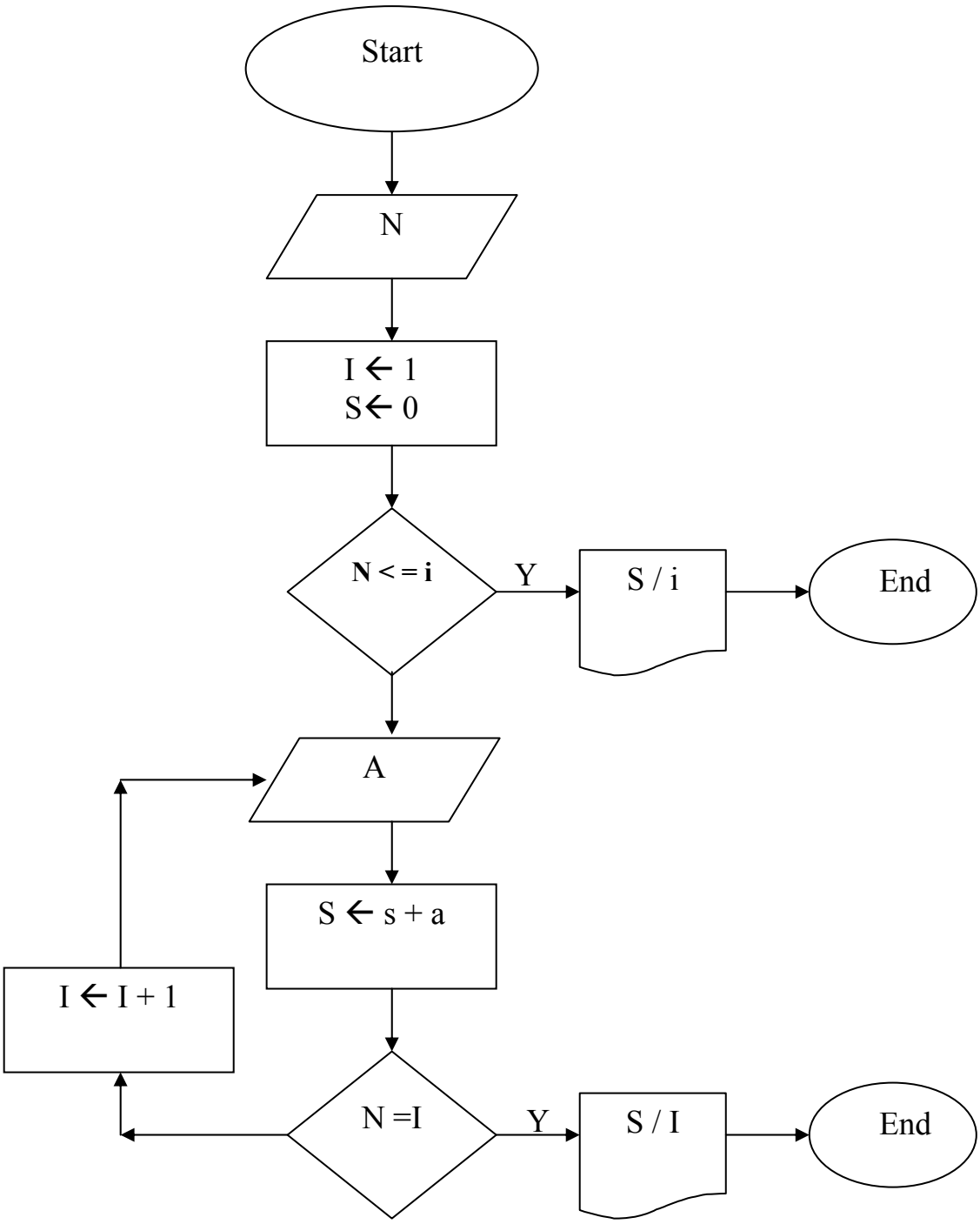
دستور output : پارامتر را در خروجی بنویس.

output(پارامتر)

دستور compare(cmp و cp) مقایسه کن a-b می شود و flag ها تاثیر می گیرند و نتیجه جایی ذخیره نمی شود.

Cmp a,b a← b-a

فلوچارتی بکشید که میانگین سه عدد را حساب کند سپس برنامه اسـمبلی این فلوچارت را بنویسید.

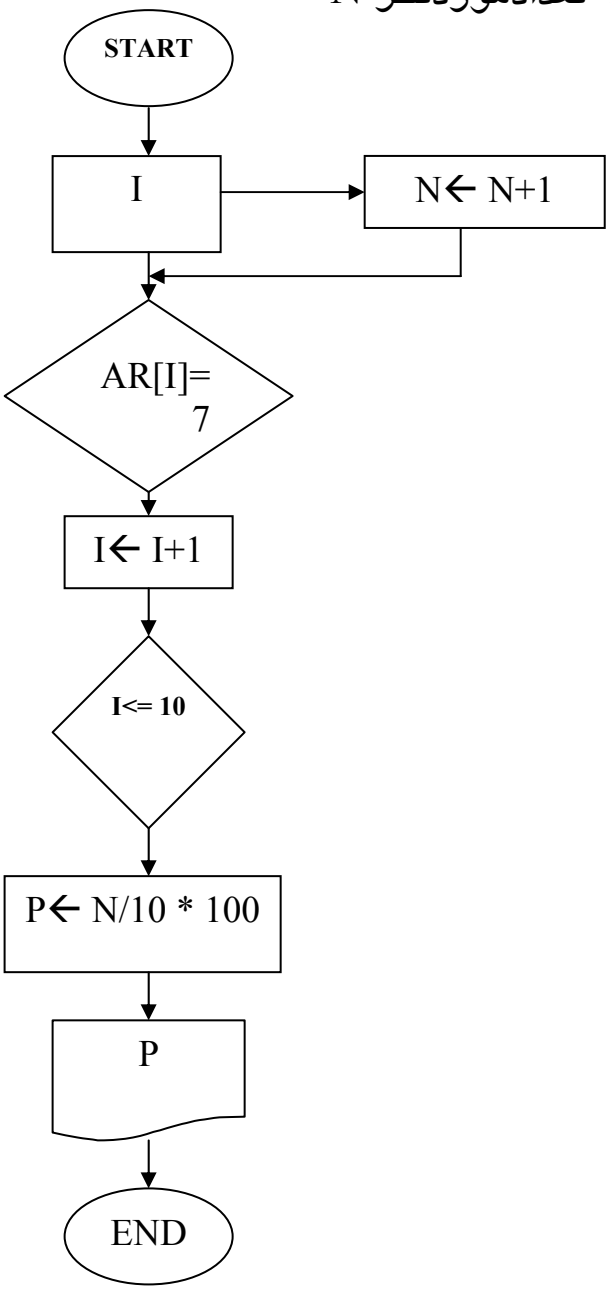


فلوچارتی بکشید که ۱۰ عدد از ورودی گرفته و درصد فراوانـی یـک عـدد را در میان تعدادی عدد بیاید.

ar

۱	۲	۳	٤	٥	٦	۷	۸	۹	۱۰
7	9	10	1	4	3	5	4	7	7

درصد عدد مورد نظر P تعدادموردنظر N



پرش های شرطی

```
Jz    jump if zero
Jnz   jump if not zero
Js    jump if negative
Jns   jump if not negative
Jc    jump if carry
Jnc   jump if not carry
```

دستور تقسیم(Divide) : div

دستور ضرب(multiply) : mul

برنامه نویسی اسمبلی:

به عنوان مثال برنامه فلوچارت قبل را می نویسیم.

۱۰ عدد در جایی از حافظه قرار دارند اولن جایی که در آنجا قرار دارند را ar میگیریم.

```
Start:
Ld    a,1
Ld    I,a
Ld    a,0
Ld    n,a
Ld    hl,ar
Ld    bc,i
Add   hl,bc
Sub   hl,1
Ld    a,hl
Cmp   a,7
Jnz   label1
Ld    a,n
Add   a,1
```

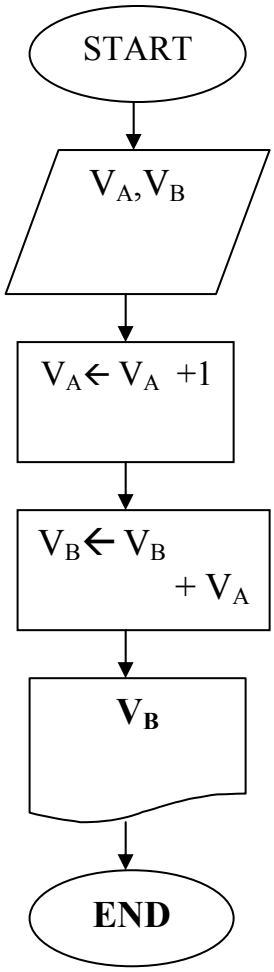
```
Ld     n,a
Label: ld   a,i
Add  a,1
Ld     I,a
Cmp a,10
Js     label2
Jz     label2
Ld     a,n
Ld     b,10
Div   a,b
Mul  a,100
Ld     p,a
Output      a
```

Hl ← h و l کنارهم ، چون یک رجیستر برای جادادن آدرس کوچک است دو تا رجیستر را در نظر میگیریم.

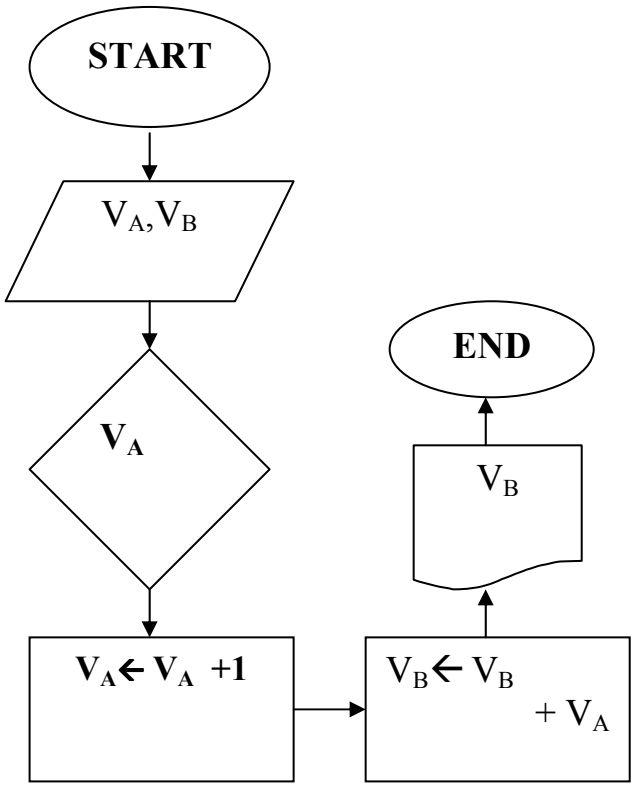
H	L

```
Ld    hl,ar        hl←1000
Ld    bc,I         bc←1
Add  hl,bc         1+1000=1001
Sub  hl,1          1001-1=1000
Add  a,b           a←a+b
```

مثال:




```
Input  VA
Input  VB
LD  A,VA
LD  B,VB
ADD  A,1
ADD  B,A
LD  VB,B
OUTPUT  B
```



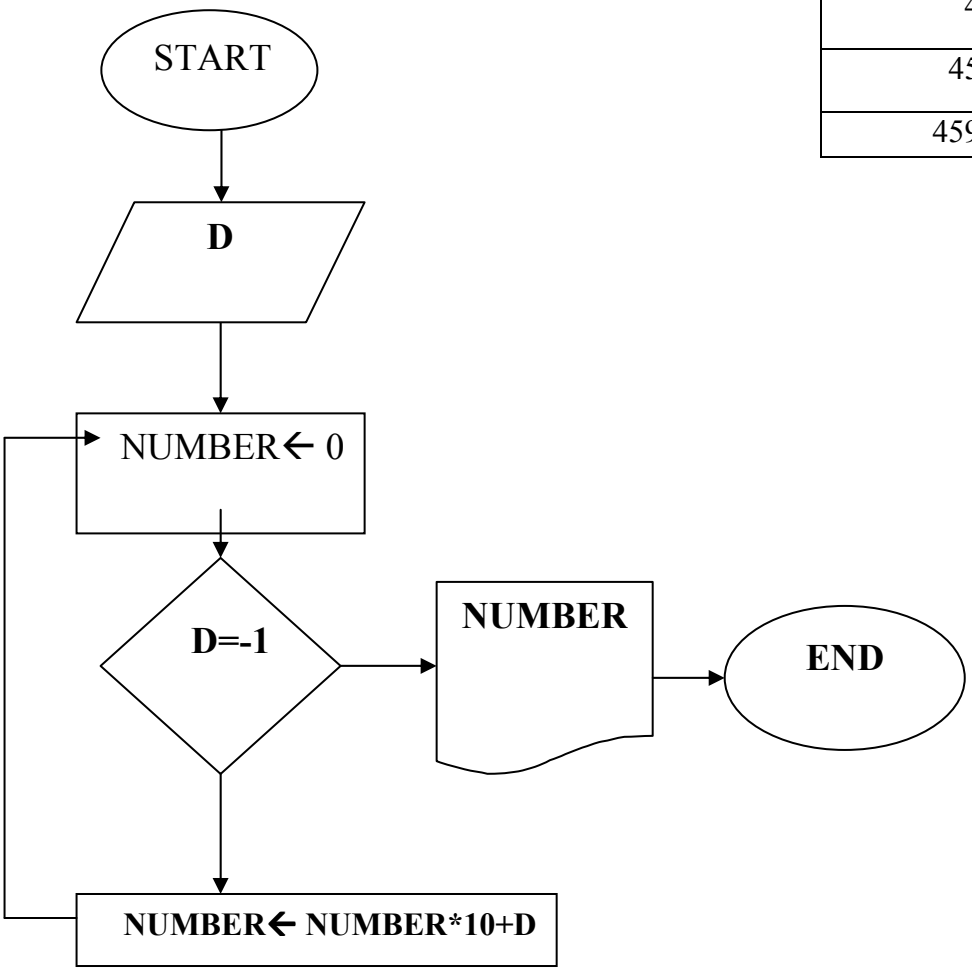

```
Input  VA
Input  VB
LD  H,VA
CMP  H,O
JZ  LABEL
JS  LABEL
LD  A,VA
LD  B,VB
ADD  A,1
ADD  B,A
LD  VB,B
OUTPUT  B
LABLEL  STOP
```

فلوچارتی بکشید که ارقام یک عدد ۵ رقمی را گرفته و خود عـدد را چـاپ کند. سپس برنامه این فلوچارت را هم بنویسید.

نمونه آخرین عدد که وارد شد پس از آن ۱- وارد می کنیم. عدد بـا ارزش بالاتر وارد میشود.

نمونه: 4,5,9,-1

Number	D	خروجی
0	4	
4	5	
45	9	
459	-1	459



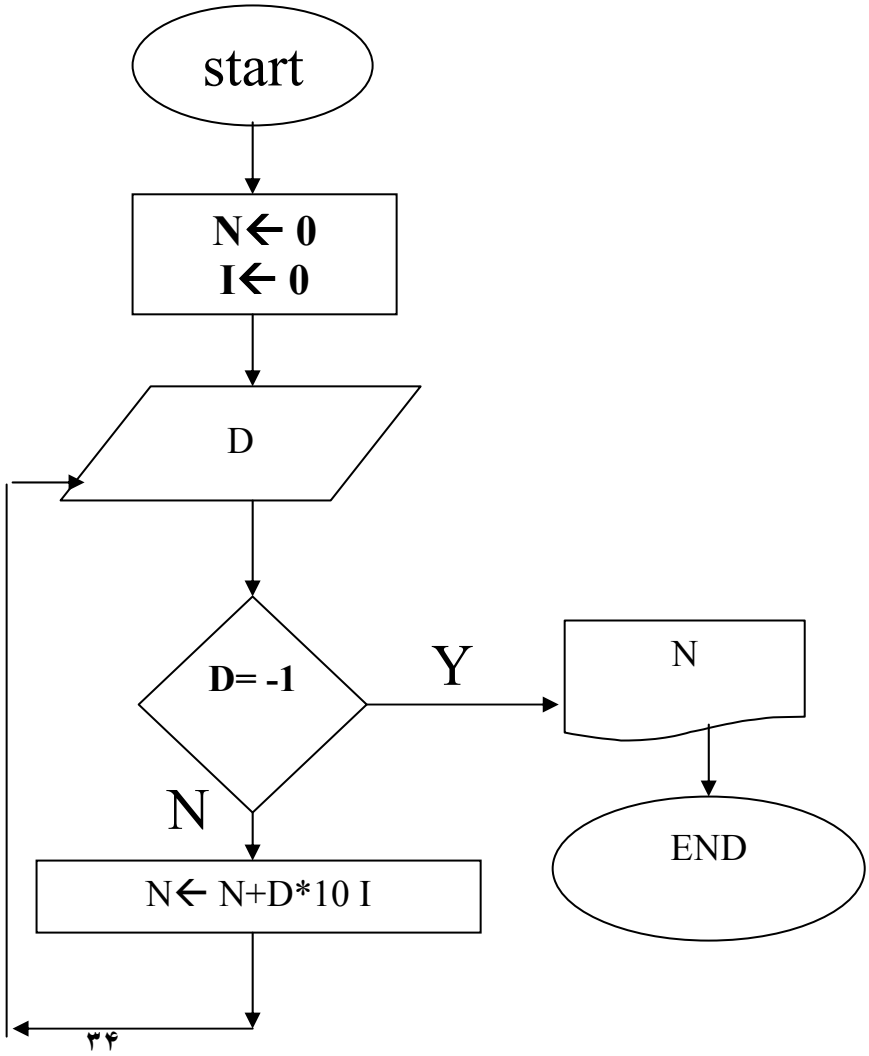

```
Start
Ld    a,0
Ld    number,a
Indigit      inputdigit
Ld    a,negative1
```

```
Ld    b,digite
Cmp a,b
Jz    printing
Ld    a,number
Mul  a,10
Add  a,digit
Ld    number,a
Jp    indigit
Printing:output number
```

فلوچارت و برنامه ای بنویسید که رقم های یک عدد N رقمی را از سمت راست عدد به ترتیب گرفته و عدد را چاپ کند.

هرگاه رقم ها به پایان رسیدند پس از آن ۱- وارد میشود.

فلوچارت روبرو را بدون استفاده از توان بنویسید و برنامه آنرا نیز بنویسید.



دستورهای منطقی در زبان اسمبلی:

دستور and : andکردن دو رجیستر با هم عبارت است از andکردن بیت های نظیر در دو رجیستر.

And r_1,r_2 $r_1 \leftarrow r_1 \wedge r_2$

a	b	A^b
0	0	0
0	1	0
1	0	1
1	1	1

1011001101
1111111011

1011001001

دستور OR : or کردن دو مقدار n بیتی با هم عبارت است از or کردن بیت های نظیر.

Or r_1,r_2 $r_1 \leftarrow r_1 \vee r_2$

a	b	Avb
0	0	0
0	1	1
1	0	1
1	1	1

10001101
01100100

11101101

دستور not : not کردن n بیت عبارت است از not کردن هریک از بیت ها.

نمونه: 1011110→0100001

دستور Exclusive OR

a	b	Aob
0	0	0
0	1	1
1	0	1
1	1	0

چند خاصیت:

۱- and شدن هر چیز با یک خود آن چیز میشود.

۲- And شدن هر چیز با مقدار تهی میشود خودش.

ماتریس ها: همانگونه که بردارها یا جدول ها تک بعدی را با ارایه ها نشان دادیم میتوانیم جدولهای دو بعدی یا ماتریس را با آرایه دو بعدی نمایش دهیم.

آدرسmatrix: matrix[i][j]= matrix+ (i-1)*3+(j-1)

دستورهای shift:

Shift: عبارت است از لغزاندن بیتهای یک ارزش چند بیتی به یک سو (چپ یا راست) به گونه ایی که هر بیت جای بیت سمت چپ یا راست خود را بگیرد و از سمت راست یا چپ ترین موقعیت یک بیت جدید وارد شود.

Shift left

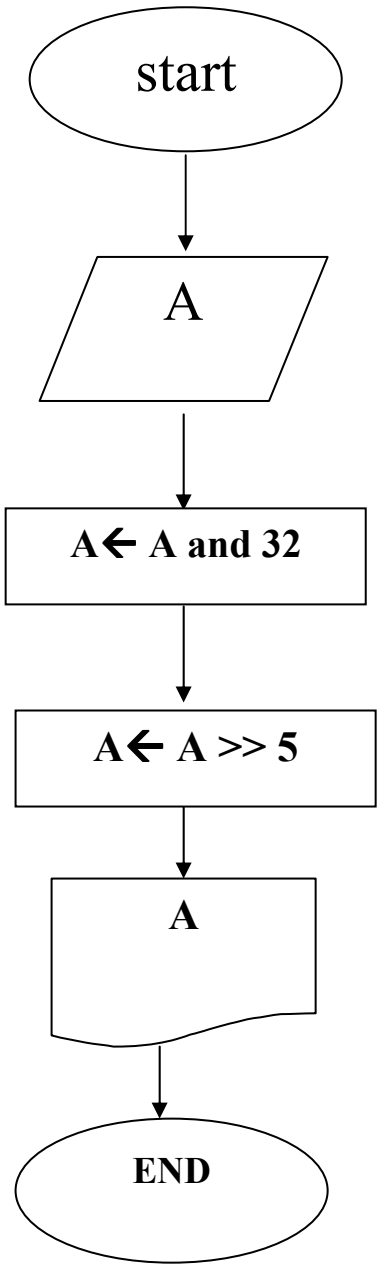
011101 →0→ (0)111010

عمل Rotate : برای چرخاندن یک ارزش چند بیتی است و همان شیفت است که در آن بیتی که از یک طرف وارد می شود همان بیتی است که از سمت دیگر خارج می شود.

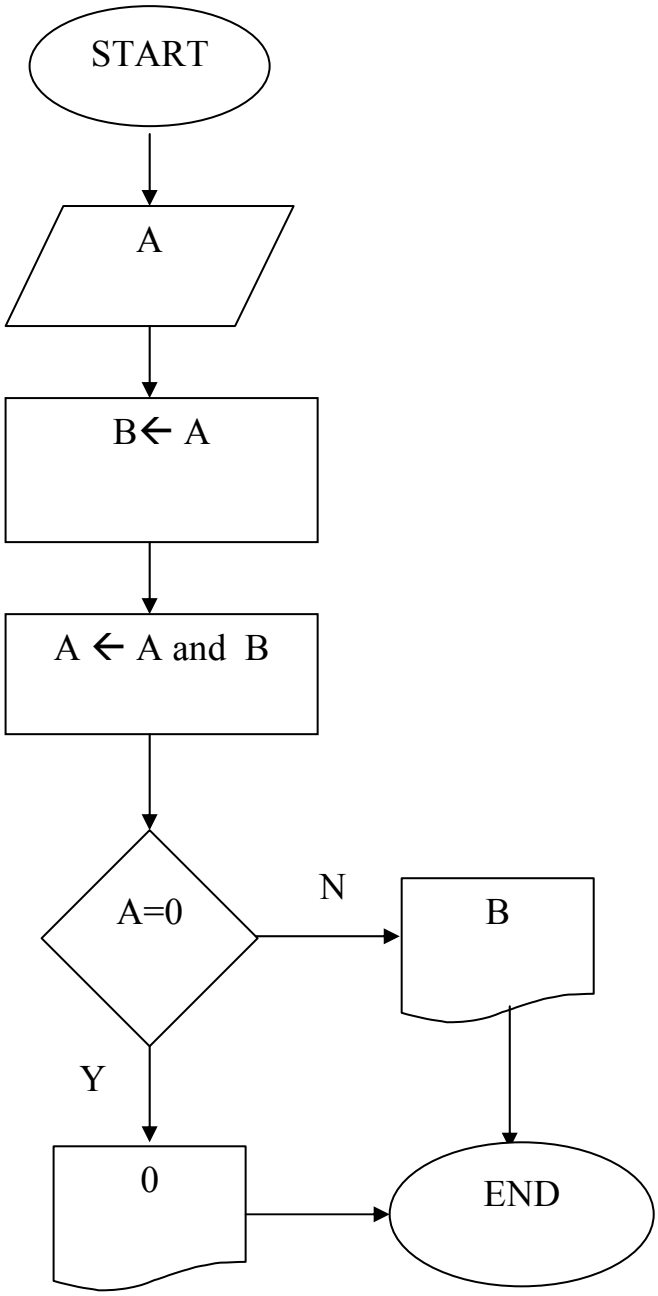
1101→ 1011→ 0111 → 1110

فلوچارت و برنامه ای بنویسید که یک عدد از ورودی گرفته وبیت ۵ آن را چاپ کند.

```
START: input a
       And a,32
Shift  Shr a
       Shr a
       Shr a
       Shr a
       Shr a
       Output a
```



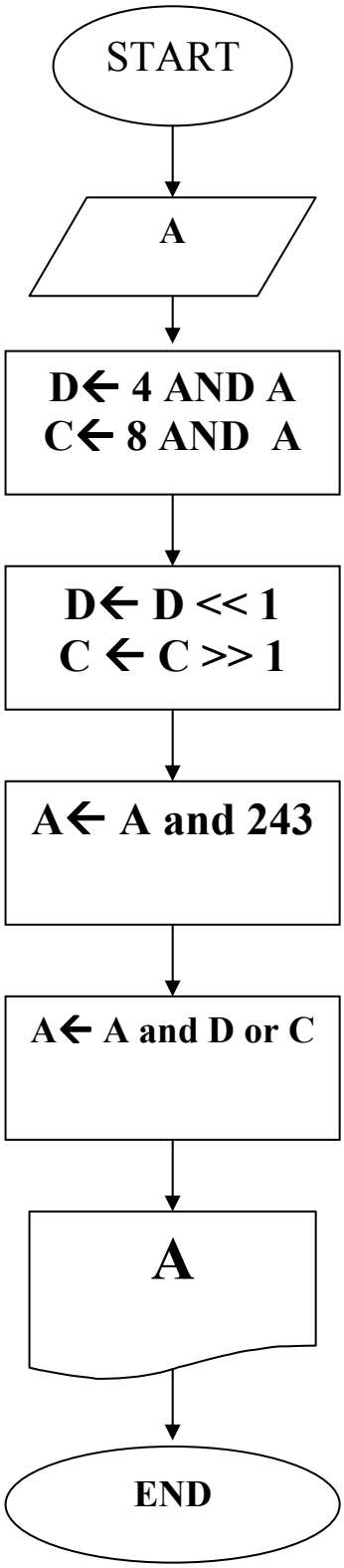
فلوچارت و برنامه ای بنویسید که اگر بیت سوم عدد ورودی ۱ بود خود عدد را چاپ کند و اگر صفر بود ، صفر را چاپ کند.




```
START
Input A
Ld b,a
And a,b
Cmp a,0
Jz label1
Output b
Jmp label2
Label1:output d
Label:end
```

برنامه و فلوچارتی بنویسید که جای بیت دوم و سوم یک عدد را عوض کند. سپس آن را (تغییر یافته عدد) چاپ کند.

```
Start input a
Ld d,4
And d,a
Ld c,8
And c,a
Shfl d
Shfr c
And a,243
Or a,c
Or a,d
Output a
```



255-4-8=243

زبان های سطح بالا : c و پاسکال
Basic,cobol,…

زبان سطح بالا:
زبانی است منطقی نزدیک به زبان انسان که برای بیان الگوریتم بکار میرود و توسط نرم افزارهایی به نام compiler درک میشود و به زبان سطح پایین ترجمه میشود.
C و خانواده آن مانند #c و C++ و ++j و java و Ansic
تقریباً ۹۰ درصد برنامه نویسی سیستم در دنیا را به خود وابسته کرده اند ، تقریباً دارای syntax همسان هستند و یادگیری یکی از آن ها م تواند یادگیری دیگری را آسان کند.

دو package بسیار مهم c عبارتند از :
Microsoft c یا همان visual c یا c.net
C#:Borland و c++ builder

Package های قدیمی تر c : ورژنهای 2.1 به بعد turbo c (تحت dos)

پاسکال:
دارای ویژگی های ساخت یافتگی قوی است.(structured)

دارای package های قدیمی (تحت dos) با نام های turbo pascal ورژنهای ٦ و ٧

دارای یک ورژن visual بنام Delphi است که توسط شـرکت Borland ارائه شده است.

نوشتن یک برنامه نمونه به زبان c و pascal

Pascal:

```
Program hello world;
Begin
Writeln('Hello world');
End.
```

C:

```
Main()
{
Printf("hello World\n");
Return(0);
}
```

```
Program  third:
Var  D,I,j : inteyer ;
Begin
Readln( I ) ;
Readln ( J ) ;
P : = I * J ;
Writedn ( J,´×´, J , ´=´, P ) ;
End .
```

مثال : برنامه ای بنویسید که عدد n را از ورودی گرفته و !n را چاپ کنـد (فرض کنید ۳۲۰۰۰ > !n باشد)

```
Program  fact ;
Var       n , I , fact , integer ;
```

```
Begin
Read in ( n ) ;
I : = 1
Fact : = 1 ;
While  i<= n      do
Begin
Fact : = fact * i,
i : = i + 1;
End ;
Writedn ( ´faction  of ´, n ,´ is ´ , fact ) ;
Read in ;
End .
```

```
main ( )
{
int   I , fact , n ;
scan f ( " % d " , f n ) ;
for ( fact = 1 , % = n ; I > 1 ;
fact = fact * I ;
return ( 0 ) ;
}
```

Reserved world : کلماتی هستند که توسط کامپایلر بـرای بیـان معنـی های ویژه ایی در نظر گرفته شده اند و معنی خاصی از انها برداشـت مـی شود.

در متن ما این کلمات عبارتند از :

Program (شروع یک برنامه)

Begin (شروع **program body** بدنه برنامه)

End(پایان)

نکته: در پاسکال بزرگی و کوچکی حروف یک کلمه مهم نیست و فرقی نمی کند اما در زبان c هر کلمه باید طبقcase خود نوشته شود.

Notation یا pointing ها : برای بیان عملیات یا پایان و آغاز جمله ها (;)

Identifier : نام هایی هستند که برنامه نویس برای نامیدن اجزاء برنامه بکار می برد و دلخواه هستند (حق استفاده از اسم های تکراری نداریم.)

روترها(procedures & Functions) : قطعه برنامه هایی از پیش نوشته شده اند که برای انجام کار خاصی نوشته شده اند و به کار برده می شوند.

مانند writeln : برنامه را در صفحه خروجی بنویس ln(line) خط
Write('hello world') : فقط روی یک خط می نویسید ، پشت سرهم می نویسید.
writeln('hello world'): ln باعث میشود که برود خط بعد بنویسد.

برنامه c
Main : **آغاز برنامه (روند اصلی)**
Printf : **یک روند است برای چاپ کردن**
new line : **\n**

نوشتن برنامه قبلی از زبان c به زبان پاسکال:

نکته هایی مربوط به مثال بعد:

۱- := همان فلش مقدار دهی است.

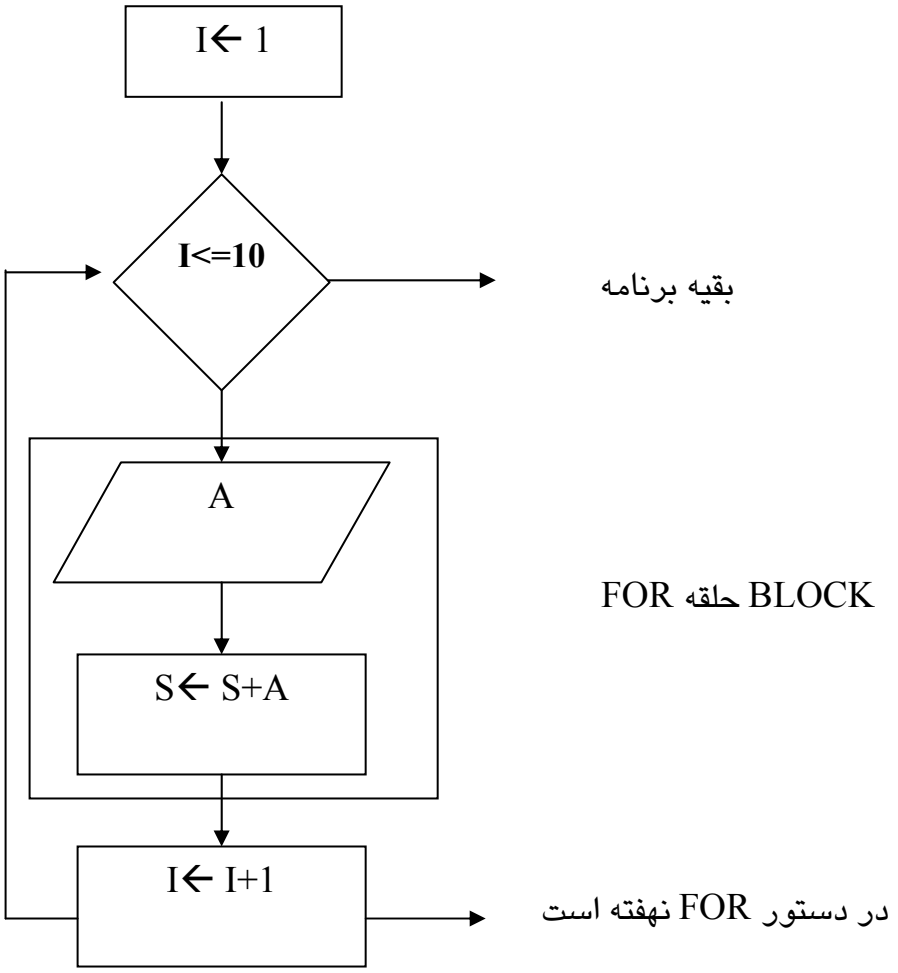
۲- همیشه در برنامه یک begin و end نقطه دار باید باشد که نشاندهنده شروع و پایان برنامه باشد.

۳- هر وقت بخواهیم دو دستور را با هم اجرا کنیم باید دو دستور را در یک بلاک گذاشته و begin و end (بدون نقطه) در شروع و پایان block می گذاریم.

برنامه ای بنویسید که ۱۰ عدد از ورودی گرفته و جمع آنها را چاپ کند.(PASCAL)

```
Program   add10;
Var
      I:integer;
      A: integer;
      S:integer;
Begin
      Sum:=0;
      For i:=1 to 10 do
      Begin
      Readln(a);
      S:=s+a;
      End;
      Writeln(s);
End.
```

For



Program header: به بخش ابتدایی برنامه گفته می شود که در بردارنده معرفی انواع اجزایی است که در برنامه بکار خواهد رفت

متغیرها

ثابت ها

روندها

فایلهای مورد استفاده در برنامه که در بردارنده کد یا داده هستند.

Program body : در بردارنده الگوریتم اصلی و دستورات برنامه است که به ترتیب از چپ به راست و بالا به پایین انجام میشوند.

Code block : تعدادی دستورات که توسط علامت block با هم تشکیل یک گروه می دهند که همگی یا باید اجرا شوند یا اجرا میشوند.

Type های اصلی داده

	c	Pascal
اعداد صحیح	Int	Integer
عدد ممیز دار	Float	Real
کاراکتر	char	char

حلقه for :

هرگاه به تعداد مشخص بخواهیم عملی را تکرار کنیم از این ساختار بهره می بریم(بیشتر برای داده های متوالی کاربرد دارد) حالت کلی آن بصورت زیر است:

For (intialization)مقدار اولیه to (final-value)مقدار نهایی
do
For statement(یا یک دستور تنهاست یا بلاکی از دستورهاست)

اجرا

در هنگام ورود به for ، عمل initiazation انجام میشود.

سپس ۱- (بررسی شرط for) مقدار شمارنده با مقدار نهایی مقایسه می شود.

اگر شرط درست بود (شمارنده=< مقدار نهایی) ۲- for-statement اجرا میشود.

۳- افزایش شمارنده و تکرار از مرحله ۱

برنامه ای به زبان c بنویسید که همه عددهای صحیح از 0 تا 100 را چاپ کند؟

```
Main ( )
{
 Int   i ;
For ( i = o ; I < 101 ; i = I + 1 )
        Printf ( " %d" , i ) ;
         Return ( o ) ;
}
```

مثال : برنامه ای بنویسید که یک عدد را از ورودی خوانده و اگر زوج بود پیام عدد زوج است و اگر فرد بود پیام عدد فرد است را چاپ کند .

```
Program        add even ;
Var  I : Integer ;
       R : Integer ;
Begin
Write ( ´ Enter  an  Inteyer : ) ;
read in ( I ) ;
r = I    mod       z
if    r = o    then  writin  ( ´ number , is    even , I ) ;
                 else
writln ( I , ' is    odd' . ) ;
End .
```

برنامه ای بنویسید که جدول ضرب را چاپ کند.(PASCAL)

به صورت

۱*۱=۱

۱*۲=۲

...

۱۰*۱۰=۱۰۰

I	J	M
1	1	1
1	2	2
1	3	
...		...
1	10	10
	11	
2	1	2

```
PROGRAM Multiply;
Var
	I,j:integer;
	M:integer;
Begin
	For i:=1 to 10 do
		For j:= to 10 do
		Begin
		M:=I * j ;
		Writeln(I ,'*' , j ,'=', M);
		End;
End.
```

برنامه ای بنویسید که دو عدد صحیح را از ورودی گرفته و اولی را بتوان دومی رسانده و چاپ کند.

A^B $2^3 \rightarrow 2*2*2$

```
Program test;
Var
	S,A,B,i: integer;
Begin
	Writeln('enter 2 number');
	Readln(A,B);
	S:=1;
	FOR i:= 1 to B do
	begin
		S:=S*A;
		Writeln(s);
	End;
End.
```

نوشتن برنامه C جدول ضرب

```
Main()
{
Int I,j,m;

For (i=1; i<= 10 ; i=i+1)
For (j=1; j<= 10 ;
j=i+1)
{
m=i*j;
Printf("\d*\d=\d \n",I,j,m);
}
Return(0);
}
```

دستور for در c :

```
For (initiazation ;  condition  ; increment)
        For statement
```

روند اجرای for در c :

هنگام ورود به for ابتدا initiazation انجام میگیرد.

سپس

۱- شرط چک میشود و اگر درست بود مرحله ۲ و اگر نادرست مرحله ٤

۲- جمله for و قسمتincrement اجرا میشود.

۳- از مرحله 1 تکرار می شود.

٤- پایان

نکته:

در زبان c فلش بصورت = ولی در زبان pascal فلش بصورت :=
نشان داده می شود.

در زبان c علامت «مساوی نیست» بصورت =! و در زبان پاسکال
بصورت <> (نه کوچکتر نه بزرگتر) نشان داده میشود.

حالت کلی دستور printf : (همیشه تمام حروف در c بصورت حروف
کوچک هستند)

Printf(لیست متغیرها و فرمت)

فرمت : یک رشته از کاراکترهاست که چگونگی و محل چاپ شدن بقیه
متغیرها را به printf می گوید.

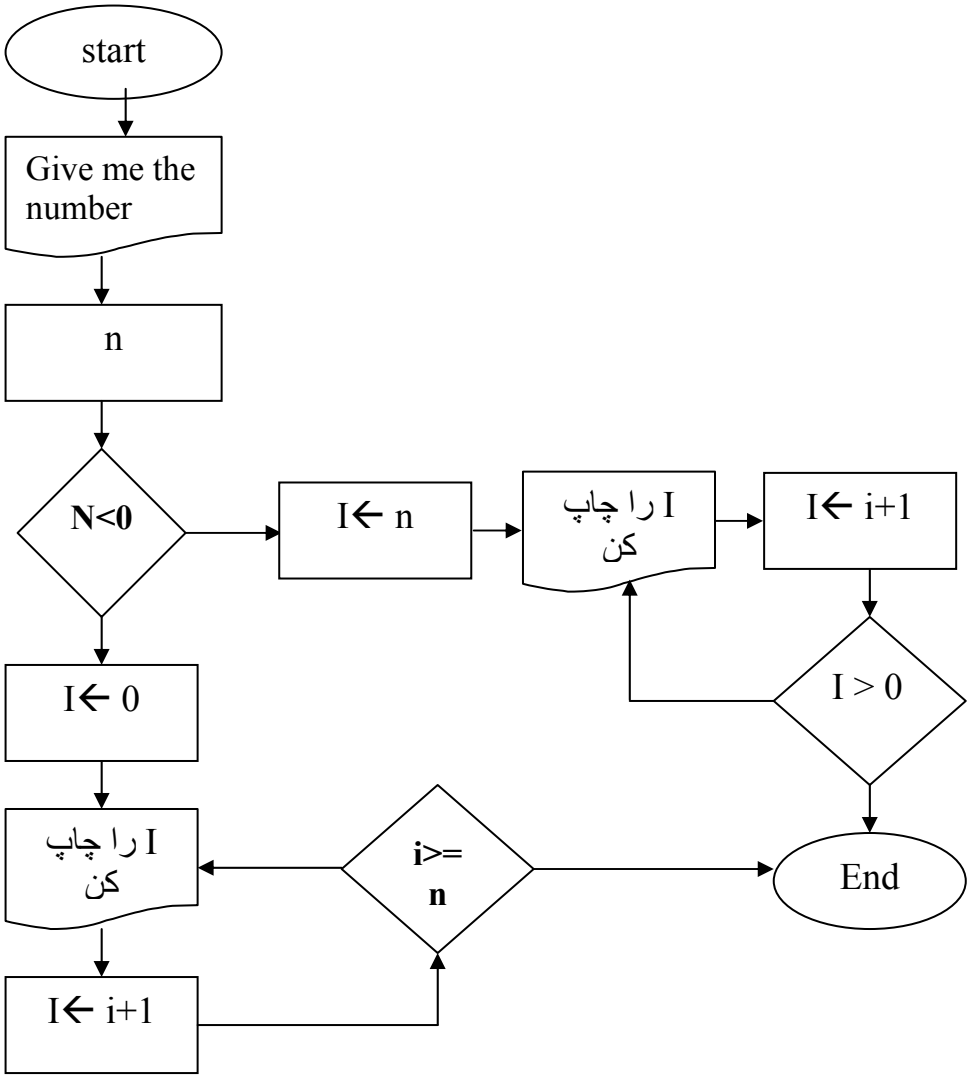
\n برو به خط جدید

خود بک اسلش (\) را بخواهیم بنویسیم می شود\\

برنامه ای بنویسید که عددی صحیح را از ورودی گرفته و همه اعداد از
صفر تا آن عدد را چاپ کند(زبان پاسکال)

```
Program zero_2_number;
Var
    I,n:integer;
Begin
    Write('Give me the number');
    Readln(n);
    If n<0 then
        For i:=n to 0 do
            Writeln(i);
```

```
If n>= 0 then
        For i:=0  to  n do
                Writeln(i);
End.
```



برنامه ای بنویسید که عددی صـحیح را از ورودی گرفتـه و بـه تعـداد آن ستاره چاپ کند

```
Program        s ;
Var      I , n : Integer ;
Begin
write ( ´ Enter  number  of  stars : " ) ;
Readln( n ) ;
For     I : = 1        To    n        Do
Write( ´ * ´ ) ;
Writeln ;
End .
```

دستورread : تمام خط را می خواند تا n/ که رسید متوقف می شود.

دستورreadln : حتی n/ را نیز می خواند و به خط بعد می رود.

از دستور else نیز می توانیم استفاده کنیم فقط آخرین ; (قبل از دستور else) را نمی گذاریم.

اگر چند شرط داشتیم از else if استفاده می کنیم.

دستور if

If شرط then جمله then [else جمله else]

هنگام رسیدن به یک ifابتدا شرط بررسی و اگر درست باشد جمله then اجرا می شود و اگر نادرست باشد اگر جمله else داشته باشیم اجرا می شود.

برنامه ای بنویسید که عدد صحیح n را از ورودی گرفته و آنرا برعکس کرده و چاپ کند. ورودی: ۷٦٤۱ خروجی : ۱٤٦۷

```
Program converse;
Var
	Ans,n:integer;
	D: integer;
Begin
	Ans:=0;
	Write('enter the number:');
	Readln(n);
	L1	d:=n mod 10;
	Ans:=ans*10+d;
	N:=n div 10;
	If n < > 0 then goto L1;
End.
```

روش دیگر نوشتن برنامه:

```
Program converse_while;
Var
	Ans,n,d:integer;
	D: integer;
Begin
	Ans:=0;
	Write('enter the number:');
	Readln(n);
	While(n>0) do
	Begin
		D:=n mod 10;
```

```
            Ans:=ans*10+d;
            N:=n div 10;
        End;
        Writeln(ans);
End.
```

نوشتن برنامه به زبان c

```
Main()
{
Int    ans,n,d;
       Ans=0;
       Printf("enter the number");
       Scanf("%d",n);
       If (n<0) return(0);
       While(n>0)
       {
       D=n%10;
       Ans=ans*10+d;
       N=n/10;
       }
       Printf("%d",ans);
       Return(0);  }
```

مثال : برنامه ای بنویسید که عددی درست را از ورودی گرفته و به تعداد آن ستاره چاپ کند ؟

؟ ۱. همان برنامه بالا ولی ستاره ها زیر هم چاپ شوند .

؟ ۲. برنامه ای بنویسید که دو عدد را از ورودیش گرفته و بگوید که بر هم بخش پذیرند یا نه ؟

؟ ۳. برنامه ای بنویسید که یک کاراکتر، یک عدد ، و سپس یک کارکتر دیگر را از ورودی بگیرد .

کاراکتر اول هر کارکتر ممکن است باشد .

عدد صحیح است

کارکتر آخر یا v است یا

اگر v بود ، کارکتر اول را به تعداد عدد دریافتی زیر هم چاپ کند .

اگرH بود ، کارکتر اول را به تعداد عدد دریافتی در یک خط چاپ کند

.

```
program  c – dive ;
Var     fristn , secondn , temp : Inteyers ;
    R   : Inteyer ;
Beyin
        Write  ( ´enter  two  Inteyers :´ ) ;
         Readdn  ( firstn , secendn ) ;
If  first number < secendn  then

Begin
temp : = f
F : = s
S : = Temp
End;

R : = f  mod  s
If   r = o      then
writeln ( firset , 'بخش پذیر است', Second )
Else
Writeln ( firstn , is not dividable by´ second ) ;
End.
```

```
Int      r , s , s , temp ;
Main ( )
{
Printf (" in Enter two Integer: ");
Scanf ( " % d % d , 8 f , 8 s );
if  ( f < s )  { Temp = f ; f = s ;  s = temp ;
}
r = f % s;
If  ( r = = o )
Print f ( " in  is   divi deble " )
Else
 Print f ( in   is  not  divi  deble " ) ;
}
```

If در زبان c :

If (شرط) جمله if [else جمله else]

While در زبان c:

While(شرط) جمله while

While در زبان pascal :

While(شرط) do جمله while

نمونه: معادل اسمبلی جمله if زیر را بنویسید:

```
If  q<r  then  r:=r-1
Else
R:=r+1
End.

LABLE  START:    LD  A,H
LD   B,R
CMP  A,B
JS  LABLE THEN
LD   A,R
ADD  A,1
LD   R,A
JP   L END
                    LABLE THEN:
LD  A,R
SUB  A,1
LD  R,A

LEND.
```

روندها:

قطعه برنامه هایی هستند که برای انجام کار ویژه ایی یکبار نوشته می شوند و به کار می روند.

روندهایی که مقداری برای ما باز میگردانند

پاسکال:function c:function

روندهایی که مقداری برای ما باز نمی گرادنند

پاسکال:procedure c: void function

حالت کلی به کار گیری یک روند:

Subroutine_name(پارامترها);

Procedure
Void function

A=function name (پارامترها)

Function

مثال :

a:=cos(c)
a:=sin(c)
a:=arctan(c)
y:=ln(x)

random : عدد اتفاقی میان ۱ و صفر بازمیگرداند.

Random(7) : یک عدد میان صفر تا ۷ بازمیگرداند.

قدر مطلق : Abs(x)

Chr(i): کاراکتری که مقدار کد آن در متغیر I است را نمایش می دهد.

Ord(c) : کد اسکی کاراکتر را نمایش می دهد.

Round(عدد حقیقی)

برنامه ای بنویسید که طول و وتر یک ضلع از مثلث گونیا را گرفته و طول ضلع دیگر را چاپ کند.

```
Program triangle;
Var
    A,b,c:real;
Begin
    Write('Enter Vatar:');
    Readln(a);
    Write('Enter one of sides:');
    Readln(b);
    If a>b then
        Begin
            A:=sqr(a);
            B:= sqr(b);
            C:=a-b;
            C:=sqrt(c);
            Writeln(c);
        End
    Else
        Writeln('Numbers are Wrong');
End.
```

برنامه ای بنویسید که ۳ عدد را از ورودی گرفته و بگویید کـه ایـن اعـداد می توانند طول قائم الزاویه باشند یا نه .

```
Program        Gaem – zavieh ;
Var
A , b , c : Read              temp : Read ;

Begin
Read in ( a , b , c ) ;
If     a < b          then
Begin
        Temp : a ;          a : b ;              b : = temp ;
End ;
A : = sqr ( a ) ;          b : = sqr ( b ) ;          c : = sqr ( c ) ;
Temp : = b + c ;
If      a = temp then
Write in   ( ´ this trianyle    is    Quadratic ´ )
Else  ( ´ is   not    Quadratic ´ ) ;
End .
```

نمونه ۳ برنامه ای بنویسید که شکل موج سینوسی را با ستاره (*) روی صفحه چاپ کند .

```
Program
Var
S : Real ;
J , X , I : integer;
begin
for  I : = 1  To    1000     Do
Begin
S : = I / s ;
S : = 40 + 4 * sin ( S ) ;
```

```
X : = 120 und ( S )
For : 1    To      X      Do
Write ( ´ … ´ ) ;
Write in ( ´ * ´ ) ;
End ;
End .
```

نوشتن برنامه به زبان c

```
Main()
{
Float a,b,c;
Printf("Enter Vatar:");
Scanf("%f",&a);
Printf("enter one of the sides:");
Scanf("%f",&b);
If (a>b)
        {
        A=sqr(a);
        B=sqr(b);
        C=a-b;
        C=sqrt(c);
        Printf("%f",c);
        }
Else
        Printf("Worng numbers \n");
}
```

برنامه ای بنویسید که جدول کاراکترهای اسکی را چاپ کند.

```
Program test;
Uses crt;
Var
        C,i:integer;
Begin
        For i=0 to 255 do
        C:=chr(i);
        Writeln(c);
End.
```

زبان c

```
Main()
{
Int i:
For(i=0; i<256; i=i+1)
        Printf("%c *** %d\n",I,i);
Return(0);
}
```

برنامه ای بنویسید که مختصات یک نقطه و معادله یک خط را گرفته و فاصله نقطه از خط را حساب کند.

```
Program distance;
Var
    X,y:real;   s,m:real;
    A,b,c:real d:real;
Begin
    Writeln("Enter X coordinate:");
    Readln(x);
    Writeln("Enter Y coordinate:");
    Readln(y);
    Writeln("Enter a,b,c of ax+by+c=:");
    Readln(a,b,c);
    S:=a*x+b*y+c;
    S:=abs(s);
    M:=a*a+b*b;
    M:=sqrt(m);
    If m< > 0 then
        Begin
            D:=s/m;
            Writeln("distance is :",d);
        End;
End.
```

آرایه در زبان های c و پاسکال :

فرمت در زبان پاسکال

```
Var
Ar: array[0..10]  of  type;
```

نوع:

Integer
Real
Char

آرایه را با بکارگیری واژه کلیدی array تعریف می کنیم.
جلوی کلمه array در کروشه دامنه تغییرات اندیس قرار می گیرد که مهم تعداد عناصر آرایه را نشان می دهد و مهم مقدار اندیس ها را سپس کلمه of می آید و پس از آن نوع هر یک از خانه های آرایه آورده می شود.

برای تعریف آرایه های چندبعدی ، باید چند range اندیس که میان آن ها کاما(،) قرار دارد در کروشه بنویسیم مانند:

```
M3*3:array[1..3,1..3] of integer;
```

M3*3

روش تعریف آرایه در c :

برای تعریف آرایه در c جلوی نام متغیر کروشه ایی قرار می گیرد که در آن تعداد عناصر آرایه نوشته شده است.

```
Int    i[20];
```

این جمله متغیر I را که ۲۰ عدد صحیح است تعریف می کند اندیس همیشه در c از صفر تا (تعداد-۱)

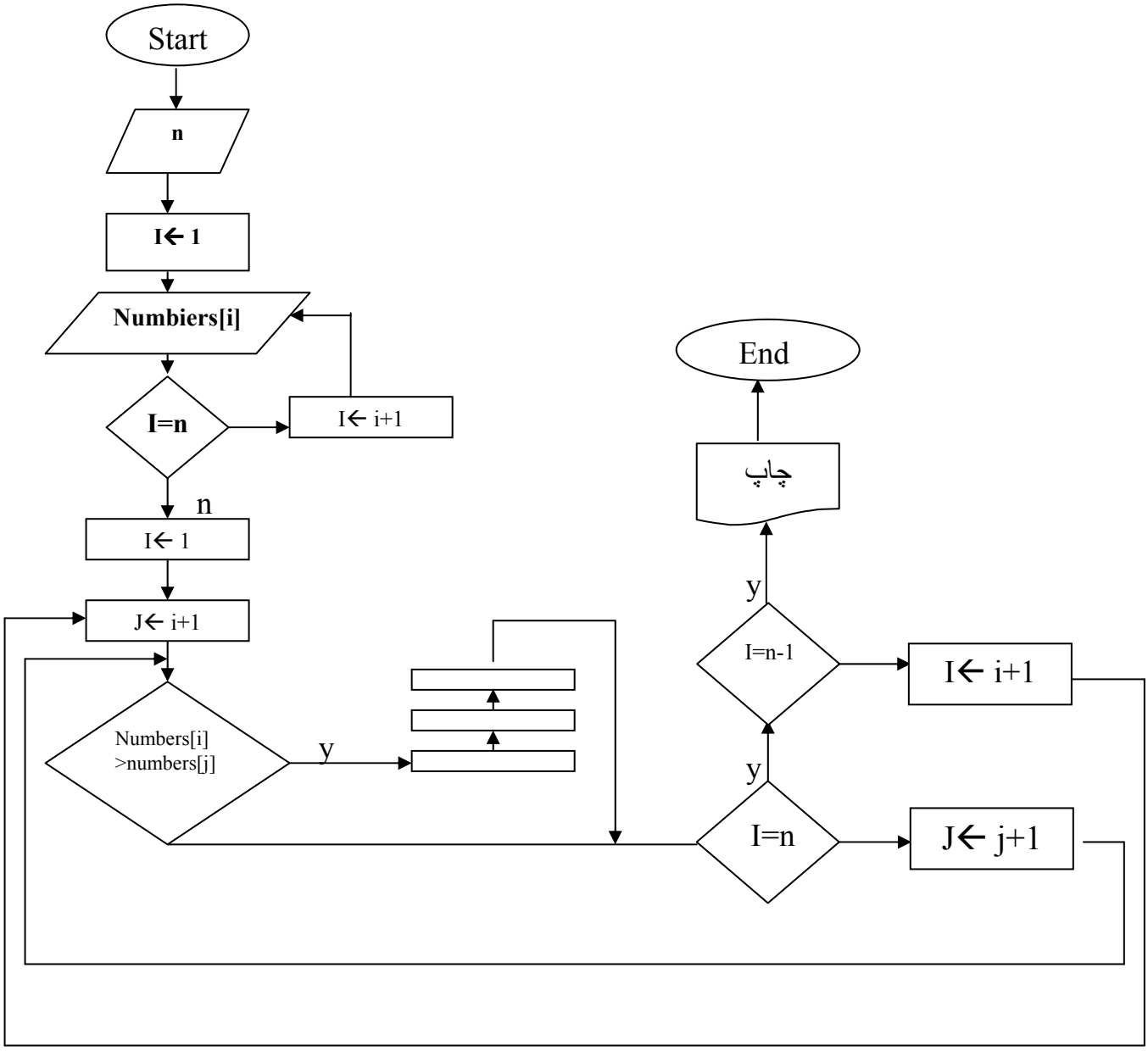
```
Char        string[100];
Int    matrix[10][10];
```

برنامه ای بنویسید که n عدد از ورودی گرفته و آنها را به روش exchangesortمرتب کرده و چاپ کند.

N را در آغاز کار از ورودی بخوانید و فرض کنید n<1000

```
Program exchangesort;
Var
      N,I,j,temp:integer;
      Numbers:array[1..1000] of integer;
Begin
      Write('enter n:');
      Readln(n);
      For i:=1 to n do
      Begin
            Write('enter a number:');
            Readln(number[i]);
      End;
      For i:=1 to n-1 do
            For j:=i+1 to n do
                  If number[i] > number [j] then
                  Begin
                        Temp:=number[i];
                        Number[i]:=number[j];
                        Number[j]:=temp;
                  End;
```

```
For i:=1 to n do
        Writeln(number[i]);
End.
```



نمونه برنامه ای بنویسید برای مرتب کردن یک آرایه n تا c از اعداد صحیح از کوچک به بزرگ بنویسید n را به عنوان اولین عدد بخوانید .
سپس n عدد را بخواند و پس از مرتب کردن آنها را چاپ کند
نمونه : برنامه ای بنویسید که دو سری ۱۰ تای عدد را در ورودی گرفته و آنها را در ۲ آرایه ۱۰ تایی ذخیره کند سپس در آرایه را مرتب کرده و پس از آن چاپ کند و پس از آن در آرایه را در یک آرایه ۲۰ تایی به گونه ای ذخیره کند که آرایه صورت شده باشد و آن را چاپ کند .

۱-برنامه ای بنویسید که ۱۰ عدد را از ورودی گرفته و آنها را به ترتیب از بزرگ به کوچک چاپ کند .
۲-برنامه ای بنویسید که n را از ورودی خوانده سپس n عدد از ورودی بخواند و آنها را مرتب کرده و چاپ کند از بزرگ به کوچک
۳-برنامه ای بنویسید که تعدادی عدد را از ورودی خوانده و آنها را مرتب کرده و چاپ کند آخرین عدد (نشانه پایان اعداد ۱- است)

۱

```
main ( )
{
int  I , j ;      int temp ;

Int       stn [ 10 ] ;

For ( i = o ; I < 10 ; stn [ I + + ] ) ;

Scan f ( " % d " , 8 stn ) ;

For ( I = o ; I < o ; + + I )
{
```

```
For ( j = I + 1 ; J < 10 ; + + )
If  ( stn [ I ] < stn [ j ] )
Temp = stn [ I ] ;
Stn [ I ] = stn [ j ] ,
Stn [ j ] = temp ;   }
print f ( " % d " , stn [ j ] ;
}
Return ( 0) ;
}
```

۳

۱- ۱٤ ۸ ۹ ۷

```
porgram  sort ;
```

I = 4تعداد اعداد

```
 type
    Array – type = array [ 1… 1000 ] of
Var
    I , J , N : inteyer ;
    Temp , a b c : inteyer ;
Begin
```

```
I : = o       read in ( N ) ;

 while : N < > -1    D
  Beyin
I : = I + 1 ;
A b c [ I ] : = N ;
Read in ( N ) ;
End ;
```

راه حل اول ۱

```
I : = o ;
If      N < > -1    then
          Repeat
  If    N < > -1   then
        Read in ( N ) ;
        bgiJ : = I + 1 ;
    a b c [ J ] : = N ;
read in ( N ) ; End ;
  unti \      N = -1 ;
```

```
N : = I ;
For I : 1     to     n-1     Do
For J : I + 6     to     N   Do
```
ادامه برنامه قبلی

رشته کاراکتری: character string

روش تعریف در پاسکال

```
Var
Strname:string;
```

متغیری است که می تواند رشته ای از کاراکترها را در خـود ذخیـره کنـد مانند "string" یا "this is a string" .

در زبان پاسکال می تـوان از اپراتورهـای > و < و = و =< و => و <> استفاده کرد و ترتیب حروف الفبا در این زمینه بکار می رود. مثلاً
'A'<'B' 'ABC'<'CDE'

با اندیس دهی به string در پاسکال از ۱ تا طول رشته میتوان به تک تک کاراکترهای یک رشته دسترسی پیدا کرد.

```
Program   fs;
Var
I:integer;
C:char;
S:string;
Begin
S:='this is an example';
I:=2;
C:=s[2];
Writeln(c);
End.
```

برنامه ای بنویسید که دو ماتریس n*n را از ورودی خوانده و در هم ضرب کرده و چاپ کند(nرا از ورودی بخوانید)

```
Program matrix;
Var
      I,j,k,n: integer;
      A,b,c:array[1..100,1..100] of real;
Begin
      Write('enter dimention n:');
      Readln(n);
      For i:=1 to n do
      For g:=1 to n do
      Begin
            Write ('enter a[',I,',',j,'j:');
            Readln(a[I,j]);
      End;
```

```
        For i:=1 to n do
        For g:=1 to n do
        Begin
                Write ('enter b[',I,',',j,'j:');
                Readln(b[I,j]);
        End;

        For i:=1 to n do
        For j:=1 to n do
        For k:=1 to n do
        C[I,j]:=c[I,j]+a[I,j]*b[I,j];
        For i:=1 to n do
        For j:=1 to n do
        Write(c[I,j],'   ');
        Writeln;
End.
```

برنامه ای بنویسید که ۱۰ نام و نمره هایشان را از ورودی دریافت کرده و آنها را به ترتیب نمره مرتب کرده و در خروجی چاپ کند.

```
Program sort_grade;
Var
Names:array[1..10]of string;
Numbers:array[1..10]of real;
I,j:integer;
Tempr:real;
Temps:string;

Begin
```

```
For i:=1 to 10 do
Begin
Write('enter a name:');
Readln(names[i]);
Write('enter its number:');
Readln(number [i]);
End;
For i:=1 to 9 do
For j:=i+1 to 10 do
If numbers[i]< numbers[j] then
Begin
Tempr:=numbers[i];
Temps:= names[i];
Numbers[i]:=numbers[j];
Names[i]:= names[j];
Numbers[j]:=tempr;
Names[j]:= temps;
End;
For i:=1 to 10 do
Writeln(names[i],'….',numbers[i]);
End.
```

برنامه ای بنویسید که ۱۰ نام و نمره هایشان را از ورودی دریافت کرده و آنها را به ترتیب نمره مرتب کرده و میانگین کلاس را حساب کند – پایین ترین و بالاترین نمره را با نام چاپ کند و درصد افتاده ها را نیز حساب کند و در خروجی چاپ کند.

```
Var
T,s:integer;
D,x:real;

Begin

S=0;
For i:=1 to 10 do
Begin
S:=numbers[i]+s;
End;

X:=s/10;
Writeln('the average is =',x);
Writeln(names[1],'=',numbers[1],'is maximum');
Writeln(names[10],'=',numbers[10],'is minimum');

T:=0
For i:=1 to 10 do
If numbers[i]<10 then
T:=t+1;
D:=(t/10)*100 ;
Writeln('the darsad is ' ,d);

End.
```

برنامه ای بنویسید که عدد صحیح n را از ورودی بخواند سپس n نمره و n نام را از ورودی خوانده و در ارایه ذخیره کند. سپس با دریافت هر عدد نام همه کسانی را که ان نمره را گرفته اند چاپ کند و اگر کسی آن نمره را نگرفته پیغام 'no one' را چاپ کند.

```
Program searcher;
Var
I,j,n: integer;
Name: array [1..100] of string;
Act: array [1..100 ] of real;
D:real;

Begin

Write("Enter the numbers:")
Readln(n);

For i:=1 ti n do
Begin
Write('enter the names:');
Readln(name[i]);
Write("enter its number :");
Readln(act[i]);
End;

I:=1;
J:=0;

Begin
Write("enter the number for search:");
Readln(d);
End;
```

```
For i:=1 to n do
If d:=act[i] then
Begin
Write(name[i]);
J:=j+1;
End;

If j:=0 then
Writeln("No one");
End.
```

برنامه ای برای نقش راهنما در بازی magic-number بنویسید.

امتیازPoints

حدسguss

عدد اتفاقیrandom

```
Program magic_number_guide;
Var
P:integer;
R:integer;
G:integer;

Begin

P:= 10;
R:=random(100);
G:= 101;
```

```
While g < > r and p>=0 do
Begin
write("enter your guss:");
readln(g);
if g<r then

begin
writeln("too small");
p:=p-1;
end

else

if g>r then
begin
writeln("too large");
p:=p-1;
end

else

writeln("OK your score is",p);

end;
if (p=0) then
writeln('you loose.the number is:',r);

end.
```

حلقه repeat until

Repeat جمله repeat until شرط;

روش اجرا : یکبار هنگام ورود به حلقه جمله repeat انجام می شود
سپس شرط بررسی می گردد و تا هنگامی که شرط برقرار نشده است.
جمله repeat تکراری می شود.

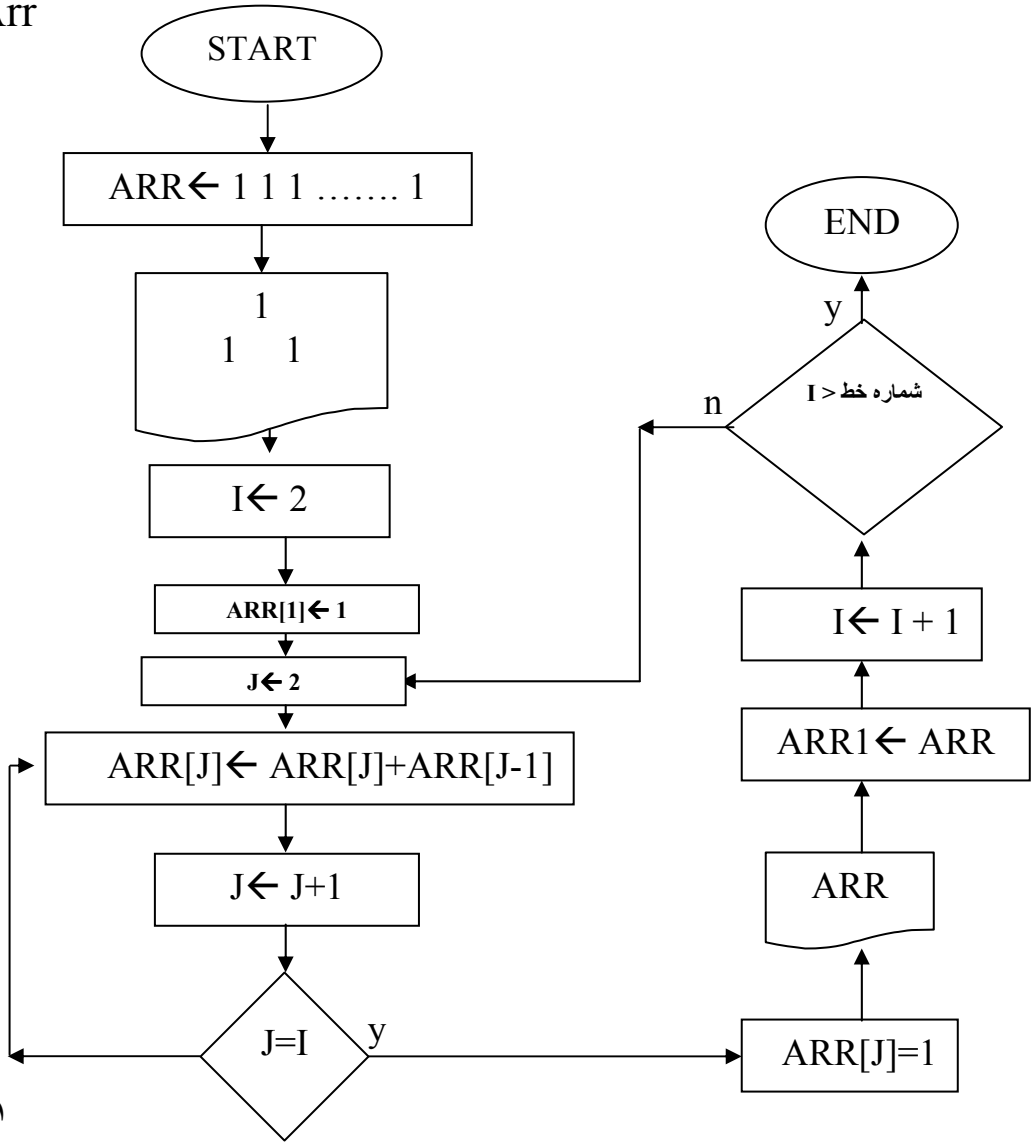
نوشتن برنامه قبل با استفاده از حلقه Repeat-until

```
Program magic_number_guide;
Var
P:integer;
R:integer;
G:integer;

Begin

P:= 10;
R:=random(100);

Repeat
Write('enter your guass :');
Readln(g);

If  r>g  then
```

```
Begin
Writeln('small');
P:=p-1;
End;

If  r<g  then
Begin
Writeln('large');
P:=p-1;
End;

If  r=g  then
Begin
Writeln('OK! Score is:',p);
End;

Until g=r or p=0;

If  (p=0) then
Writeln("you loose. The number is",r);

End.
```

حل مثلث خیام نیوتن:

```
Program khayam;
Var
Arry1,arr2:array[1..20]of integer;
K,I,j:integer;
Begin
For i:=1 to 20 do
Arry1[i]:=1;
For i:=1 to 10 do
Begin
Ini=0;
J:=1;
Arr2[j]=ini+arry1[1]
For j:=2 to i
Arr
```



Function و procedure در زبان های c پاسکال:
گاهی در برنامه های ما پیش می آید که یک عمل تکراری یا بخشی از یک کد را بارها و بارها استفاده می کنیم.
همچنین گاهی طولانی شدن برنامه یا پیچیدگی کارهای انجام شونده در برنامه باعث سردرگمی و غیر قابل فهم شدن برنامه میشود.
در چنین مواردی بهتر است که کارهای گوناگون را در زیر برنامه ها انجام دهیم و برنامه اصلی تنها به فراخوانی زیر برنامه ها اختصاص یابد.

```
Program arrange_3;
Var
A,b,c:integer;
Temp:integer;

Begin

Readln(a,b,c);

If a<b then
Begin
Temp:=a;
A:=b;
B:=temp;
End;

If a<c then
Begin
Temp:=a;
A:=c;
c:=temp;
End;

If b<c then
Begin
Temp:=b;
b:=c;
```

```
c:=temp;
End;

Writeln(a,b,c);
End.
```

راه کوتاهتر :

```
Program arrange_3;
Var
A,b,c:integer;

Procedure p_swap(var p1,p2:integer);
Var   temp:integer;
Begin
Temp:=p1;       p1:=p2;    p2:=temp;
End;

Begin
Readln(a,b,c);
If a<b then p_swap(a,b);
If a<c then p_swap(a,c);
If b<c then p_swap(b,c);
writeln(a,b,c);
end.
```

یک روند را با واژه کلیدی procedure می توان در بخش var از header برنامه تعریف کرد.
پس از کلمه procedure نام روند قرار می گیرد که یک identifier است .
سپس درست مانند برنامه می توان متغیرها و ثابت های نیاز برای procedure را در بخش های var و const و type از procedure تعریف کرد که مجموع اینها ، procedure را در بخش های var و const و type از procedure تعریف کرد که به مجموع اینها ، procedure header ، بدنه procedure همانند بدنه برنامه آورده می شود که تفاوت آن ; در end ، procedure در نقطه در end آخر برنامه است.

سپس در برنامه اصلی باید procedure را فراخوانی کرد (call)
فراخوانی شامل ذکر نام procedure و دادن متغیرهای برنامه به procedure به عنوان پارامتر می باشند.

ذکر کلمه var در معرفی پارامترها به procedure امکان می دهد که بتواند تعداد پارامترها را تغییر دهد.

VAR P1,P2: INTEGER;

Scope: فضای دید ، برد دید.

۱- هر متغیری که درون function یا procedure یا در سربرگ function یا procedure با یک نام تعریف می شود ، برای بخش های خارج از function یا procedure قابل شناسایی نیست (مانند اینکه temp را درون برنامه بیاوریم)
۲- اگر نامی هم در داخل یک function یا procedure و هم در بیرون آن استفاده شود هرگاه درون function یا procedure به آن رجوع کنیم منظور تعریف بیرونی است.

حالت کلی type :

Type
Type-name=تعریفtype

۱- یک پروسیجر برای خواندن یک ماتریس n*nاز ورودی بنویسید که n و ماتریس را بعنوان پارامتر بپذیرد.

۲- یک پروسیجر برای ضرب دو ماتریس n*n درهم بنویسید که a,b,c را که سه ماتریس هستند بعنوان ورودی گرفته و عمل c← a*b را انجام دهد.

۳- Procedure برای چاپ یک ماتریس n*n بنویسید که n و ماتریس یاد شده را بعنوان پارامتر بپذیرد.

۴- با بکارگیری این سه زیر برنامه برنامه ای بنویسید که دو ماتریس n*n را از ورودی خوانده و ضرب آنها را چاپ کند(n را از ورودی بخوانید.)

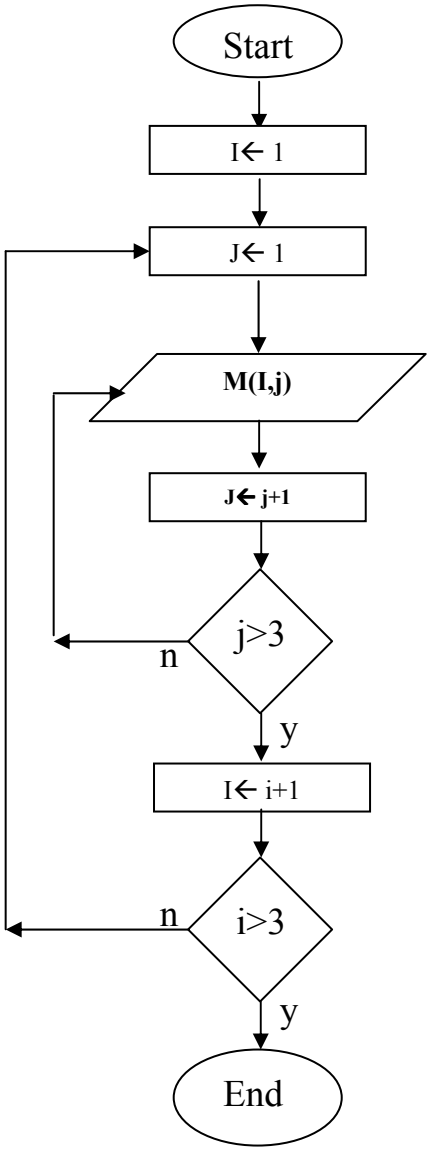
```
Program a*b_modular;
Type
Matrix10*10=array[1..10,1..10]of integer;
Var
N:integer;
```

```
M1,m2,m3:matrix10*10;

Procedure matrixread(dim:integer;var a:matrix10*10);
Var
I,j: integer;

Begin

For i:=1 to dim do
For j:=1 to dim do
Begin
Write('enter m[',I,',',j,']:');
Readln(a[I,j]);
End;

End;

Procedure mul(a,b:matrix10*10;var c:matrix10*10;
dim:integer   );
Var
I,j,k : integer;
For i:= 1 to dim do
For j:=1 to dim do
Begin
C[I,j]=0;
For k:=1 to dim do
C[I,j]+a[j,k]*b[k,j];
End;
End;

Procedure matrixwrite(dim:integer;a:matrix10*10);
Var
I,j: integer;
Begin
For i:=1 to dim do
```

```
Begin
For j:=1 to dim do
Write(a[I,j],'….');
Writeln;
End;
End;

Begin{main block}

Readln(n);
Matrixread(n,m1);
Matrixread(n,m2);
Mul(m1,m2,n,m3);
Matrixwrite(n,m3);

End.
```

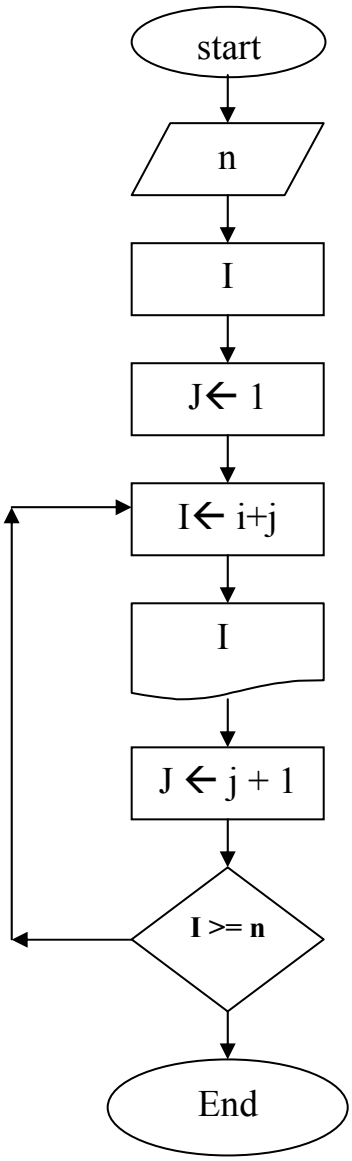
فلوچارتی برای خواندن یک ماتریس ۳*۳ از ورودی و ذخیره ان در حافظه بکشید و برنامه اسمبلی آن را نیز بنویسید.

```
Start
Ld    a,1
Ld    I,a
Label2:ld  a,1
Ld    j,a
Label1:ld  b,i
Ld    c,j
Sub   b,1
Sub   c,1
Mul   b,3
Add   b,c
Ld    c,m
Add   c,b
Inputb
Ld    (c),b
Ld    a,j
Add   a,1
Ld    j,a
Ld    a,j
Cmp 3,a    3-j
Jns   label1
Ld    a,i
Add   a,1
Ld    I,a
Ld    a,i
Cmp 3,a
Jns   label2
End_label;
```



فلوچارت سری زیر را بنویسید.

1,3,6,10,15,…,n



Recordها و structure ها :

گاهی وقت ها پیش می آید که نیاز داریم داده هایی از type های گوناگون را در کنار هم بصورت یک داده واحد دسترسی کنیم. این نیاز به دلیل ارتباط ویژه این داده ها با هم پیش می آید برای نمونه :

```
{
Name: string;
Family: string;
Entrance-year: integer;
Units: integer;
Average:real;
}
Program student
Type
Stdrecord:=record
Name: string;
Family:string;
Av:real;
Units:integer;
End;

Stdarray:array[1..100] of stdrecordl

Var
Stda:stdarray;
```

برنامه ای بنویسید که عدد صحیح n را از ورودی خوانده و سپس n بار نام و فامیل و معدل و تعداد واحد را برای n دانشجو از ورودی بخواند و سپس آنها را براساس معدل مرتب کرده و نام ها و فامیل آن ها را چاپ کند.

```
Program student;

Type
Stdrecord=record
Name:string;
Family:string;
Av:real;
Units:integer;
End;

Stdarray=array[1..100]of stdrecord;
Var
Stda:stdarray;
Stdtemp:stdrecord;
I,n:integer;

Begin
Writeln('enter number of students:');
Readln(n);
For i:=1 to n do
Begin
Write('enter name:');
Readln(std[i].name);
```

```
Write('enter family:');
Readln(std[i]. family);

Write('enter Av:');
Readln(std[i].av);

Write('enter units:');
Readln(std[i].units);

End;

For i:=1 to n-1 do
For j:=i+1 to n do
If stda[i].av < stda[j].av then
Begin
Stdtemp:=stda[i];
Std[i]:=stda[j];
Stda[j]:=stdtemp;
End;

For i:=1 to n do
Writeln(I,'_',stda[i].name,stda[i].family);
End.
```

برنامه ایی بنویسید که اندازه شعاع دایره را از ورودی خوانده و محیط و مساحت آن را چاپ کند.

```
Program circle;

Const
N_pi=3.14;

Var
R:real      env:real    a:real;

Begin
Readln(r);
A:=r*r*n_pi;
Env:=  2*r*n_pi;
Writeln(env,a);
End.
```

کاربرد case :

```
Var
C:char;
Begin
Repeat until keypressed;
If c='1' then      writeln('1')
Else if c='2' then       beep
Else if c='3' then       writeln('three')
Else if c='4' then       exit;
End;
```

نحوه نوشتن برنامه قبل با استفاده از ساختارcase

```
Case c of
'1':writeln(1);
'2':beep;
'3':writeln('three');
'4':exit(0);
```

برنامه ای بنویسید که با بـه کـارگیریinseel ،تعـداد nعـدد را از ورودی خوانده و آنها به گونه ایی در آرایه ایی ذخیـره کنـد کـه ارایـه خودبخـود sort شده باشد(فرض اعداد تکراری نیـستند)(insert_sort) نـشانه پایـان اعداد ۱- است.

اول 4 1 7 16 3 آخر

```
Main
{
Int el;
Int ar[1000];
Int n=-1;
Int I;
Scanf("%d",&del);
While(el != -1)
{
If (n=-1) inseel(ar,0,&n,el);
Else
{
```

```
For(i=0;ar[i]< = el , I <= n ; ++i);
Inseel(ar,I,&n,el);
}
Scanf("%d", &el);
}
For (i=0;I <= n ; ++i)
Printf("%d",ar[i]);
Return(0);
}
```

برنامه به زبان پاسکال

```
Program ins_sort;
Var
El,n,i:integer;
Ar:array[1..100]of integer;

Begin
Readln(el);
While el < > -1
Begin
If n=1 then
Inseel(ar,0,n,el)
Else
Begin
I:=0;
While ar[i] < = el  and i<=n do
I:=i+1;
Inseel(ar,I,n,el);
End;
Readln(el);
End;
For i:=0 to n do
```

```
Writeln(ar[i]);
End.
```

زیربرنامه ای بنویسید که یک رشته عددی – کاراکتری به عنـوان پـارامتر دریافت کرده و عددهای ان را جداکرده و در یک ارایه ذخیـره کنـد.فـرض می کنیم در زبان c انتهای رشته کاراکتری ، کاراکتر null(کاراکتر بـا کـد ۰) باشد.

```
Function isdigite(c:char):bool;
Begin

If ord(c) <= ord('9') and
Ord(c)>=ord('0')

Then

Isdigite:=true
Else
Isdigite:=false;
End;
```

```
Procedure seprate(s1:string; var s2:string)
Var
I:=1;
S2:='';
While i<= lengh (s1) do
Begin
If is digit(s1[i]) then
S2:=s2+s1[i];
I:=i+1;
End;
```

```
End;
```

زبان c :

```
Int isdigit(char c)
{
If  c>='0' & c>='9'
Return(1)
Else
Return(0);
}
```

0	false
1	true

برنامه ای بنویسید که ٦ عدد را که مختصات x و y سه نقطه هستند را بگیرد و معادله خط های اضلاع یک Δ را که این سه نقطه می سازند چاپ کند . فرض: این سه نقطه بر یک خط نیستند.

```
Procedure line_equation(x1,y1,x2,y2:real);
Begin
M:=(y2-y1)/(x2-x1);
C:=y1-m*x1;
Writeln('y=',m,'*x+(',c,')');
End;
If c>0 then
Writeln('y=',m,'*x+',c);
Else
If c<0 then
Begin
C:= -c;
Writeln('y=',m,'*x+',c);
```

```
Else
Writeln('y=',m,'*x');
```

برنامه کلی

```
Program line3;
Var
X1,y1,x2,y2,x3,y3:real;
Begin
Readln(x1,y1);
Readln(x2,y2);
Readln(x3,y3);

Line_equation(x1,y1,x2,y2);
Line_equation(x1,y1,x3,y3);
Line_equation(x2,y2,x3,y3);

End.
```

فرض ۱۰۰۰۰ وام گرفته ایم. هر قسط بدون سود = ۱۰۰۰ ماهیانه

ماهیانه ۵ ٪ سود به پول تعلق می گیرد.

تعداد قسط ها / (کل پول + کل سود)=مبلغ هر قسط با سود

برنامه ای بنویسید که میزان وام و تعداد ماه های بازگشت را گرفته و مبلغ هر قسط را با سود چاپ کند.

قسط با سود=(اصل + سود)/(تعداد ماه ها)

```
Program lean;
Var
Perceny:real;
Ben:real;
Share:real;
Money:real;
Monthes:integer;

Begin
Readln(percent,money,months);
Ben:=(percent/100)*month*(month+1)*money/(monthes*2)
Share:=(money+ben)/monthes;
Writeln(share);
End.
```

تهیه کننده :

امین جمشیدی نیا

شماره دانشجویی :

۷۹۱۶۱۴۴۳