مبانی برنامه نویسی
C

دانشکده برق و رباتیک

دانشگاه صنعتی شاهرود

حسین خسروی

۱۳۹۵

HosseinKhosravi@gmail.com

درس مبانی کامپیوتر و برنامه نویسی

- منبع: چگونه به زبان C برنامه بنویسیم
  - C How To Program
  - مولف: دیتل
- ارزشیابی (تقریبی)
  - میانترم: ۵ نمره
  - پایانترم: ۱۰ نمره
  - تکالیف و پروژه و کوئیز: ۵ نمره

ویژگیهای کامپیوتر در مقایسه با انسان

- سرعت زیاد
- عدم خستگی از تکرار
- دقت زیاد
- قدرت ذخیره سازی بالا

ویژگیهای انسان در مقایسه با کامپیوتر

- خلاقیت
- تفکر
  - کشف راه حل جدید
- اراده
  - انتخاب راه حل بهتر با توجه به شرایط
- درک و احساس

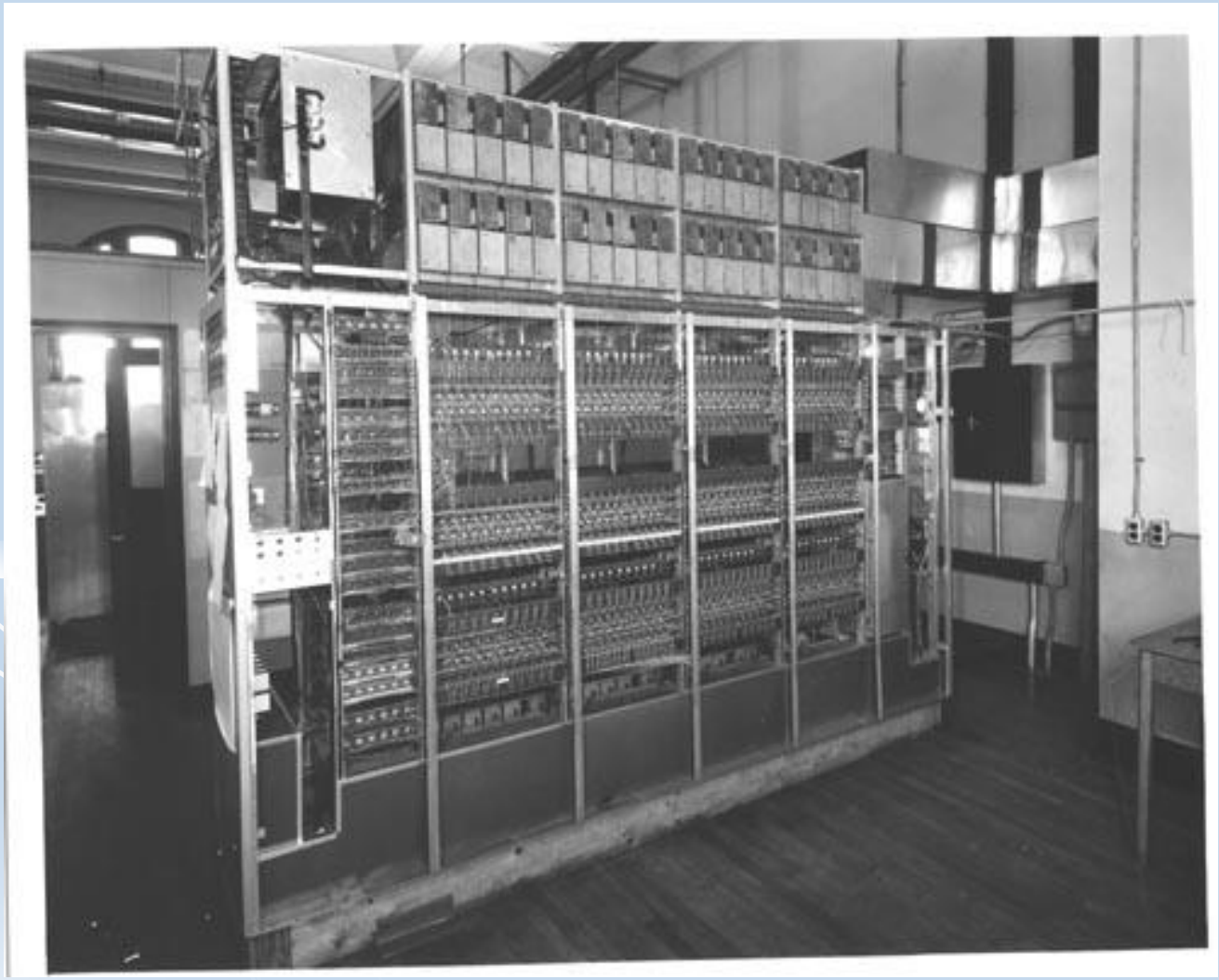
انواع کامپیوتر

- سوپر کامپیوترها
- سرورها
- میکرو کامپیوترها
  - کامپیوترهای شخصی
  - ایستگاههای کاری
  - کامپیوترهای قابل حمل: لپ تاپ، تبلت، موبایل

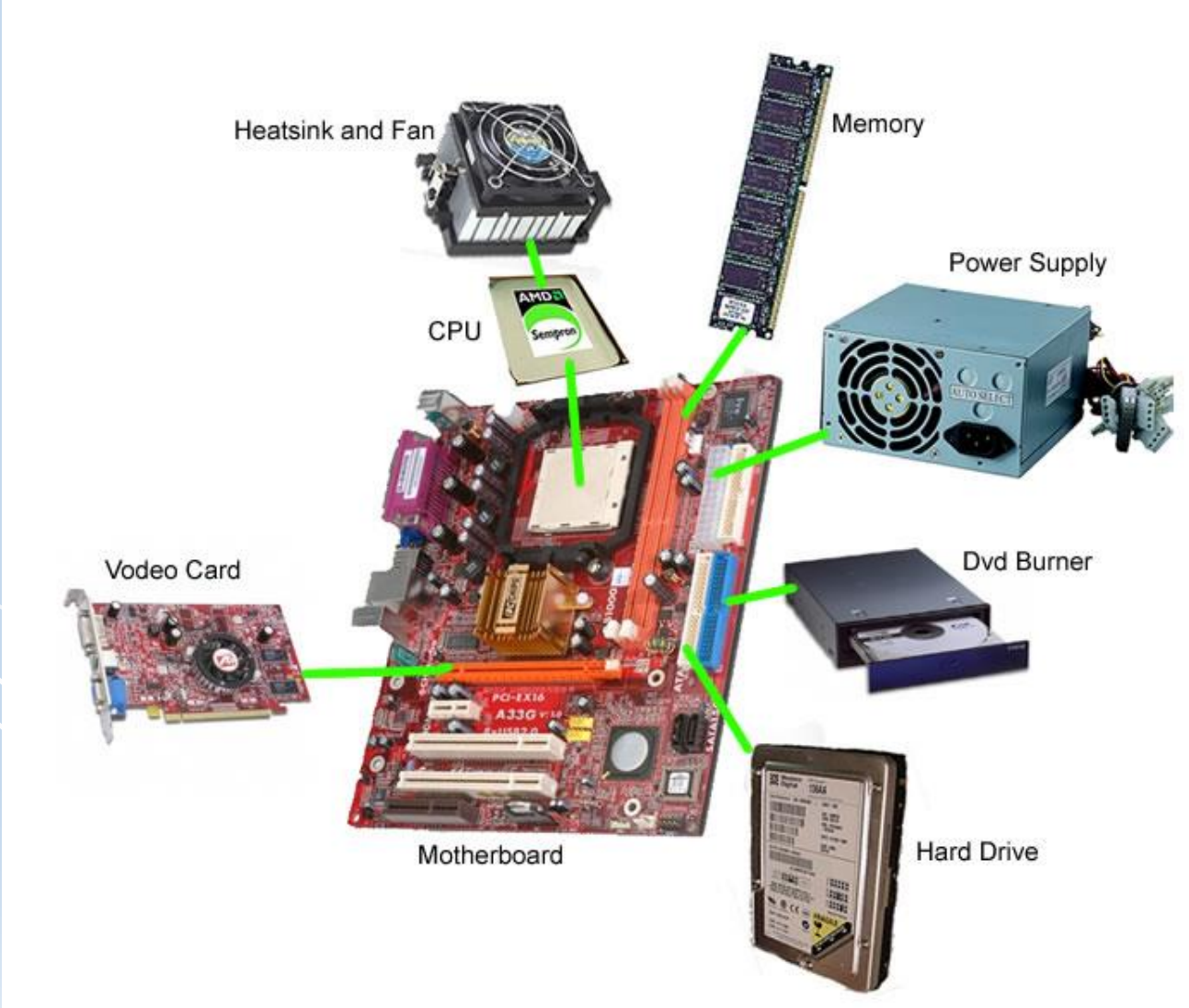
ابر کامپیوترها (Super Computers)

- سریعترین، قدرتمندترین و گرانترین کامپیوترهای موجود
- کاربرد در زمینه های تحقیقاتی، فضایی، هواشناسی، شبیه سازیها
- سرعت چند ده ترافلاپس (Tera FLOPS)
- هر ترافلاپس معادل هزار میلیارد محاسبه اعشاری در ثانیه
- Blue Gene شامل ۱۳۰ هزار پردازشگر در مساحتی به اندازه نصف زمین تنیس

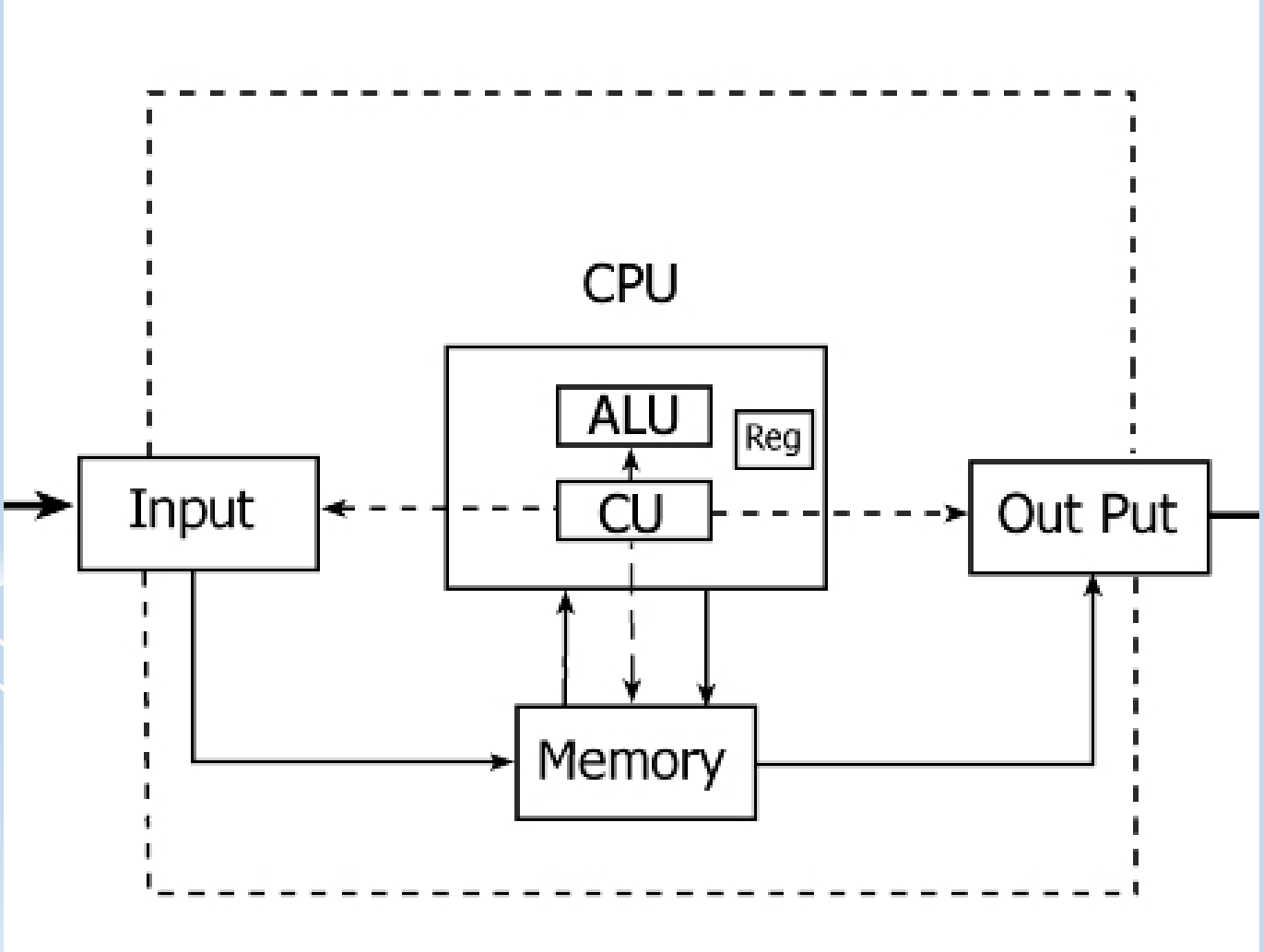
کامپیوتر های قدیمی



اجزاء کامپیوتر شخصی



ساختمان یک کامپیوتر



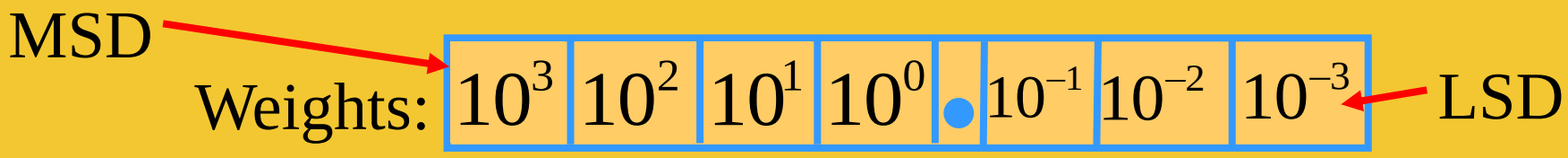
دستگاههای جانبی

- ورودی
  - ماوس
  - کیبورد
  - انواع حافظه
- خروجی
  - نمایشگر
  - چاپگر
  - انواع حافظه
- ارتباطی
  - کارت شبکه
  - مودم
- انواع حافظه
  - حافظه جانبی یا دیسک سخت
  - حافظه اصلی RAM
  - CD و DVD
  - حافظه فلش
  - فلاپی دیسک

آشنایی با سیستم اعداد

مرور سیستم دهدهی

- پایه متداول، مبنای ۱۰ است و ارقام آن ۰، ۱، ... ۹ می باشند.
- برای اعداد بزرگتر از ۹، یک رقم با اهمیت تر به سمت چپ اضافه کنید. مثلا: ۱۹>۹
- هر محل دارای یک وزن است:

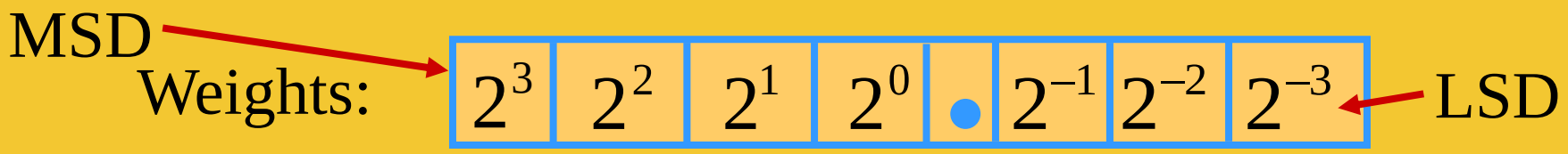


- ✓ به عنوان مثال عدد ۱۹۳۶.۲۵ را می توان به صورت زیر نمایش داد:

$$1\times10^3+9\times10^2+3\times10^1+6\times10^0+2\times10^{-1}+5\times10^{-2}$$

سیستم عدد نویسی دودویی

- پایه ۲ است و ارقام ۰، ۱هستند.
- برای اعداد بزرگتر از ۱، یک رقم با اهمیت تر به سمت چپ اضافه کنید. مثلا: ۱۰>۱
- هر محل دارای یک وزن است:



✓ به عنوان مثال عدد ۱۰۱۱۱.۰۱ را می توان به صورت زیر محاسبه کرد:

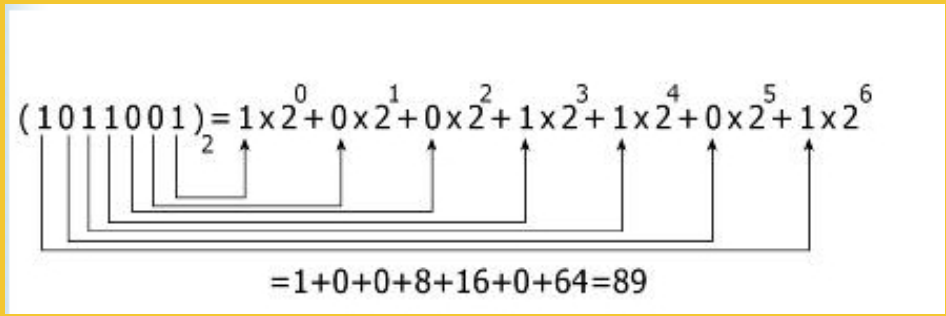
$$1\times2^4+0\times2^3+1\times2^2+1\times2^1+1\times2^0+0\times2^{-1}+1\times2^{-2}=$$
$$=1\times16+0\times8+1\times4+1\times2+1\times1+0\times0.5+1\times0.25=23.25$$

مبنای دو (باینری)

ارزش مکانی در مبنای 10 توانهای 10 است (یکان، دهگان، صدگان،...) و ارزش مکانی رقم ها (بیتها) در مبنای 2 توانهای 2 است.

$2^0 = 1$ $2^1 = 2$ $2^2 = 4$ $2^3 = 8$ $2^4 = 16$

$2^5 = 32$ $2^6 = 64$ $2^8 = 256$ $2^7 = 128$ $2^9 = 512$ $2^{10} = 1024$



سیستم عدد نویسی دودویی

- $(110000.0111)_2 = (\ ?\ )_{10}$
  - جواب: ۴۸.۴۳۷۵

در دنیای کامپیوتر:

- $۲^{۱۰}$=۱۰۲۴ با K (کیلو) نشان داده می شود.
- $۲^{۲۰}$=۱۰۴۸۵۷۶ با M (مگا) نشان داده می شود.
- $۲^{۳۰}$= G (گیگا)
- $۲^{۴۰}$ = T (ترا)

- چه تعداد بیت در یک حافظه 16GByte وجود دارد؟

مبناهای ۸ و ۱۶

- مبنای ۸
  - پایه ۸ است و رقمها 0, 1, 2, 3, 4, 5, 6, 7 هستند

$$(236.4)_8 = (158.5)_{10}$$

$$2\times8^2+3\times8^1+6\times8^0+4\times8^{-1}=158.5$$

- مبنای ۱۶
  - پایه ۱۶ است و رقمهای ۰ تا ۹ از سیستم دهدهی قرض گرفته شده اند و از A, B, C, D, E, F به ترتیب برای نمایش رقمهای ۱۰، ۱۱، ۱۲، ۱۳، ۱۴، ۱۵ استفاده می گردد.

$$(D63FA)_{16} = (877562)_{10}$$

$$13\times16^4+6\times16^3+3\times16^2+15\times16^1+10\times16^0=877562$$

تبدیل از دهدهی به دودویی

تبدیل اعداد اعشاری:

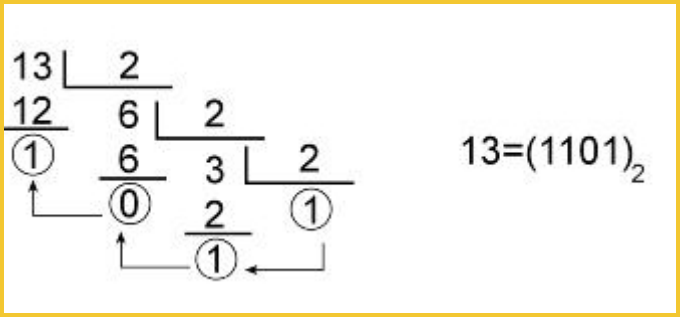
- ✓ معادل دودویی $(0.8542)_{10}$ را تا شش رقم دقت پیدا کنید.

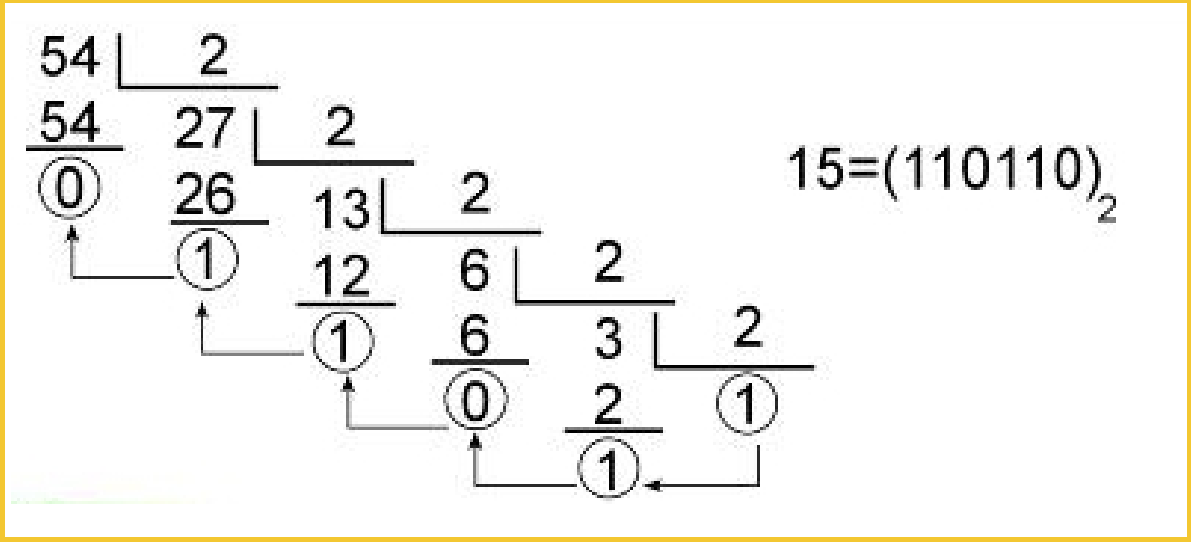
$$0.8542 \times 2 = 1 + 0.7084 \quad a_{-1} = 1$$
$$0.7084 \times 2 = 1 + 0.4168 \quad a_{-2} = 1$$
$$0.4168 \times 2 = 0 + 0.8336 \quad a_{-3} = 0$$
$$0.8336 \times 2 = 1 + 0.6672 \quad a_{-4} = 1$$
$$0.6672 \times 2 = 1 + 0.3344 \quad a_{-5} = 1$$
$$0.3344 \times 2 = 0 + 0.6688 \quad a_{-6} = 0$$

$$(0.8542)_{10} = (0.a_{-1}a_{-2}a_{-3}a_{-4}a_{-5}a_{-6})_2 = (0.110110)_2$$

$$(53.8542)_{10} = (\quad ? \quad)_2$$

تبدیل از مبنای ده به مبنای دو





جمع دودویی

$\begin{array}{r} 1 \\ 1 \\ \hline 1\ 0 \end{array}$ +	$\begin{array}{r} 1 \\ 0 \\ \hline 1 \end{array}$ +	$\begin{array}{r} 0 \\ 1 \\ \hline 1 \end{array}$ +	$\begin{array}{r} 0 \\ 0 \\ \hline 0 \end{array}$ +

در جمع آخر، 1، رقمی نقلی است که با بیتهای بعدی جمع می شود.

مثال : جمع زیر را در مبنای 2 انجام دهید.

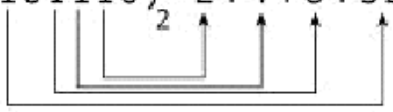
$29=(00011101)_2$

$17=(00010001)_2$

$\begin{array}{r} 29 \\ 17 \\ \hline ? \end{array}$ +

$\begin{array}{r} {\scriptstyle 1\ \ 1} \\ 00011101 \\ 00010001 \\ \hline (00101110)_2 \end{array}$ +

$(00101110)_2 = 2+4+8+32=46$ امتحان نتیجه



شیوه های نمایش اعداد منفی

- مکمل ۱
- مکمل ۲ (مناسبترین روش)
- مقدار و علامت
- مثال: عدد ۴ بیتی ۱۰۱۱ در روش مکمل ۱، مکمل ۲ و مقدار علامت به ترتیب نمایش دهنده : ۴- ، ۵- و ۳- است

(اگر پر ارزشترین بیت، ۱ بود به معنای منفی بودن عدد است)

جزئیات بیشتر در درس سیستمهای دیجیتال

نرم افزار کامپیوتر(Software)

- بخشی از کامپیوتر است که جهت بکارگیری سخت افزار، فرمان دادن و هدایت آن مورد استفاده قرار می گیرد.
- عملکرد کامپیوتر توسط یکسری دستورالعمل به نام برنامه که توسط ما نوشته می شود، تعیین می گردد.
- نرم افزار، برنامه ای است جهت بکارگیری سخت افزار
- برنامه نیز مجموعه ای دستور العمل به منظور انجام یک کار خاص (مثل یافتن میانگین چند عدد) است

انواع نرم افزار کامپیوتر

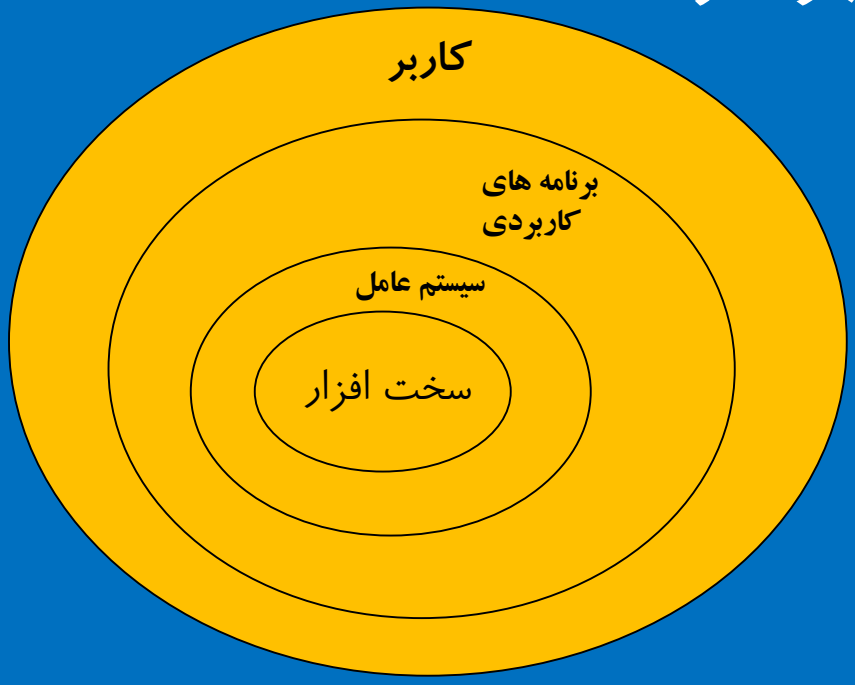
- **دو دسته کلی**
  - نرم افزارهای سیستمی (System Software)
    - برنامه هایی که رایانه یا بخشی از آن بدون آنها بی استفاده است
      - مثل سیستم عامل، مترجمها، درایورها
    - کارهای مربوط به سیستم کامپیوتر را انجام می دهند.
    - غالبا توسط شرکتهای بزرگ نوشته می شوند.
  - نرم افزارهای کاربردی (Application Software)
    - توسط هر کسی که دانش برنامه نویسی دارد قابل نوشتن است.
    - برای انجام کارهای بسیار ساده مثل ماشین حساب تا سنگین مثل فوتوشاپ، مجموعه آفیس، شبکه های اجتماعی و نرم افزارهای پردازشی هوشمند

سیستم عامل

- **مهمترین نرم افزار کامپیوتر است.**
- **برنامه بزرگ و مفصلی است.**
- **رابط بین کاربر و سخت افزار است.**
- **مدیریت منابع را بر عهده دارد.**
- **در یک کامپیوتر بطور معمول وظایف زیر را انجام می دهد:**
  - فعال کردن برنامه ها و آوردن آنها به حافظه جهت اجرا
  - نظارت بر اجرای نرم افزارها
  - دریافت ورودیها و ارسال اطلاعات به خروجیها
  - خواندن و نوشتن فایلها
  - توقف برنامه ها / تقسیم بندی زمان پردازنده بین برنامه ها و ....

سیستم عامل

- سیستم عامل اولین برنامه ای است که پس از روشن شدن کامپیوتر وارد حافظه RAM می شود.
- و سپس کنترل را در دست می گیرد و کمک می کند تا برنامه های دیگر بتوانند اجرا شوند.



چند نمونه سیستم عامل

- **DOS**
- **Windows**
  - Windows 3.1 / DOS
  - Windows 9x,ME,XP, Vista, … Windows 10 [Workstation]
  - Windows NT, 2000, 2003,…, 2015 [Server]
- **UNIX**
  - Solaris
  - BSD
  - SVR4
- **Linux**
  - RedHat
  - SUSE
  - Mandrake
  - FreeBSD
  - Debian
- **MAC OS**
- **Android**
- **iOS**

نرم افزارهای سیستمی- مترجم های زبان

- **دو دسته کلی مترجم داریم:**
  - کامپایلر Compiler
    - کل برنامه ورودی را یک جا به زبان ماشین ترجمه می کند.
    - یک برنامه مستقل و قابل اجرا ارائه می دهد.
  - مفسر Interpreter
    - برنامه مورد نظر را خط به خط خوانده، ترجمه و اجرا می کند.
    - برنامه مستقلی برای اجرا ارائه نمی دهد.
    - بلکه برای اجرا همواره به برنامه مفسر نیاز است.

زبانهای برنامه نویسی

- **مجموعه ای از از نشانه ها، قواعد و دستورالعملهایی که توسط آنها میتوانیم مقاصد و منظور خود را با آن زبان بیان کنیم**
- **با ترجمه آن توسط مترجم، این برنامه خود را برای ماشین قابل فهم نماییم.**
- **انواع زبانهای برنامه نویسی**
  - زبانهای سطح بالا High Level Language
    - به زبان طبیعی و محاوره ای ما انسانها نزدیکتر هستند
    - برای ما قابل درک هستند اما نه مستقیما برای کامپیوتر
    - وابستگی کمی به سخت افزار و ماشین خاصی دارند.
    - کار با آنها(برنامه نویسی با آنها) برای ما ساده تر است
    - امکان خطایابی و رفع خطای آنها ساده تر است
    - سرعت اجرای پایین تری دارند.
  - زبانهای سطح پایین Low Level Language
    - درست برعکس زبانهای سطح بالا

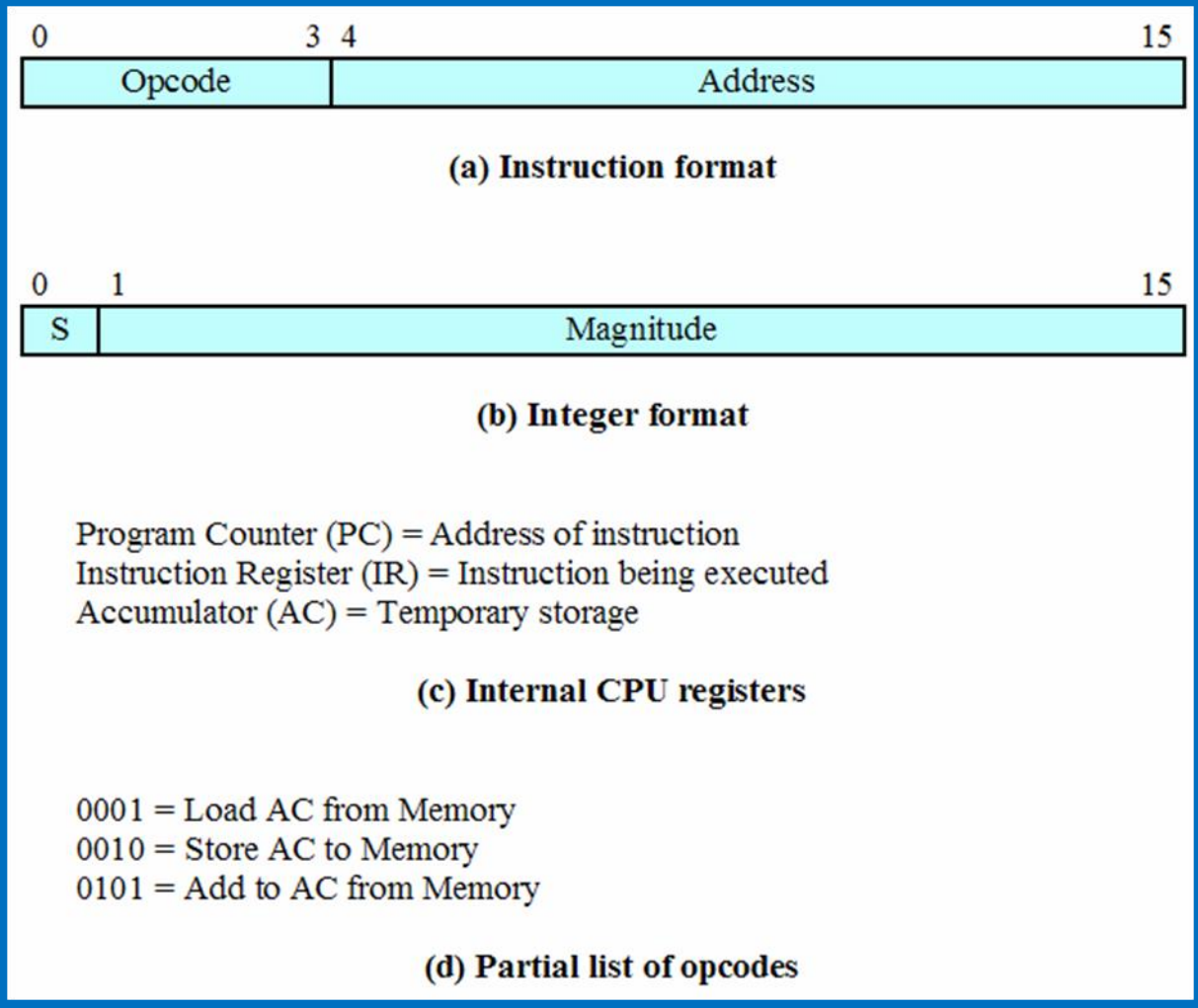
زبانهای برنامه نویسی سطح پایین

- به دو دسته تقسیم می شوند:
  - زبان ماشین (Machine Language)
    - تمام دستورات و داده ها بصورت رشته هایی از 1 , 0 بیان میشوند.
    - بطور مثال: دستورات زیر (که جهت سادگی نمایش آنها می توان از مبنای ۱۶ نیز استفاده کرد)
      - 10010001 [91H]
      - 11010101 [D5H]
      - 01001111 [3FH]
  - زبان اسمبلی (Assembly Language)
    - به جای استفاده از رشته های 0,1 از نمادهای معنادار به جای آنها استفاده می شود.
    - نمادهای کوتاه و با معنی
    - توسط برنامه اسمبلر (Assembler) این دستورات به زبان ماشین ترجمه شده و برای اجرا روی ماشین مورد نظر قابل استفاده هستند.

```
MOV     AX, BX
ADD     AX, CX
MOV     CX, DX
```

مثالی از اجرای یک برنامه

- مشخصات یک ماشین فرضی
- قالب دستورات
- قالب اعداد



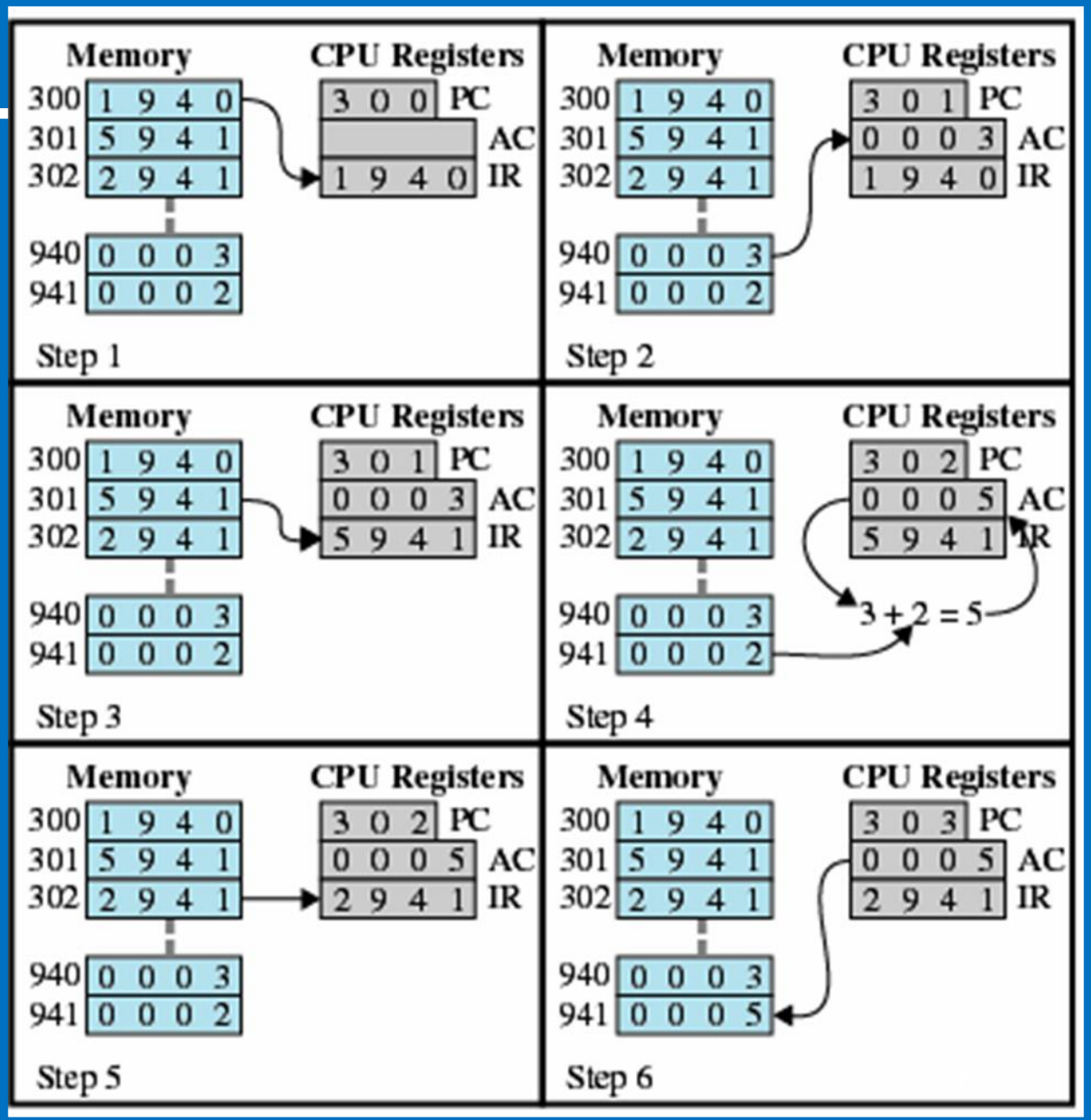
Adding Two Numbers (3+2)

Assumptions:

1 → Load (1 940) →
Load from address 940

5 → Add

2 → Store



الگوریتم و فلوچارت

- الگوریتم: قدمهای حل یک مساله
- برگرفته از نام ابومحمد بن موسی الخوارزمی
- مولف کتاب الجبر و المقابله

- در کتب قدیمی، بیشتر برای «الگوریتم اقلیدس برای تعیین بزرگترین مقسوم علیه مشترک» بکار می رفته است.

الگوریتم اقلیدس

- الگوریتم اقلیدس: دو عدد طبیعی m و n داده شده است. بزرگترین مقسوم علیه مشترک این دو را پیدا کنید.

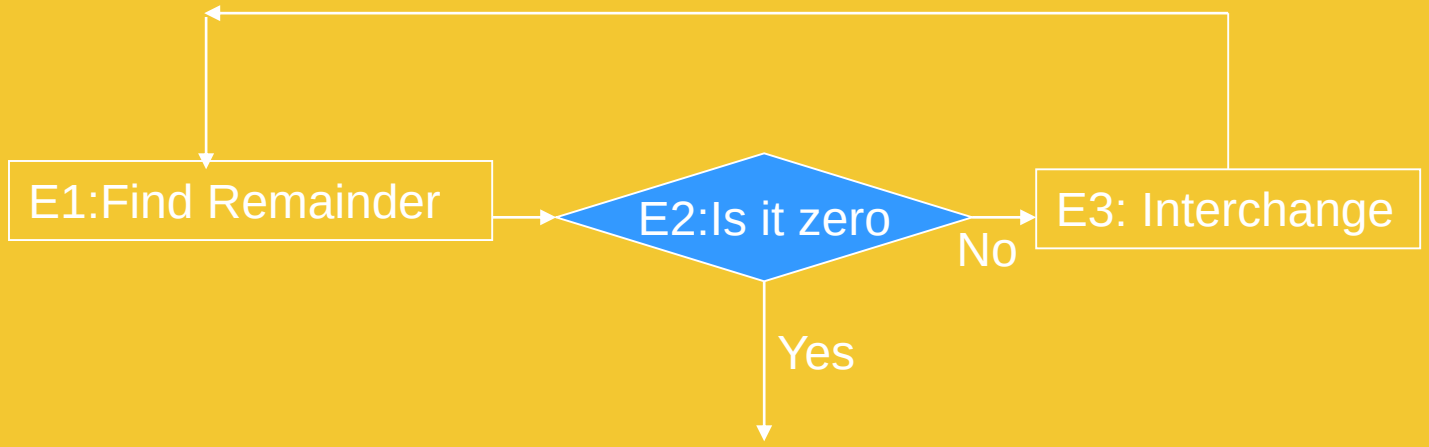
- بزرگترین مقسوم علیه مشترک اعداد طبیعی m و n، بزرگترین عدد طبیعی است که m و n بر آن بخشپذیر باشند

- عدد طبیعی N بر عدد طبیعی k بخشپذیر است اگر، باقیمانده تقسیم N بر k صفر باشد

الگوریتم اقلیدس

- **E1**: [find remainder] Divide **m** by **n** and let **r** be the reminder.*(Clearly, 0 <=r <=n)*
- **E2:** [is it zero?] if **r =0** , the algorithm terminates, **n** is the answer
- **E3:** [Interchange] Set **m** ← **n** , **n** ← **r** and go back to step E1

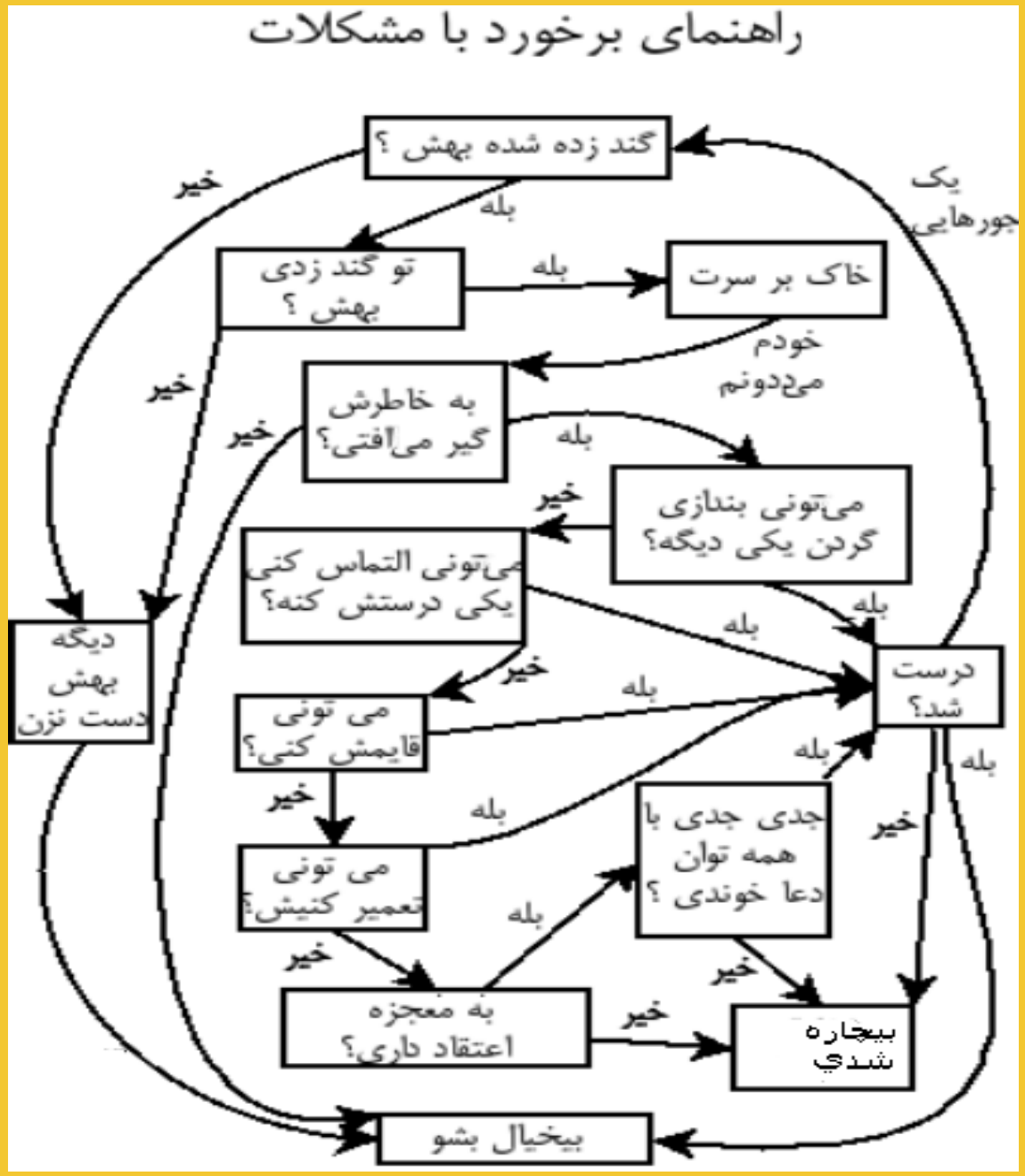
flowchart



GW-Basic Program

```
10 r = m mod n
20 if r = 0 then print n : exit
30 m=n : n= r : goto 10
```

راهنمای برخورد با مشکلات



۲- الگوریتم و فلوچارت

- شناخت مسئله
  - بررسی داده ها و یا معلومات(ورودیها)، مجهولات (خروجیها) و یافتن ارتباط منطقی بین داده ها و مجهولات
  - مثال : یافتن مساحت یک مثلث با داشتن اندازه قاعده و ارتفاع آن
    - داده ها (ورودی) – اندازه ارتفاع و قاعده مثلث
    - مجهولات (خروجی) – مساحت مثلث
    - رابطه منطقی : روش محاسبه مساحت مثلث (ارتفاع x قاعده x ۰.۵)
  - اغلب، مسائل دارای راه حلهای گوناگونی می باشند، یافتن بهترین راه حل به ابتکار، تمرین و از همه مهمتر تجربه بستگی دارد.

تعریف الگوریتم

- مجموعه دستورالعملهایی که مراحل حل یک مسئله مشخص کند با ویژگیهای زیر :
  - با یک زبان واضح، روشن و بدون ابهام و پیچیدگی
  - با جزئیات کافی(از دید اجرا کننده الگوریتم)
  - شروع عملیات
  - ترتیب اجرای دستورات
  - پایان عملیات

مجری الگوریتم

- الگوریتم ها می توانند به دو صورت انجام شوند
  - توسط ماشین (کامپیوتر)
    - بعد از تبدیل شدن به زبان مناسب برای ماشین بطور اتوماتیک توسط ماشین قابل اجرا هستند.
  - توسط انسان
    - برای حصول اطمینان از عملکرد صحیح الگوریتم، گاهی خودمان آن را بصورت دستی دنبال نموده و در واقع خود مجری الگوریتم می شویم.

کاربرد الگوریتم

- همه ما در طی روز برای انجام کارهای روزمره از روش الگوریتمی(و یا منطقی) استفاده می کنیم.
  - مانند مطالعه کتاب
  - تعویض چرخ پنجر شده
  - پختن غذا
  - پختن کیک و ....
- در واقع برای انجام هر یک از این کارها، لازم است تعدادی دستورالعملهای ساده تر را به ترتیب مناسب اجراء کرده تا به نتیجه مطلوب برسیم.

مثالی از کاربرد الگوریتم

- کتابی داریم و می خواهیم آنرا مطالعه نماییم، برای مطالعه کتاب از ابتدا تا انتها باید مراحل زیر را انجام دهیم:

1. شروع
2. باز کردن کتاب
3. از خط اول شروع به خواندن می کنیم
4. آیا به انتهای صفحه رسیده ایم یا خیر؟
5. اگر به انتهای صفحه رسیده ایم به مرحله بعدی و در غیر اینصورت به مرحله ۹ می رویم
6. آیا تا انتهای کتاب خوانده شده است یا خیر؟
7. اگر به انتهای کتاب رسیده باشیم به مرحله ۱۰ می رویم
8. صفحه بعدی را باز کرده و به مرحله ۳ می رویم
9. خواندن را ادامه می دهیم و به مرحله ۴ می رویم
10. پایان

چند نکته

- دستورات بیان شده در الگوریتم بایستی برای مجری الگوریتم قابل درک باشد.
  - بطور مثال، هنگام نوشتن نسخه توسط پزشک، بیان می شود که مثلا از این شربت ۱ قاشق مرباخوری خورده شود (و نه اینکه مثلا ۲.۵ سی سی میل شود- که برای ما که مجری الگوریتم نسخه پزشک هستیم زیاد قابل فهم نیست!)
- هنگام بیان الگوریتم بایستی سعی شود که دستورات ارائه شده بصورت عمومی و قانونمند باشد
  - بطور مثال در الگوریتم خواندن کتاب دو نوع می توان بیان الگوریتم نمود:
    - خواندن صفحه اول سپس صفحه بعدی تا اینکه به انتها برسیم.
    - خواندن صفحه اول، صفحه دوم، صفحه سوم و .....

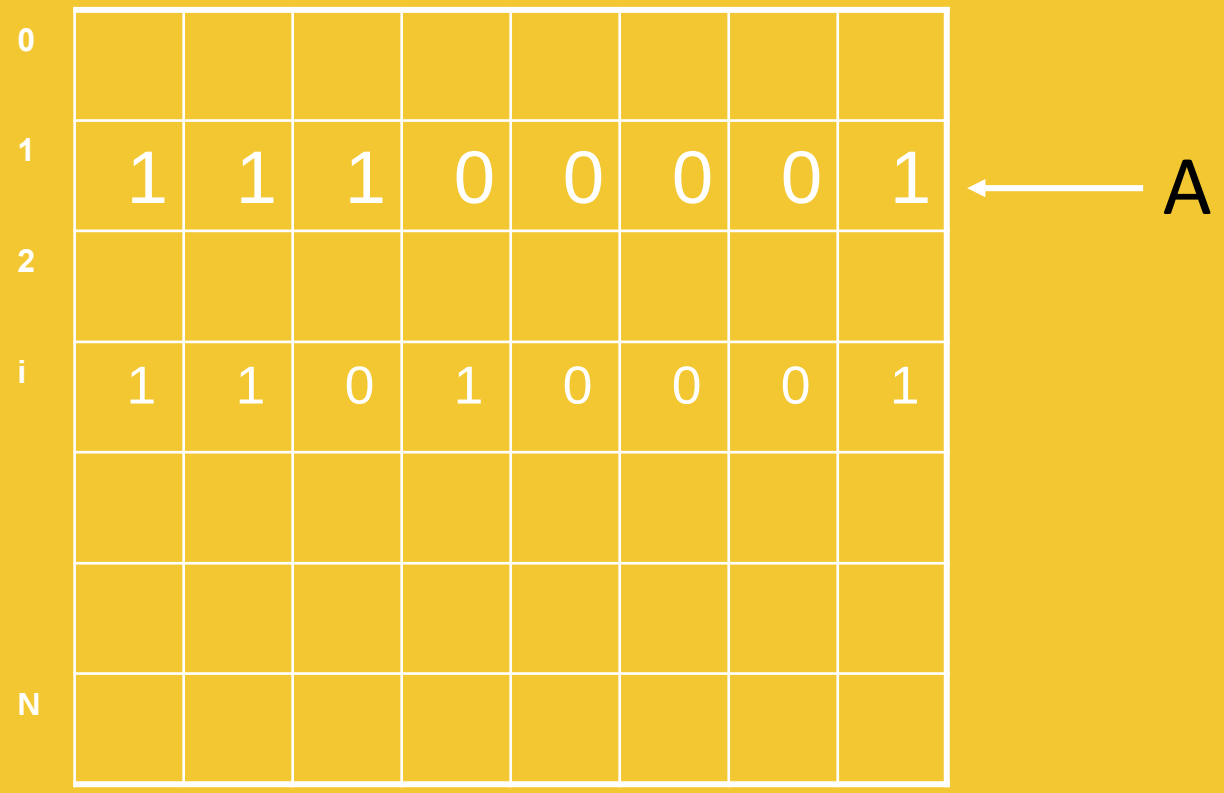
اجزای اصلی الگوریتم

- هر مساله راه حل و الگوریتم خاص خود را دارد.
- می توان برای حل یک مساله روشهای گوناگونی را ارائه داد.
- اما تمام الگوریتم ها دارای این اجزاء هستند:

1. نقطه شروع: حل مساله از کجا آغاز می گردد
   - فقط یک نقطه شروع در الگوریتم وحود دارد
2. نقطه پایان: جایی که مراحل حل مساله پایان می پذیرد.
   - به هر حال الگوریتم بایستی در یک نقطه خاتمه یابد.
   - می توان چندین نقطه پایان برای الگوریتم داشت.
3. دستورالعملها و یا جملات اجرایی

متغیر (Variable)

- به خانه ای از حافظه که داده ها و اطلاعات ورودی یا خروجی و یا اطلاعات موقت را در خود نگه می دارد متغیر گفته می شود.
- مقدار متغیر می تواند در طول اجرای الگوریتم تغییر داشته باشد.



مثالی از یک الگوریتم

- الگوریتم محاسبه و چاپ مجموع دو عدد ۱۰ و ۲۰

1. شروع
2. عدد ۱۰ را در خانه (متغیر) A قرار بده
3. عدد ۲۰ را در خانه B قرار بده
4. محتویات خانه های A , B را با هم جمع کن و در خانه C قرار بده
5. مقدار خانه C را به عنوان نتیجه چاپ کن
6. پایان

استفاده از بیان ریاضی

- بیان الگوریتم در قالب جملات نوشتاری طولانی و فهم الگوریتم را دشوار می سازد.
- الگوریتم محاسبه و چاپ مجموع دو عدد ۱۰ و ۲۰

1. شروع
2. A←10
3. B←20
4. C← A+B
5. چاپ مقدار C
6. پایان

انواع جملات مورد استفاده در الگوریتم ها

- جملات شرطی
- جملات عملیاتی (یا محاسباتی)
- جملات ورودی / خروجی
- جملات توضیحی

مثال

- الگوریتمی بنویسید که اعداد زوج دو رقمی را چاپ کند.
- (می دانیم که کوچکترین عدد زوج دو رقمی ۱۰ و اعداد زوج به اندازه ۲ واحد از هم فاصله دارند)

1. شروع
2. J ← 10
3. J را چاپ کن
4. J ← J + 2
5. اگر J <=98 است آنگاه به مرحله ۳ برو
6. پایان

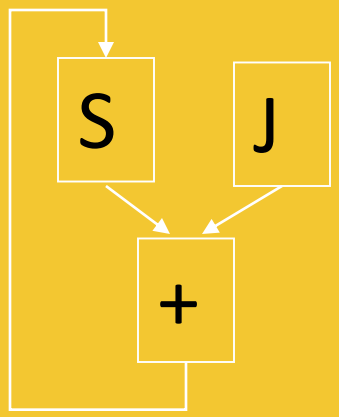
مثال

- الگوریتمی بنویسید که اعداد زوج از ۱۰۰۰ تا ۲۰۰۰ را تولید کرده و مجموع آنها را هم محاسبه کند.

1. شروع
2. J ← 1000 ، S ←0
3. J را چاپ کن و S ← S + J
4. J ← J + 2
5. اگر J <= 2000 است آنگاه به مرحله ۳ برو در غیر اینصورت مقدار S را چاپ کن
6. پایان

نکته

- اگر بخواهیم مقدار حاصل جمعی را محاسبه کنیم (مانند مثال قبل که مجموع اعداد زوج از ۱۰۰۰ تا ۲۰۰۰)
  - ابتدا متغیری (مانند S) در نظر می گیریم و مقدار اولیه آن را صفر می گداریم (یعنی اینکه هنوز هیچ مجموعی را حساب نکرده ایم) ، $S \leftarrow 0$
  - سپس ***تک تک*** جملاتی که قرار است با هم جمع کنیم را تولید کرده (با کمک یک متغیر دیگر، مثلا J در مثال قبلی) و با متغیر S جمع نموده و حاصل را در S قرار می دهیم (در واقع مقدار جدید را به حاصل قبلی می افزاییم – مانند انباره)، $S \leftarrow S + J$
  - در نهایت حاصل مجموع در این متغیر S قرار می گیرد.



ویژگیهای یک الگوریتم خوب

اگر چه یک مساله راه حلهای مختلفی دارد، مهم یافتن بهترین راه حل است

- **سادگی**
  - حتی الامکان ساده و عاری از ابهام و پیچیدگی باشد.
- **در نظر گرفتن تمام حالات خاص**
  - الگوریتم بتواند در برابر حالات و شرایط مختلف پاسخ و جواب مناسبی ارائه دهد.
  - بطور مثال هنگام حل معادله درجه دوم حالتهای منفی زیر رادیکال را در نظر بگیرد.
- **روان بودن متــن الگوریتم**
  - دستورالعملها گویا بوده و منظور آنها بسادگی درک شود.
- **حداقل بودن تعداد دستورات و جملات**

ایجاد حلقه های تکرار(Loops)

- گاهی اوقات برای حل مساله باید یک یا چند مرحله از دستورات را تکرار نمود.
- به مراحلی از الگوریتم که اجرای آنها چندین بار تکرار می شود حلقه (Loop) و یا حلقه تکرار گفته می شود
- بطور کلی حلقه های تکرار از اجزای زیر تشکیل شده است:
  - شمارنده حلقه (Counter)
    - یک متغیر کمکی که پیش از شروع حلقه به آن مقدار اولیه داده می شود
    - از طریق آن می توان تعداد دفعات تکرار حلقه را نشان داد.
  - گام افزایش (Step)
    - مقداری که پس از هر بار مراحل حلقه به شمارنده اضافه می شود.
  - شرط پایانی
    - مقدار و یا متغیری است که پس از اجرای دستورات حلقه با شمارنده حلقه مقایسه می گردد و زمان پایان اجرای دستورات حلقه را مشخص می سازد.
  - بدنه حلقه
    - دستورالعملها و جملاتی که عملیات اصلی حلقه را تشکیل می دهند.

استفاده از الگوریتم های فرعی

- با بزرگتر و پیچیده تر شدن مساله، الگوریتم آن نیز پیچیده و بزرگتر خواهد شد.
- پیچیده شدن الگوریتم چندین مشکل بوجود می آورد:
  - از وضوح و روانی الگوریتم می کاهد
  - منطق الگوریتم را دشوارتر می سازد.
  - اشکال زدایی (Debug) و پیدا کردن نقاط ضعف الگوریتم را سخت تر می سازد.
- برای رفع این مشکلات از الگوریتم های فرعی استفاده می شود
  - به این معنی که ابتدا مساله را به تعدادی بخش های مختلف و مستقل تقسیم نمود
  - و برای هر کدام الگوریتم جداگانه ای نوشت

استفاده از الگوریتم های فرعی

مثال : الگوریتمی بنویسید که با دریافت دو عدد طبیعی m,n مقدار زیر را محاسبه کند.

$$C_n^m = \frac{m!}{n!(m-n)!}$$

بهترین راه حل این مساله این است که:

الگوریتم فرعی برای محاسبه فاکتوریل نوشت مثلا با نام Fact(z) که با دریافت z بعنوان ورودی، فاکتوریل آنرا محاسبه کرده و بر می گرداند.

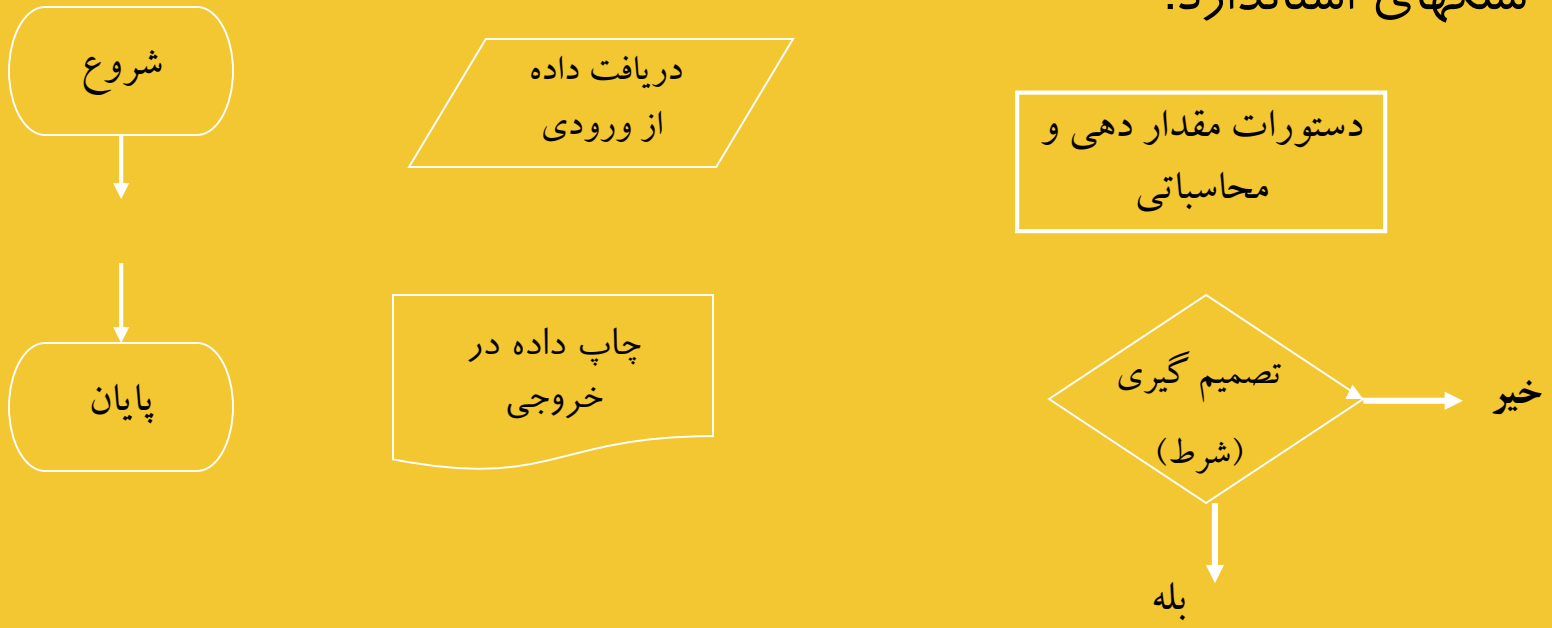
و در برنامه اصلی از این الگوریتم اصلی استفاده نمود.

A=Fact(m), B=Fact(n), C=Fact(m-n)

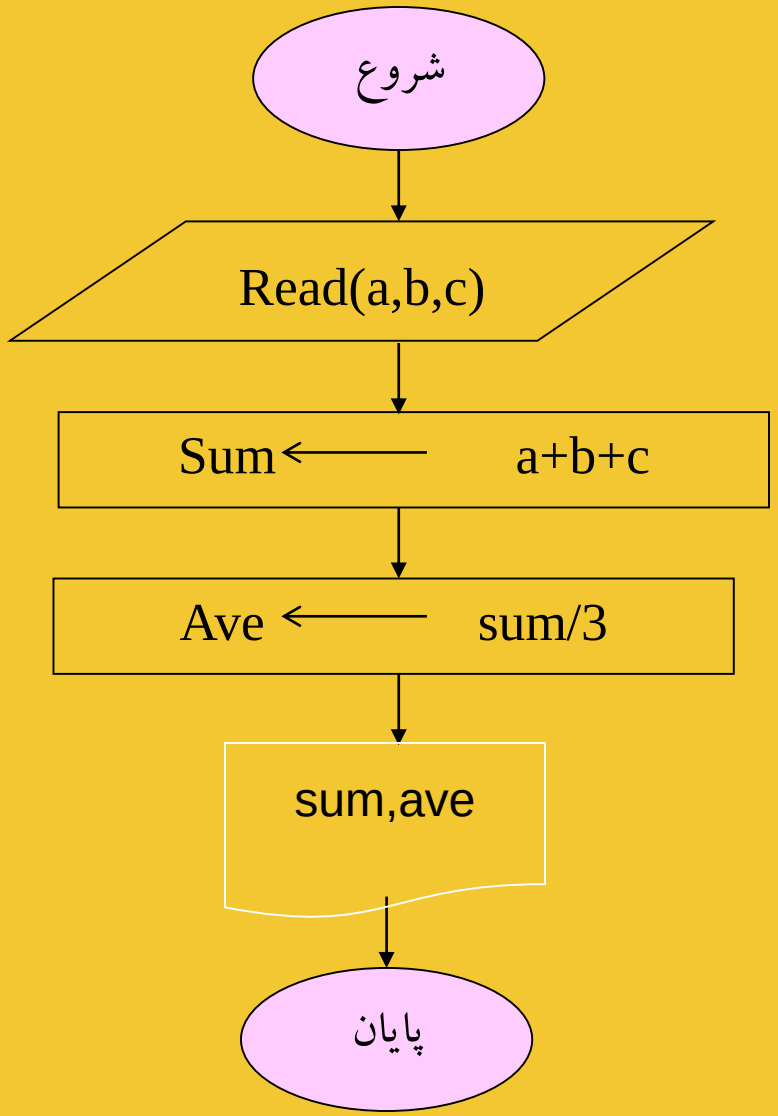
$$C_n^m = A/(B \times C)$$

فلوچارت(Flowchart) یا نمودار گردشی

- فلوچارت، بیان تصویری الگوریتم با کمک مجموعه ای استاندارد از اشکال ساده می باشد
- فلوچارت یکی از روشهای برقراری ارتباط منطقی بین مراحل مختلف حل مساله است.
- شلکهای استاندارد:



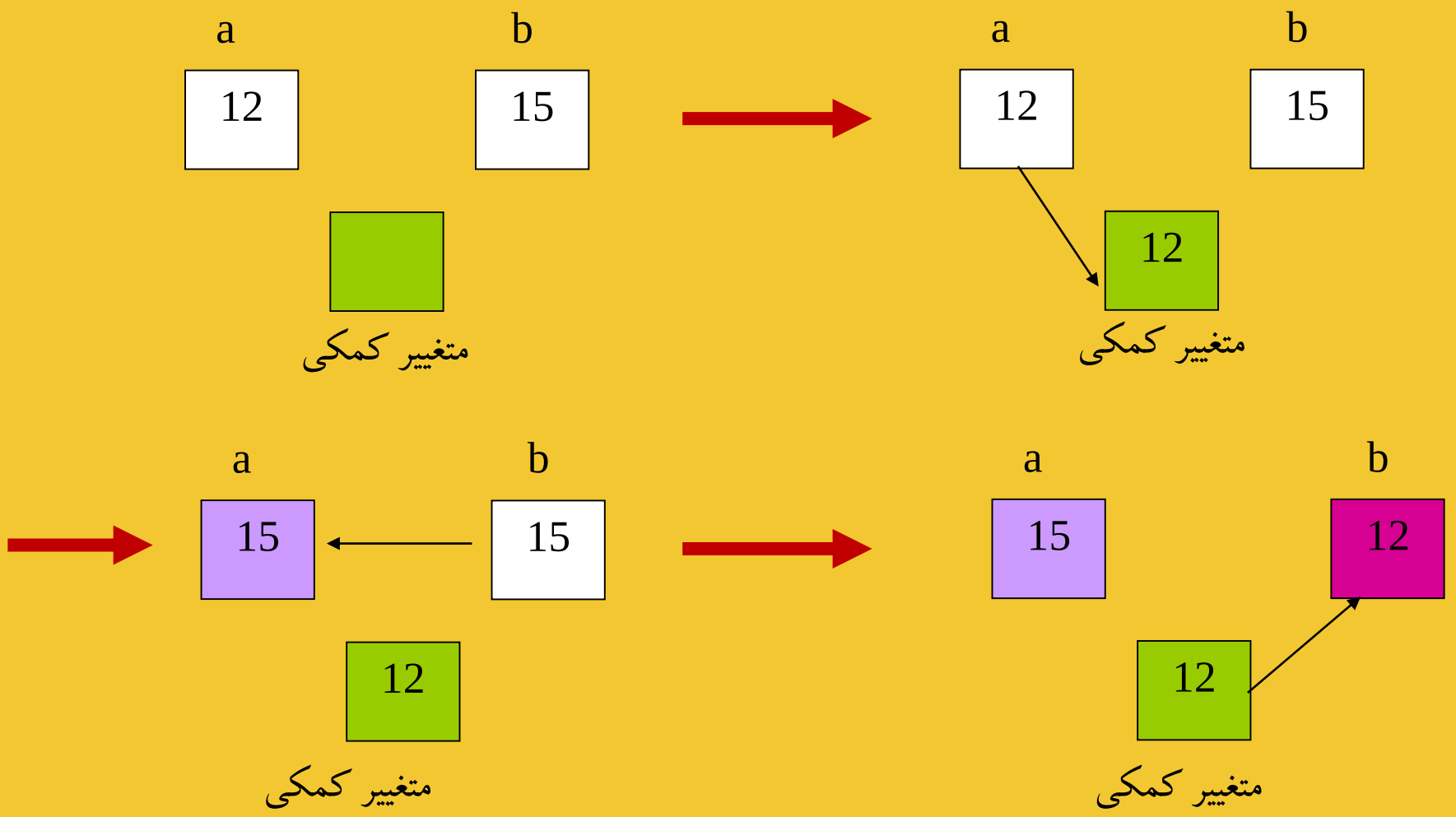
فلوچارت محاسبه مجموع و میانگین سه عدد



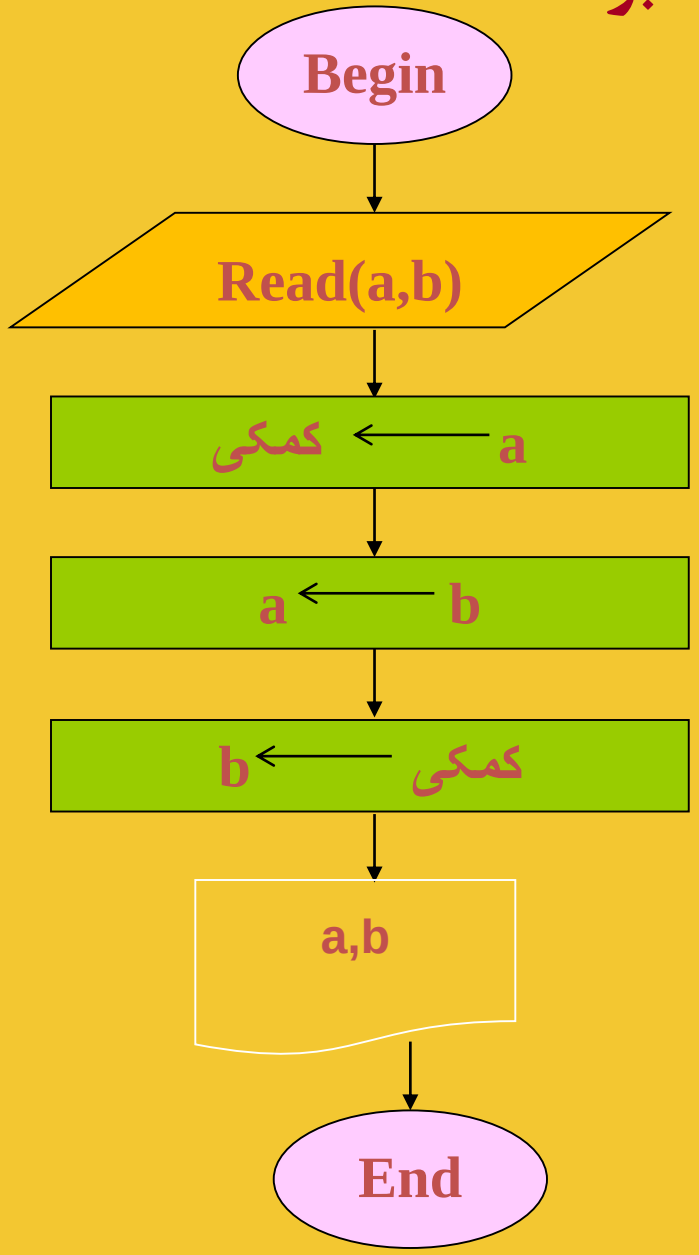
مثال

- مثال: فلوچارتی رسم نمائید که دو عدد از ورودی دریافت کرده سپس محتویات دو عدد را با هم جابجا نماید.

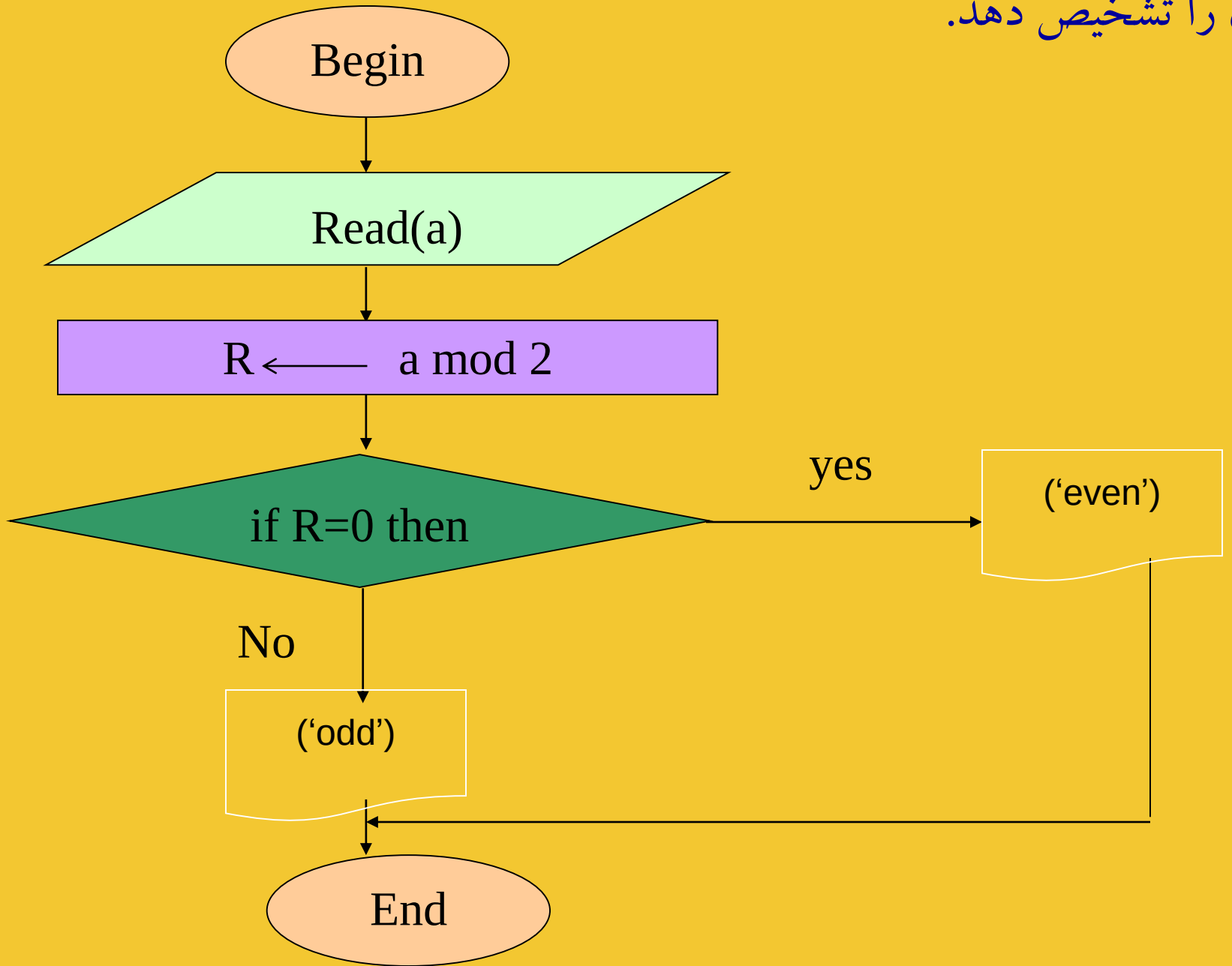
- برای حل این مسئله a , b را دو متغیر در نظر می‌گیریم. سپس با استفاده از یک متغیر کمکی محتویات این دو عدد را جابجا می‌کنیم:


a
b
12
15
متغییر کمکی
a
b
12
15
12
متغییر کمکی
a
b
15
15
12
متغییر کمکی
a
b
15
12
12
متغییر کمکی

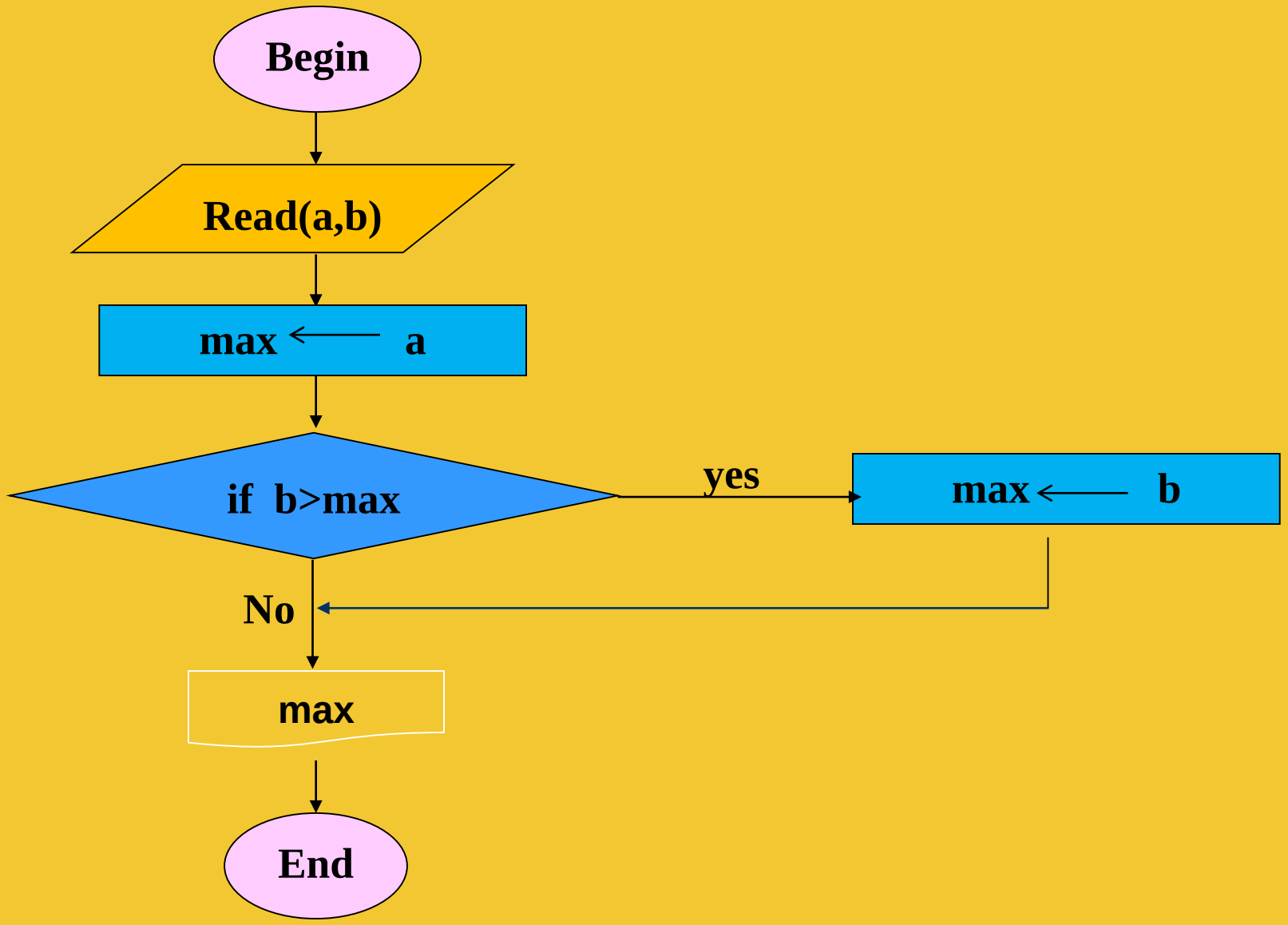
فلوچارت مسئله بالا بصورت زیر خواهد بود:



مثال : فلوچارتی رسم نمائید که عددی را از ورودی دریافت کرده، فرد یا زوج بودن آن را تشخیص دهد.



مثال : فلوچارتی رسم کنید که دو عدد از ورودی دریافت کـرده بزرگترین عدد را پیدا کرده در خروجی چاپ نماید.



مثال : فلوچارتی رسم نمائید که سه عدد از ورودی دریافت کرده، کوچکترین عدد را یافته در خروجی چاپ نماید: