UOA141 مبادئ علوم الحاسبات

المستوى الأول

الجزء الرابع

What is a Programming Language?

A set of rules that provides a way of telling a computer what operations to perform

لغة البرمجة : Programming language عبارة عن مجموعة من الأوامر، تكتب وفق مجموعة من القواعد تحدد بواسطة لغة البرمجة، ومن ثم تمر هذه الأوامر بعدة مراحل إلى ان تنفذ على جهاز الحاسوب.

تقسم لغات البرمجة بناء على قربها من اللغات الإنسانية إلى High-level (قريبة من اللغة التي يفهمها البشر) مثل C, C++, Java ..etc. و Low-level program (كلغة الأسيمبلي Assembly وهي قريبة من لغة الآلة).

1- لغة الآلة :

وتسمى *اللغة الثنائية* حيث إنها تتكون من سلسلة من ٠ و ١، وهي اللغة الوحيد التي يفهمها الحاسب الآلي، حيث تحول جميع اللغات إلى لغة الآلة، حتى تتمكن معدات الحاسب الآلي من التفاهم معها، ولأنها تتكون من صفر وواحد، لذا فقد تميزت هذه اللغة بالصعوبة نظراً لما تتطلبه من حفظ ودقة في كتابة سلسلة طويلة من صفر وواحد بترتيب معين، مما ينتج عنه أخطاء كثيرة من الترميز، ويجب أن يحدد المبرمج كل شي.

2- لغة التجميع :

ظهرت لغة التجميع بوصفها أو لغة ترميز، تستخدم الرموز SYMBOLIC CODE للتعبير عن تعليمات لغة الآلة، وذلك لمواجهة صعوبة الترميز بلغة الآلة، ولغة التجميع لغة قريبة من لغة الآلة التي يفهمها الحاسب الآلي، وتسمى هذه اللغات بلغات المستوى البسيط .

ويتم استعمال مختصرات ورموز يسهل حفظها وكتابتها لكل تعليمة من تعليمات لغة الآلة، ولغة التجميع كما في لغة الآلة مصممة للعمل على حاسب معين، مما يوفر قدرة أكبر على استغلال موارد الحاسب الآلي ووحدة المعالجة المركزية بشكل أفضل، ويقوم البرنامج المسمى المجمع ASSEMBLERبترجمة البرنامج المكتوب بلغة المجمع إلى لغة الآلة .

3- اللغات العليا :

سميت بهذا الاسم لأنه أصبح بإمكان المبرمج كتابة البرنامج دون معرفة تفاصيل كيفية قيام الحاسب بهذه العمليات، كمواقع التخزين وتفاصيل الجهاز الدقيقة، وتعبيرات لغات المستوى العالي هي تعبيرات شبيهة إلى درجة كبيرة باللغة الطبيعية التي يستخدمها الإنسان في حياته للتواصل، والتخاطب مع الآخرين . ومن مميزات اللغات العليا التي تميزها من لغات المستوى البسيط، بالإضافة إلى ما سبق، أن هذه اللغات غير مرتبطة بجهاز معين . أي يمكننا تنفيذ البرنامج المكتوب بلغة من لغات المستوى العالي، كالفورتران أو الكوبول أو البيسك على أكثر من جهاز، كما يمكن استخدام أكثر من لغة ترجمة على حاسب معين . كذلك، فإن اكتشاف الأخطاء وتصحيحها أصبح أكثر سهولة بسبب سهولة قراءة البرامج وتتبعها وفهمها .

Levels of Programming Languages

High-level program

```
class Triangle {
  ...
  float surface()
    return b*h/2;
  }
```

Low-level program

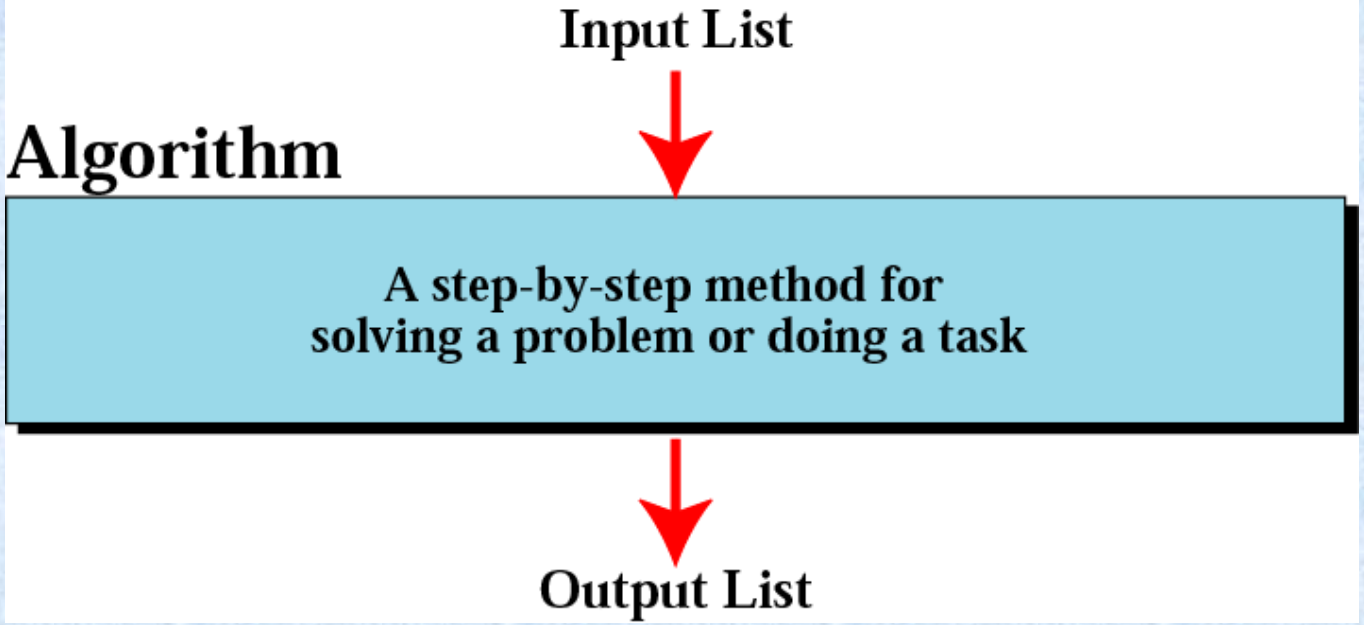
```
LOAD r1,b
LOAD r2,h
MUL r1,r2
DIV r1,#2
RET
```

Executable Machine code

```
000100100100010100100 1
00111011001010110100 1.
..
```

ما المقصود بالخوارزميات ؟

الخوارزمية هي مجموعة من الخطوات الرياضية والمنطقية والمتسلسلة اللازمة لحل مشكلة ما. وسميت الخوارزمية بهذا الاسم نسبة إلى العالم المسلم أبو جعفر محمد بن موسى الخوارزمي الذي ابتكرها في القرن التاسع الميلادي. الكلمة المنتشرة في اللغات اللاتينية والأوروبية هي algorithm



تراكيب الخوارزميات : التسلسل والاختيار والتكرار.

التسلسل: تكون الخوارزمية عبارة عن مجموعة من التعليمات المتسلسلة، .. على سبيل المثال بفرش قطعة موكيت في أحد الساحات ، تحتاج للقيام بمجموعة من الخطوات .

الاختيار: بعض المشاكل لا يمكن حلها بتسلسل بسيط للتعليمات، وقد تحتاج إلى اختبار بعض الشروط وتنظر إلى نتيجة الاختبار، إذا كانت النتيجة صحيحة تتبع مسار يحوي تعليمات متسلسلة، وإذا كانت خاطئة تتبع مسار آخر مختلف من التعليمات. هذه الطريقة هي ما تسمى اتخاذ القرار أو الاختيار.

التكرار: عند حل بعض المشاكل لا بد من إعادة نفس تسلسل الخطوات عدد من المرات. وهذا ما يطلق عليه التكرار.. لملأ غرف البيت بالكراسي تحتاج إلى تكرار عملية الدخول للغرف الواحدة تلو الأخرى و تكرار عملية صف الكراسي

Ex1:

Write algorithm to : Find the average marks of three courses.

١.تحديد عناصر المسألة

أ- **المخرجات :** متوسط الدرجات ،ونرمز له بـ (Av)

ب- **المدخلات :** الدرجات الثلاث ونرمز لهم بـ (M1, M2, M3)

ت- **العمليات :** قانون متوسط الدرجات Av=(M1+M2+M3)/3

1- Start
2- Read m1,m2,m3
3- Av=(m1+m2+m3)/3
4- Print Av
5- End

Ex2:

Write algorithm : Find the average marks of three courses. If the average student scores greater than 90 degrees Print message "Excellent“ with the average .

١.تحديد عناصر المسألة

أ- **المخرجات :** متوسط الدرجات ،ونرمز له بـ (Av)

ب- **المدخلات :** الدرجات الثلاث ونرمز لهم بـ (M1, M2, M3)

ت- **العمليات :** قانون متوسط الدرجات Av=(M1+M2+M3)/3

```
1- Start
2-Read m1,m2,m3
3- Av=(m1+m2+m3)/3
4- if (av>=90) then print  Av, “Excellent”
     else  print Av
5- End
```

Ex3 :

Write algorithm : Print the words: (**Pass**) when the student success . And print the words: (**Failed**) when the student failure.

١.تحديد عناصر المسألة

أ- **المخرجات :** طباعة عبارة "Pass" أو "Failed"

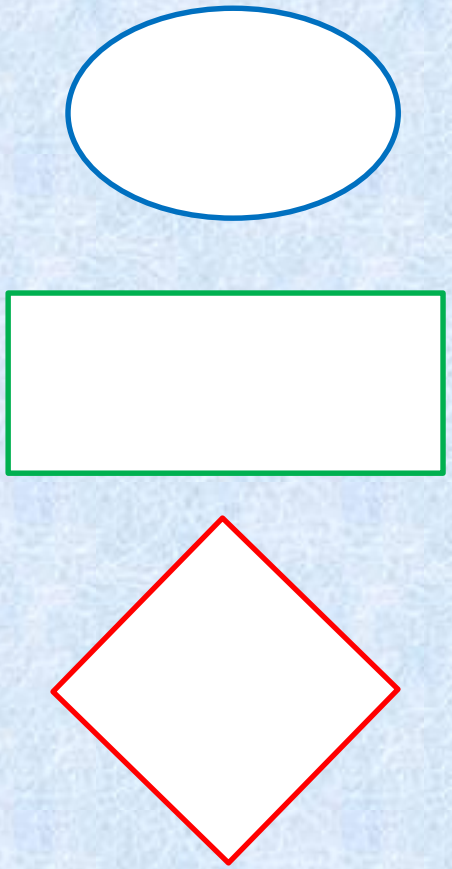
ب- **المدخلات : معدل** درجات للطالب ونرمز له بالرمز (Av)

١. **العمليات :** اذا كان مجموع الدرجات A>=50 اطبع Pass

وإذا كان مجموع الدرجات A<50 اطبع Failed

1- Start
2- Read Av
3- if Av >= 50 then Print "Pass"
else Print "Failed"
4- End

المخطط التدفقي (أو المخطط الانسيابي Flowchart) :



لتحديد بداية الخوارزمية ونهايتها :

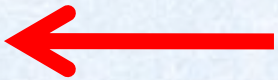
عمليات المعالجة:

العمليات التي ترتبط باختبار تحقق شرط ما وتتطلب قرارًا منطقيا :

المخطط التدفقي (أو المخطط الانسيابي Flowchart) :

عمليات الإدخال والإخراج :

عمليات الربط في حال تعدد الصفحات :



اتجاه تنفيذ الخوارزمية :

مثـال /اكتب الخوارزمية الكلامية والرمزية والمخطط التدفقي لإيجاد مساحة ومحيط المستطيل ؟

❖ الخوارزمية الكلامية :

المدخلات :

الطول والعرض

المعالجة :

المساحة (s)=الطول * العرض

المحيط (m) =(الطول+ العرض)*٢

المخرجات :

المساحة والمحيط

❖ الخوارزمية الرمزية :

المدخلات :

X و y

المعالجة :

$(s) = x * y$

$(m) = (x + y) * ٢$

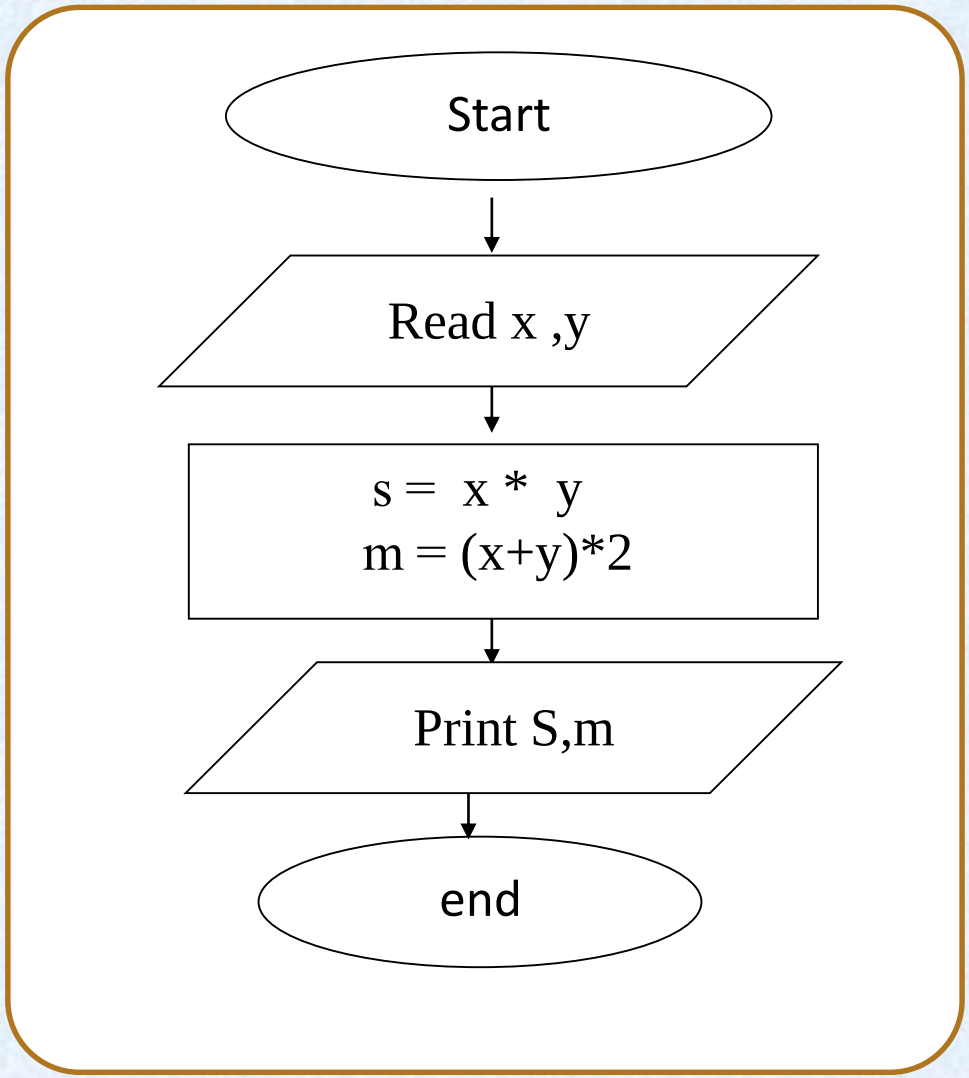
المخرجات :

M , S

```
Algorithm :

1- Start
2- Read x ,y
3- S=x*y
4- M=(x+y)*2
5- Print M,s
6- End
```

المخطط الانسيابي /Flowchart



المثال الثاني

اكتب الخوارزمية الرمزية والمخطط التدفقي لإدخال x(متغير) وإيجاد قيمة / y= (x-2)/x

❖ الخوارزمية الرمزية :

المدخلات :

x

المعالجة :

إذا كانت (x = 0) عندئذ "

أعد إدخال قيمة x من جديد لأنه ل

ا يمكن القسمة على 0 "

وإلا فاحسب : y = (x – 2) / x

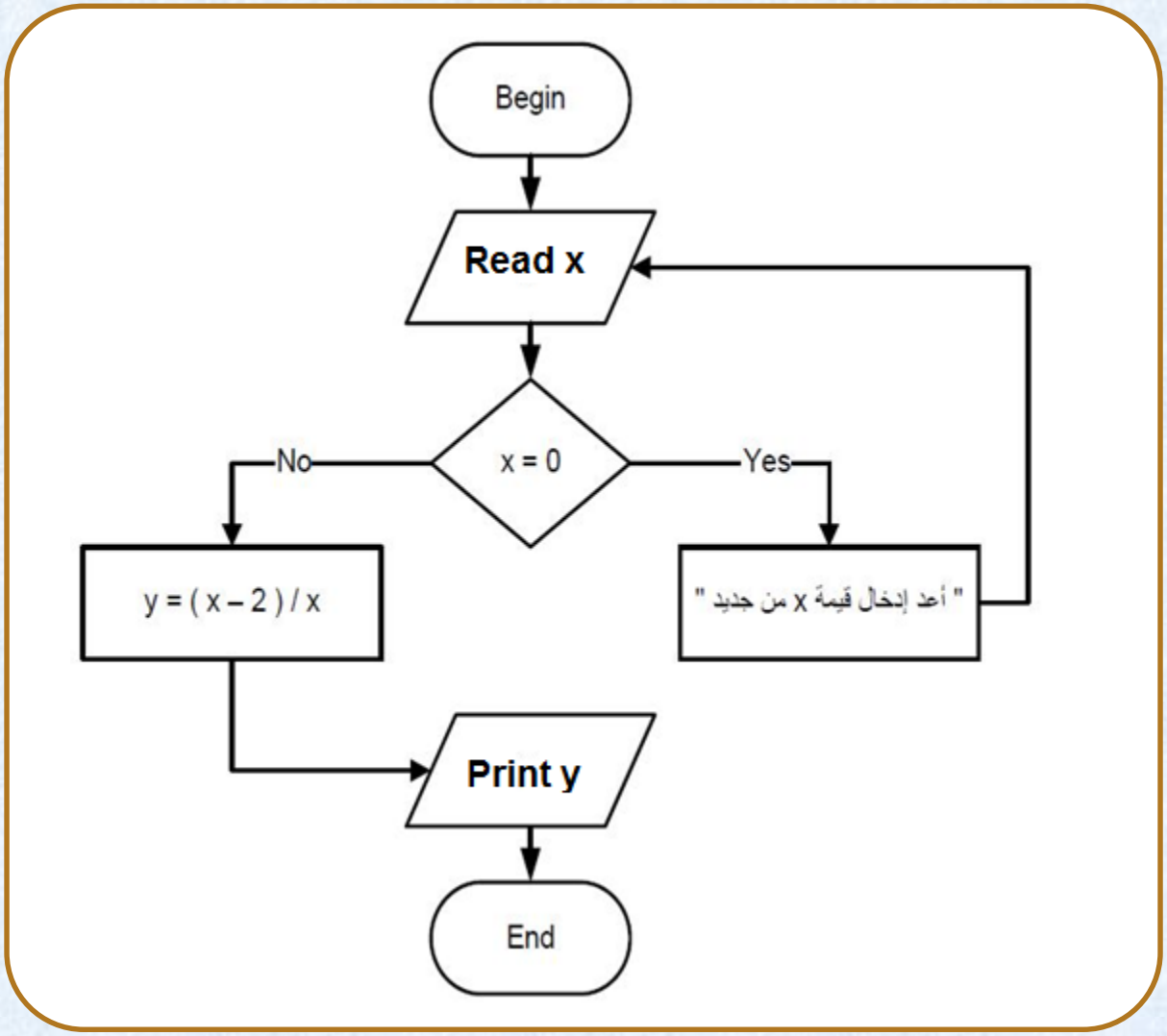
المخرجات :

y

Algorithm :

```
1- Start
2- Read x
3- if x= 0 then goto  2
4- y=(x-2)/x
5- Print  y
6- End
```

المخطط الانسيابي /Flowchart



المثال الرابع

اكتب الخوارزمية الرمزية والمخطط التدفقي لحل المعادلة $a X + b = 0$

مناقشا جميع الحالات الممكنة لـ a,b

❖ الخوارزمية الرمزية :

المدخلات :

a , b

المعالجة :

إذا كانت (a=0 , b<>0) اطبع : " مستحيلة الحل"

وإلا إذا كانت (a=0 , b=0) اطبع : " حالة متطابقة "

وإلا (a<>0) نجد : x = -b/a

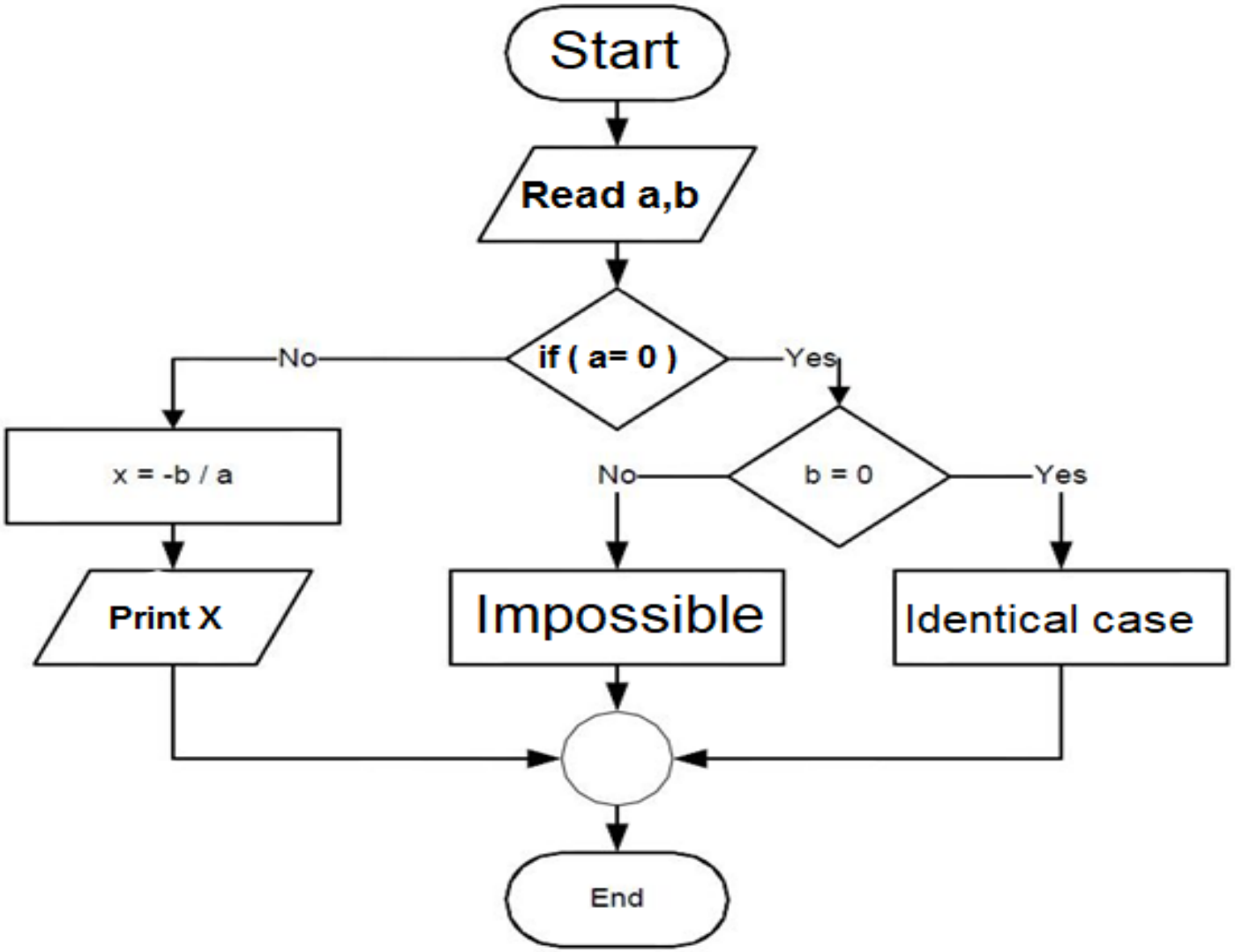
المخرجات :

x

```
Algorithm :

1- start
2- Read a,b
3- if (a=0) & (b <> 0 )
          then print " Impossible "
4- if (a=0) & (b=0)
          then print " Identical case "
5- if (a<>0)
          then  x=(-b/a)
6 -print x
7- End
```

المخطط الانسيابي /Flowchart



تمارين /

المثال الخامس

اكتب الخوارزمية الرمزية والمخطط التدفقي(الانسيابي)
لإيجاد قيمة y المعطاة بالشكل التالي :

$$y=\begin{cases} 2/(x-2) & x>2 \\ -4/(5-x) & x<=-2 \end{cases}$$

حل المثال /

❖ الخوارزمية الرمزية :

المدخلات :

x

المعالجة :

إذا كانت (x > 2) عندئذ : y = 2 / (x – 2)

وإلا إذا كانت (x <= -2) عندئذ : y = -4 / (5 – x)

وإلا أعد إدخال x

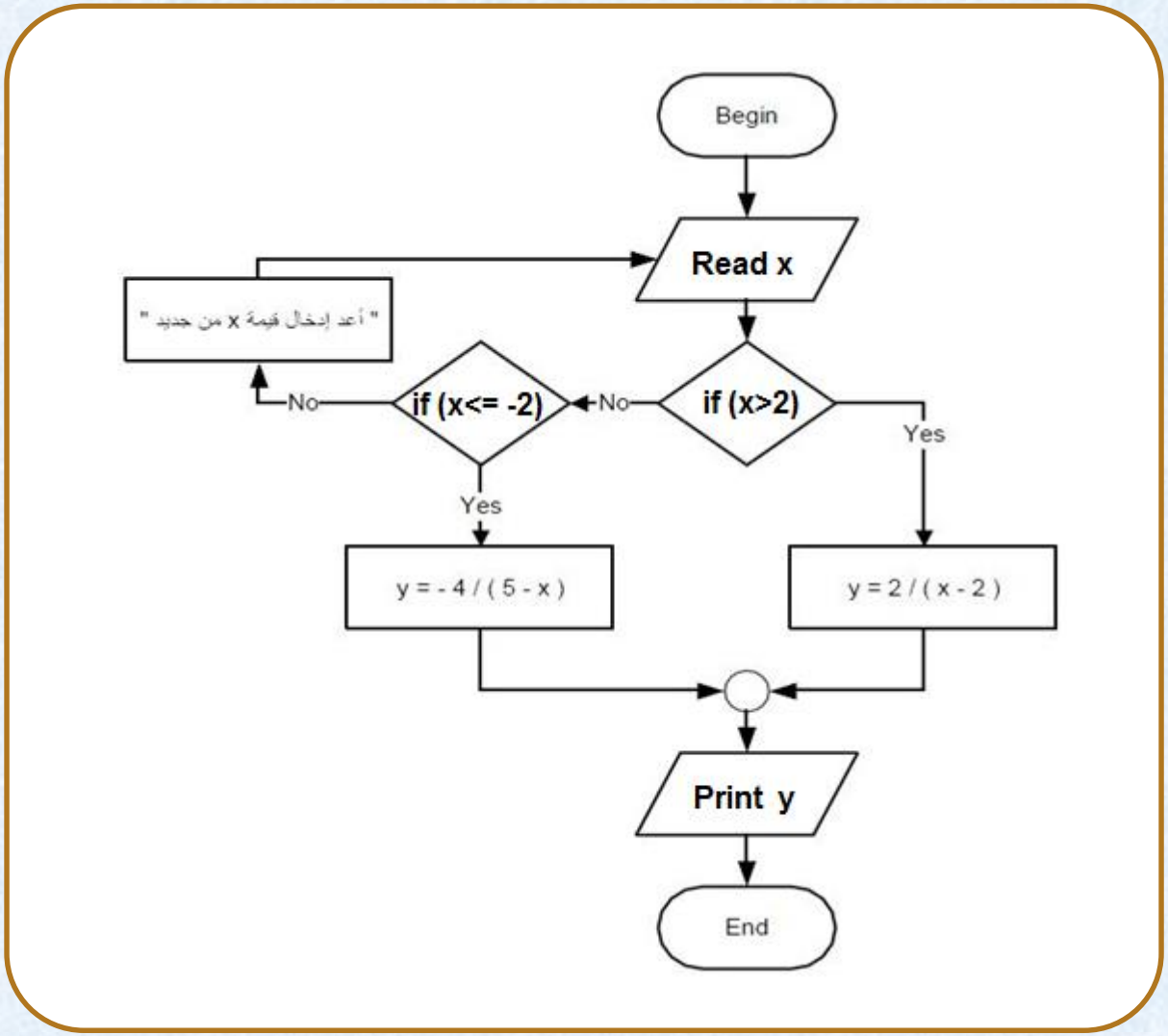
المخرجات :

y

Algorithm :

```
1- start
2- Read x
3-If (x>2) then
            y= 2/(x-2)
4- if (x<= -2) then
            y= -4/(5-x)
    Else goto 2
5- print y
6- end
```

المخطط الانسيابي /Flowchart



تمارين /

المثال الخامس

اكتب الخوارزمية الرمزية والمخطط التدفقي لإدخال عدد صحيح (x) موجب وطباعة إذا كان فرديا أم زوجيا ؟

حل المثال /

❖ الخوارزمية الرمزية :

المدخلات :

x

المعالجة والمخرجات :

إذا كان باقي قسمة العدد على 2 يساوي صفر

(x mode 2 = 0) فإن

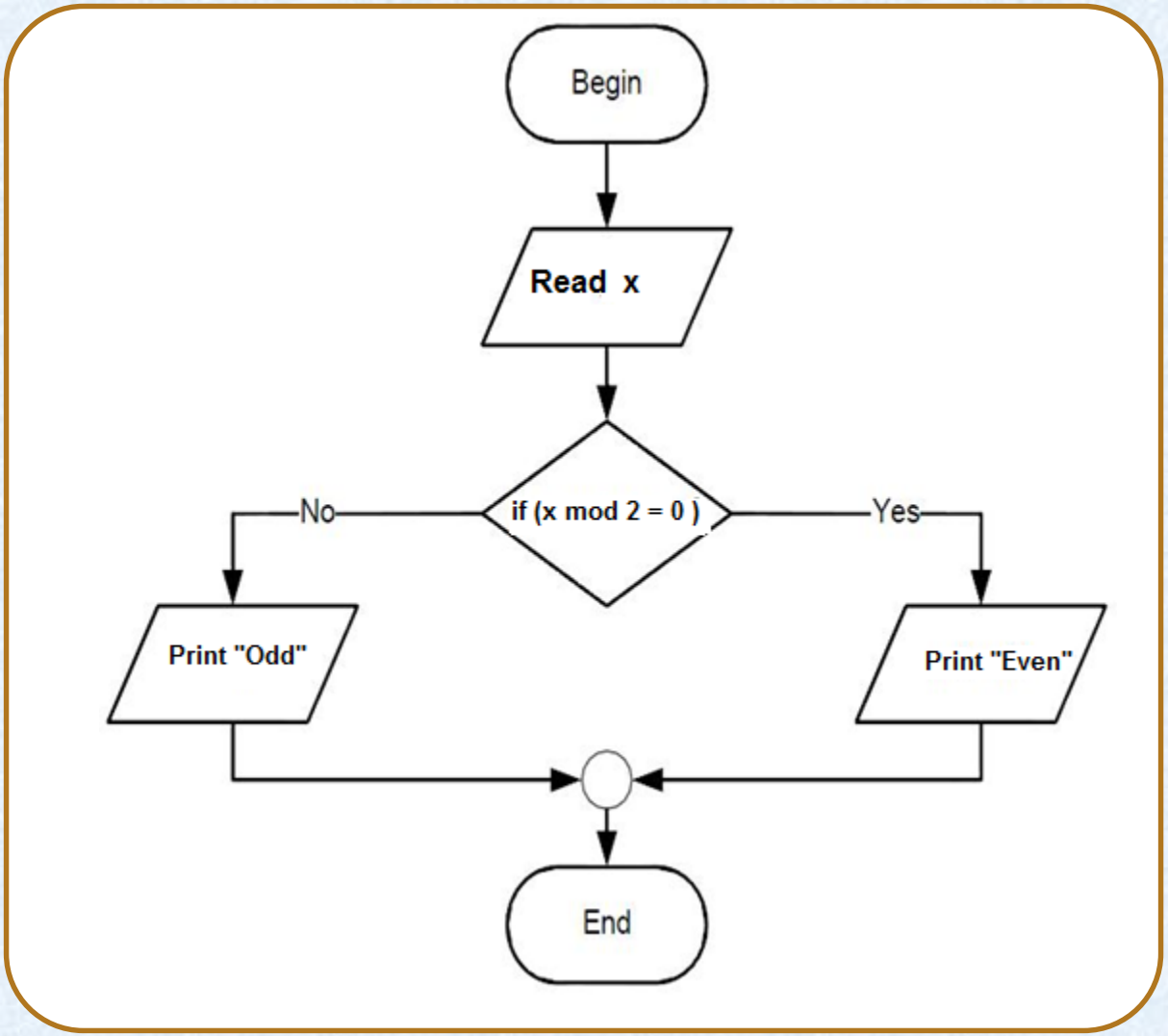
اطبع : " العدد زوجياً Even

وإلا اطبع : " العدد فردي odd "

```
Algorithm :

1- Start
2- Read x
3- if (x mod 2=0  ) then print "Even"
                        Else print "Odd "
4- End
```

المخطط الانسيابي /Flowchart



خرائط سير البرنامج Programs Flowchart

ويستعمل هذا النوع من الخرائط لبيان الخطوات الرئيسية التي توضع لحل مسألة ما وذلك بشكل رسوم اصطلاحية تبين العلاقات المنطقية بين سائر خطوات الحل وموقع ووظيفة كل منها في إطار الحل الشامل للمسألة، ويكن تصنيف خرائط سير البرنامج إلى أربعة انواع رئيسية:

أ- خرائط التتابع البسيط (Simple sequential Flowchart).
ب- الخرائط ذات الفروع (Branched Flowchart).
ج- خرائط الدوران الواحد (simple-Loop Flowchart).
د- خرائط الدورانات المتعددة (Multi-Loop Flowchart).
ويمكن للبرنامج الواحد أن يشمل أكثر من نوع واحد من هذه الأنواع .

أ- خرائط التتابع البسيط (Simple sequential Flowchart):

يتم ترتيب خطوات الحل لهذا النوع من الخرائط بشكل سلسلة مستقيمة من بداية البرنامج حتى نهايته ولا تحتوي على أية تفرعات أو دورانات .

المتغيرات

نستطيع ان نعرف المتغيرات بمنظورين، بالمنظور الرياضي يعرف المتغير على انه مجهول س يحتوي على قيمة معينة، اما بالمنظور البرمجي- وهو الاهم - يعرف المتغير على انه قيمة تحفظ في ذاكرة الجهاز . وتختلف المساحة المحجوزة لحفظ هذه القيمة باختلاف نوع المتغير، فمتغير من النوع Byte لا يستهلك سوى بايت واحد من ذاكرة الحاسب، في حين أن متغير من نوع String قد يحجز مساحة تصل الى 2 جيجابايت.

مثال: اكتب الخوارزمية اللازمة لحساب مساحة ومحيط الدائرة بمعلومية نصف قطرها ، ثم ارسم خريطة سير البرنامج Flowchart المناظرة له؟

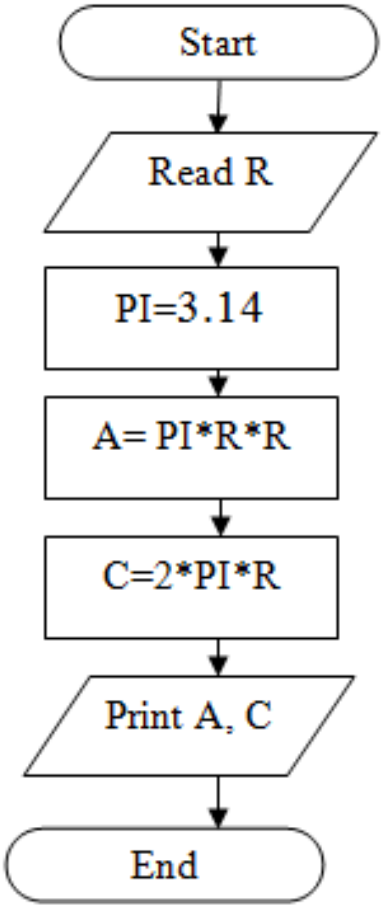
Write an Algorithm and Draw a Flowchart to calculate circumference and area of circle and display result?

الحل:

مساحة الدائرة = ΛR^2

محيط الدائرة = $2\Lambda R$

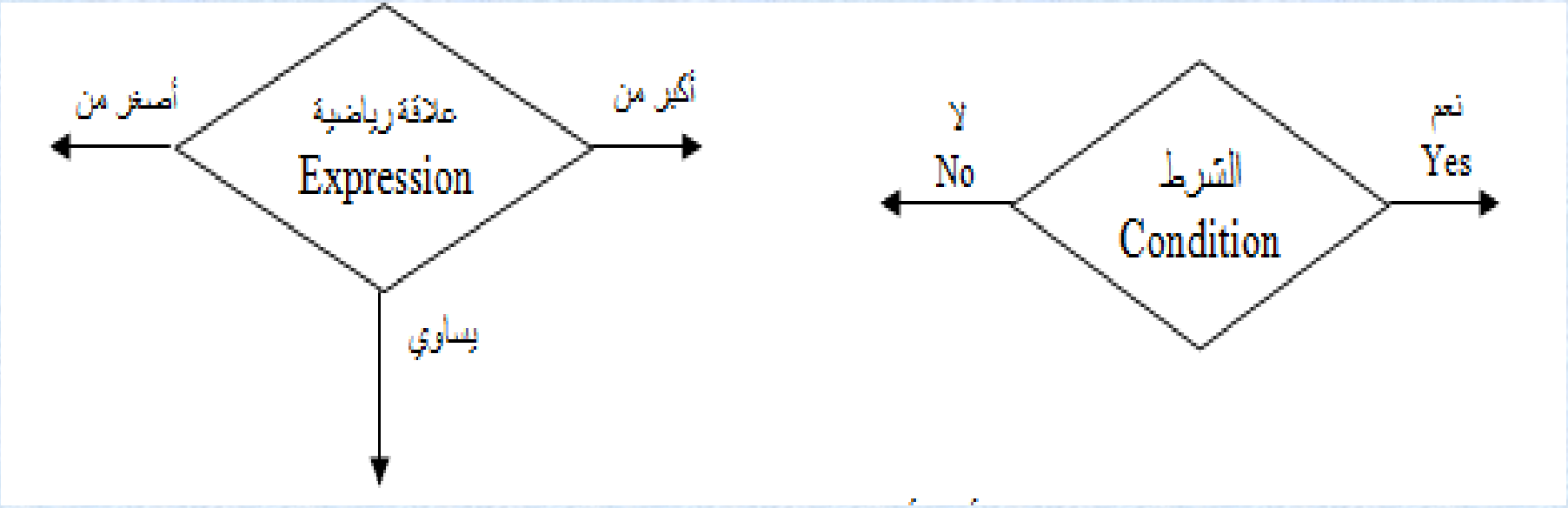
حيث Λ = النسبة التقريبية وقيمتها العددية ثابتة وتساوي 3.14، R نصف القطر وهو متغير

خريطة سير البرنامج Flowchart	الخوارزمية Algorithm	
	1- Start 2- Read R 3- PI=3.14 4- A= PI*R*R 5- C=2*PI*R 6- Print A, C 7- End	١ – ابدأ ٢ – اقرأ قيمة R ٣ – ضع قيمة PI= 3.14 ٤ – احسب مساحة الدائرة A من المعادلة : A= PI*R*R ٥ – احسب محيط الدائرة C من المعادلة: C=2*PI*R ٦ – اطبع قيم كل من A,C ٧ – توقف

ب- الخرائط ذات الفروع (Branched Flowchart):

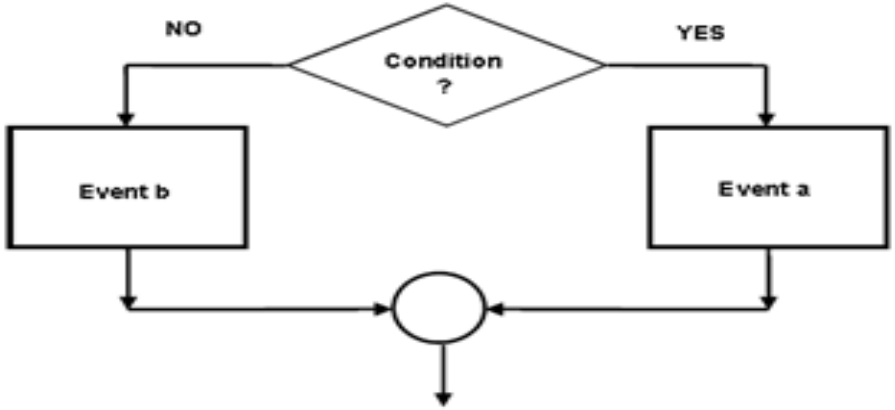
يحدث التفرع داخل البرنامج عند الحاجة لاتخاذ قرار أو مفاضلة بين اختيارين أو أكثر فيسير كل اختيار في طريق مستقل (تفرع) عن الآخر. وهناك نوعان من القرارات يمكن للمبرمج استعمال أحدها حسب المسألة المراد حلها.

أ) قرار ذو تفرعين ب) قرار ذو ثلاثة تفرعات

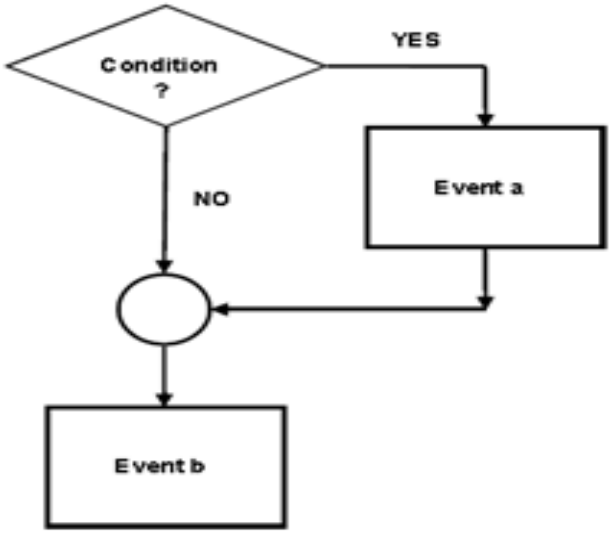


وبشكل عام فإن خرائط التفرع يمكن أن تأخذ إحدى الصورتين التاليتين:

شكل (أ)



شكل (ب)



- **في شكل(أ) يبين أنه إذا كان جواب الشرط "نعم" فإن الحدث التالي في التنفيذ هو الحدث** (a) **، أما إذا كان جواب الشرط "لا" فإن الحدث التالي في التنفيذ هو الحدث** (b).

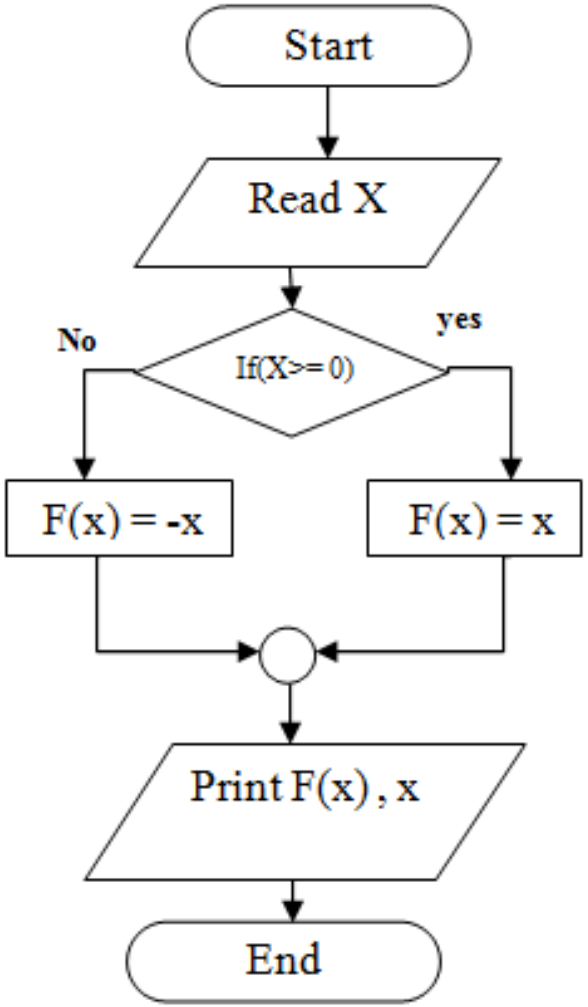
- **في شكل (ب) يبين أنه إذا كان جواب الشرط "نعم" فإن الحدث التالي في التنفيذ هو الحدث** (a) **يليه الحدث** (b) **،أما إذا كان جواب الشرط "لا" فإن الحدث التالي في التنفيذ هو الحدث** (b) **مباشرة.**

مثال: يوضح القرار ذو التفرعين

اكتب الخوارزمية اللازمة لحساب قيمة الدالة F(x) المعرفه كما يلي

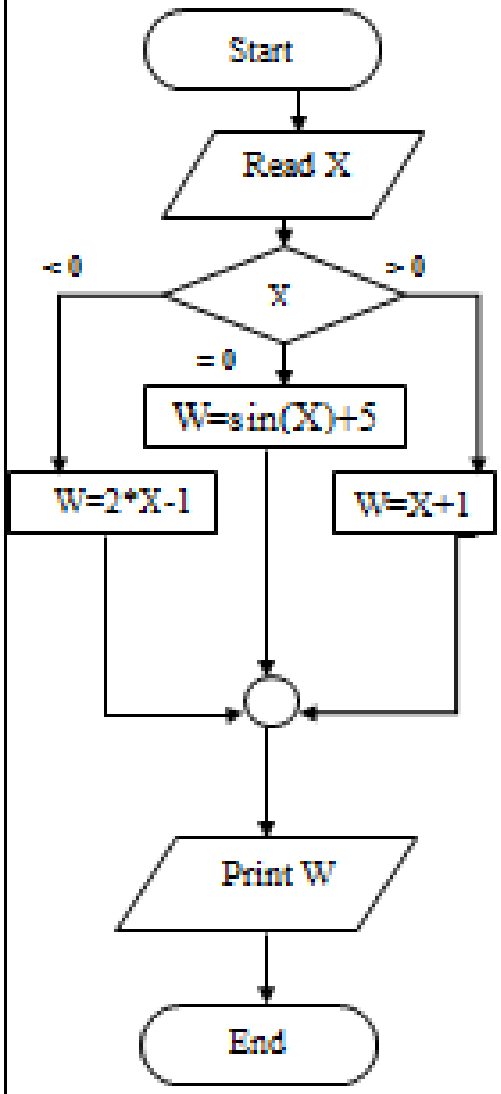
$$F(x)=\begin{cases} x\ , x\geq 0 \\ -x\ , x<0 \end{cases}$$

، ثم ارسم خريطة المسار المناظرة له.

Flowchart خريطة سير البرنامج	Algorithm الخوارزمية	
	1- Start 2- Read x 3- If x >= Zero then go to step (4), else go to step (5). 4-F(x) = x ,and then go to step (6) 5-F(x) = -x 6- print F(x) , x 7- End.	١ – ابدأ ٢ – اقرأ قيمة X ٣ – إذا كانت X أكبر من أو تساوي صفر اذهب إلى الخطوة(٤). وإلا فاذهب إلى الخطو (٥). ٤ – احسب قيمة الدالة F(x) من F(x) = x ثم اذهب إلى الخطوة (٦). ٥–احسب قيمة الدالة F(x) من F(x) = -x . ٦ – اطبع قيمة x, F(x) ٧ – توقف.

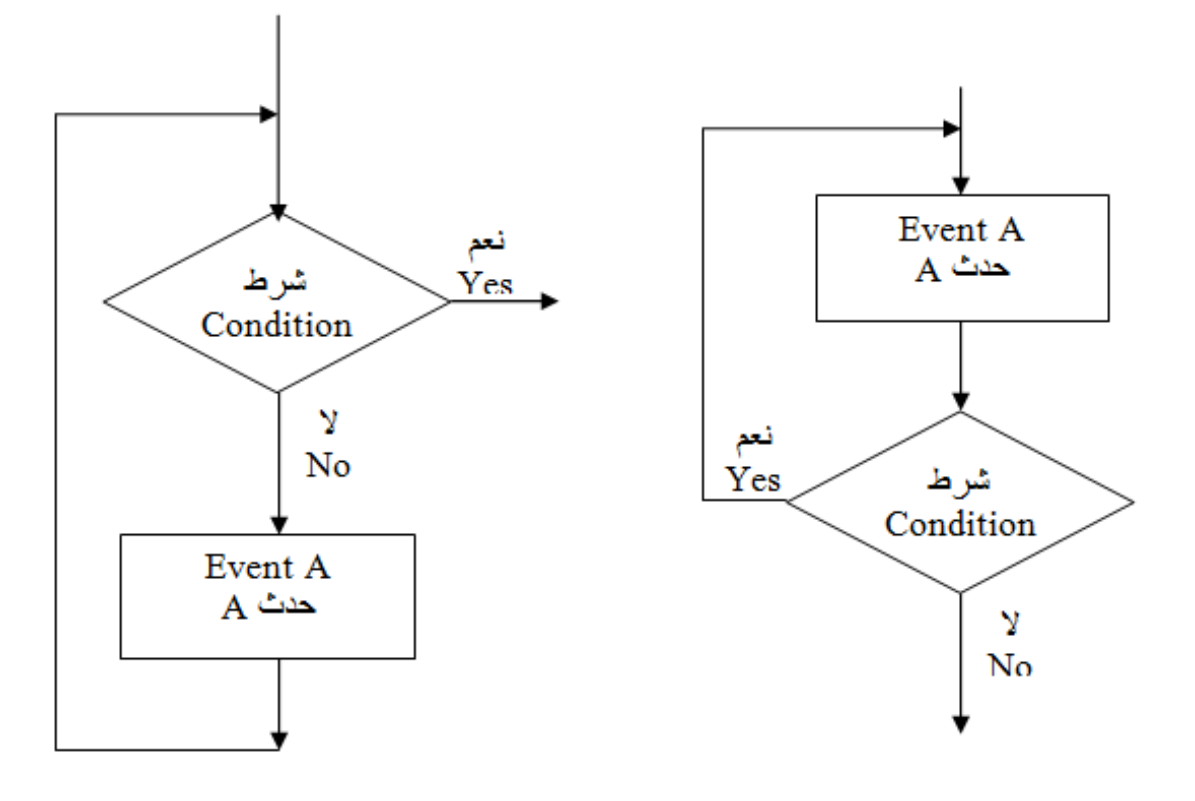
مثال: يوضح القرار ذو ثلاثة تفرعات.
اكتب الخوارزمية و ارسم خريطة سير العمليات لحساب قيمة W المعطاة من العلاقة التالية

$$W = \begin{cases} X + 1 , X > 0 \\ \sin(X) + 5, X = 0 \\ 2X - 1 , X < 0 \end{cases}$$

Flowchart خريطة سير البرنامج	Algorithmالخوارزمية	
	1- Start 2- Read x 3- If x > Zero then go to step (4), If x = Zero then go to step (5), If x < Zero then go to step (6). 4-W =X+1,and then go to step (7) 5-W=sin(X)+5,and then go to step (7) 6- W =2*X-1,and then go to step (7) 7- print W 8- End.	١ – ابدأ ٢ – اقرأ قيمة X ٣ – إذا كانت X أكبر من صفر اذهب إلى الخطوة(٤). وإذاكانت X تساوي صفر فاذهب إلى الخطو (٥) أما إذا كانت X أقل من صفر إذهب إلى خطوة (٦). ٤ – احسب قيمة W من المعادلة: W =X+1 ثم اذهب إلى خطوة(٧). ٥ – احسب قيمة W من المعادلة: W =sin(X)+5 ، ثم اذهب إلى خطوة(٧). ٦ – احسب قيمة W من المعادلة: W =2*X-1، ثم اذهب إلى خطوة(٧). ٧ – اطبع قيمة W ٨ – توقف.

ج- خرائط الدوران الواحد (simple-Loop Flowchart):

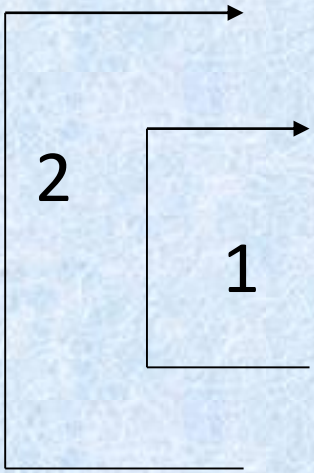
نحتاج هذا النوع من الخرائط لإعادة عملية أو مجموعة من العمليات في البرنامج عددا محدودا أو غير محدود من المرات ويكون الشكل العام لمثل هذه الخرائط كما يلي:



وقد سميت هذه الخرائط بخرائط الدوران الواحد لأنها تستعمل حلقة واحدة للدوران ، وتسمى أحيانا بخرائط الدوران البسيط.

د- خرائط الدورانات المتعددة (Multi-Loop Flowchart):

في هذه الحالة تكون الدورانات داخل بعضها البعض بحيث لا تتقاطع ، فإذا كان لدينا دورانان من هذا النوع كما في الشكل التالي فيسمى الدوران رقم (١) دورانا داخليا Inner Loop بينما الدوران رقم (٢) دورانا خارجيا outer Loop ، ويتم التنسيق بين مثل هذين الدورانين ، بحيث تكون أولوية التنفيذ للدوران الداخلي.

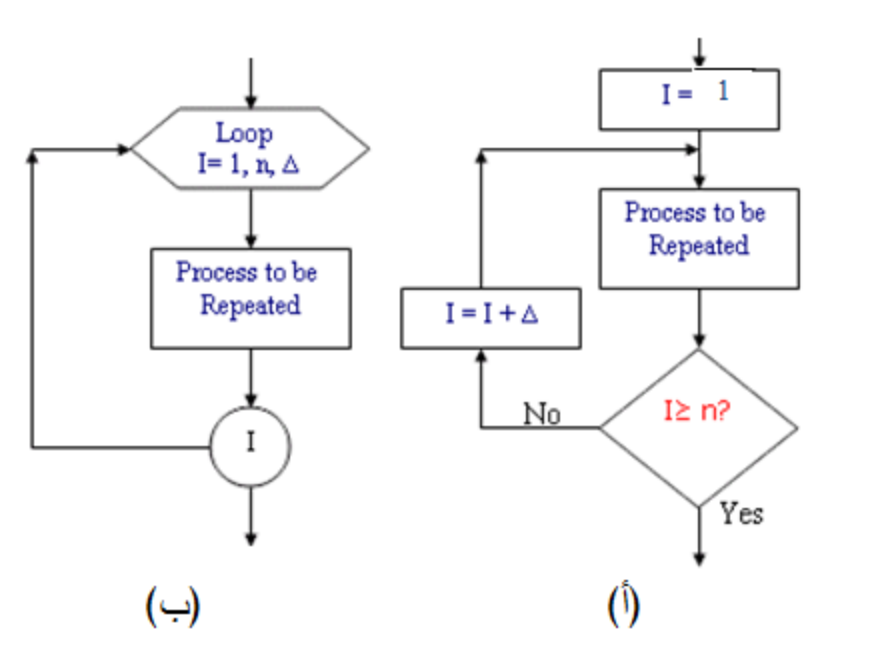


وقد سميت هذه الخرائط بخرائط الدورانات المتعددة لأنها تستعمل أكثر من حلقة دوران واحدة، وقد تسمى أحيانا بخرائط الدورانات المتداخلة أو المتراكبة أو الضمنية .

صيغة الدوران باستعمال الشكل الاصطلاحي (الدوران التكراري Loop):

في الفقرتين السابقتين تعلمنا مفهوم الدوران البسيط والدورانات المتعددة(المتداخلة) ويمكننا الآن استخدم الشكل الاصطلاحي (الدوران التكراري Loop) .
نلاحظ من المثال السابق أننا نحتاج إلى العناصر الآتية:

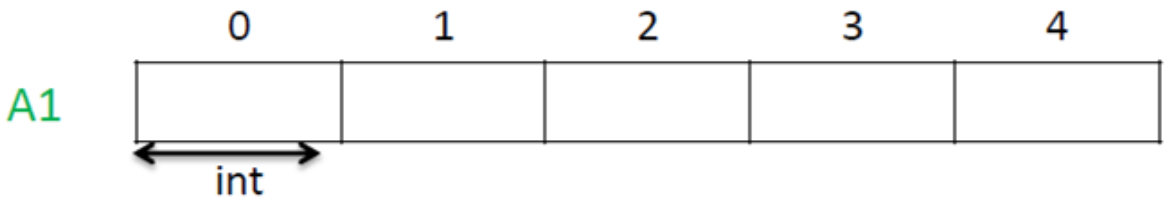
* القيمة الأولية للعداد I (هنا I=1).
* القيمة النهائية للعداد I (هنا I=N).
* قيمة الزيادة عند نهاية كل دورة Δ .



Arrays

An **array** is a series of elements of the same type placed in contiguous memory locations that can be individually referenced by adding an index to a unique identifier. That means that, for example, we can store 5 values of type int in an array without having to declare 5 different variables, each one with a different identifier. Instead of that, using an array we can store 5 different values of the same type, int for example, with a unique identifier.

For example, an array to contain 5 integer values of type int called A1 could be represented like this:



where each blank panel represents an element of the array, that in this case are integer values of type int. These elements are numbered from 0 to 4 since in arrays the **first index** is always 0, independently of its length.

Initializing arrays.

When declaring a regular array of local scope (within a function, for example), if we do not specify otherwise, its elements will not be initialized to any value by default, so their content will be undetermined until we store some value in them.

The elements of global and static arrays, on the other hand, are automatically initialized with their default values, which for all fundamental types this means they are filled with zeros.

In both cases, local and global, when we declare an array, we have the possibility to assign initial values to each one of its elements by enclosing the values in braces { }. For example:

```
A1 [5] = { 54, 8776, 32, 98, 2};
```

This declaration would have created an array like this:

	0	1	2	3	4
A1	54	8776	32	98	2

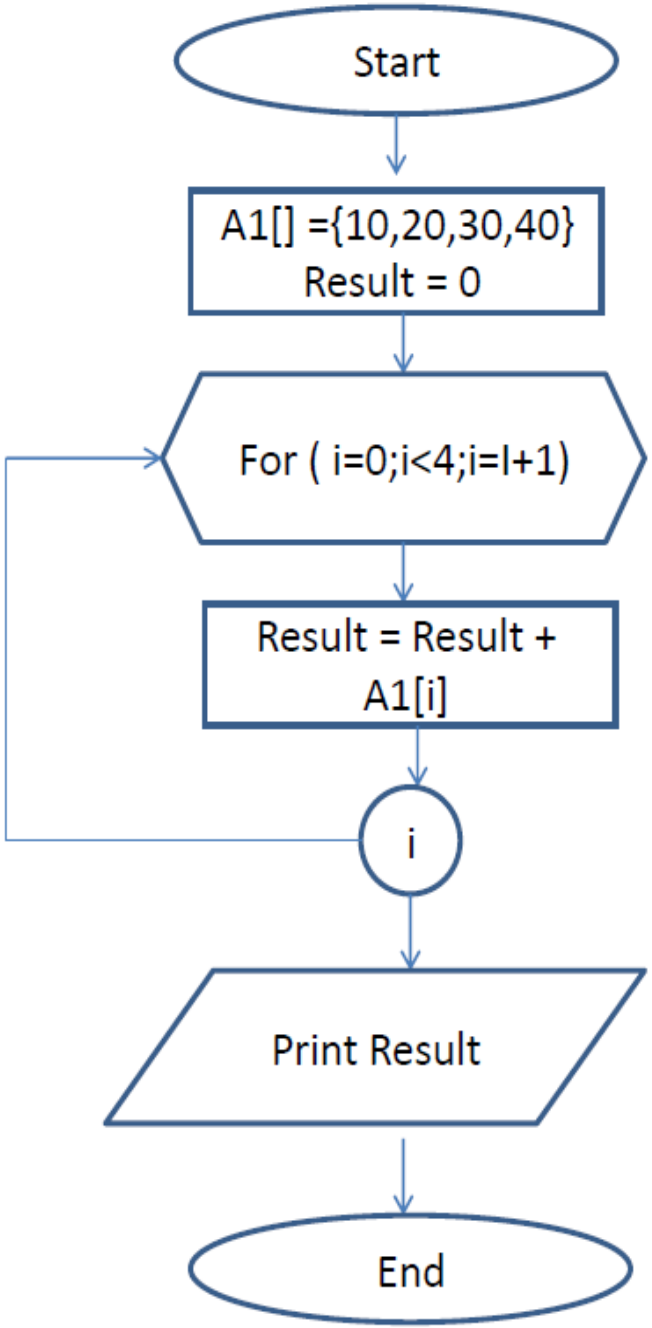
The amount of values between braces { } must not be larger than the number of elements that we declare for the array between square brackets [ ]. For example, in the example of array A1 we have declared that it has 5 elements and in the list of initial values within braces { } we have specified 5 values, one for each element.

When an initialization of values is provided for an array, C++ allows the possibility of leaving the square brackets empty [ ]. In this case, the compiler will assume a size for the array that matches the number of values included between braces { }:

A1 [ ] = { 54, 8776, 32, 98, 2};

After this declaration, array A1 would be 5 int's long, since we have provided 5 initialization values.

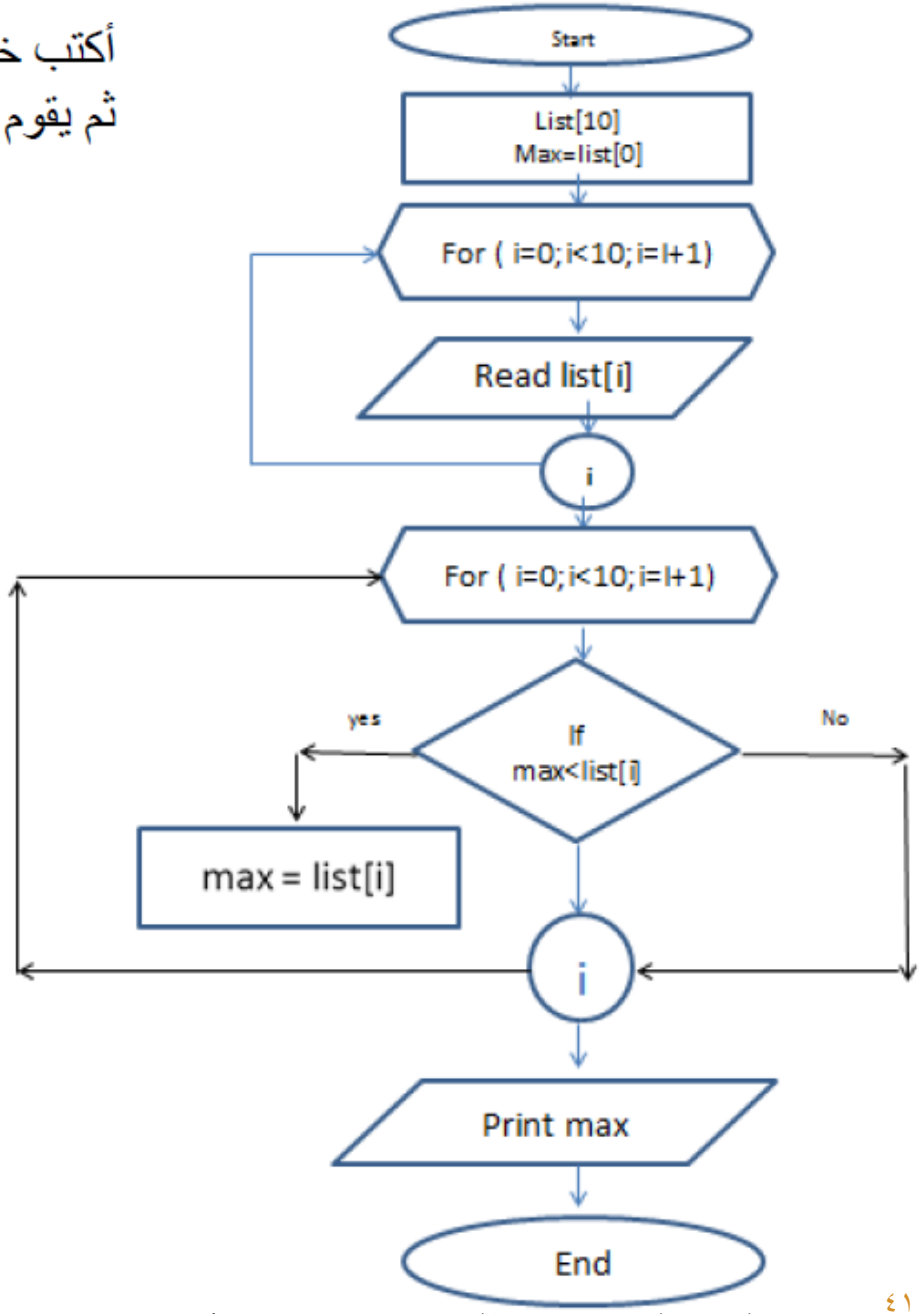
```
1- Start
2- A1[]={10,20,30,40}
3- result=0
4- for (i=0;i<4;i=i+1)
      result = result + A1[i]
5- print result
6-End
```



أكتب خوارزميه تقوم بإدخال قيم مصفوفه ذات بعد واحد ثم يقوم بإيجاد القيمة الأكبر بينهم

1- Start
2- list[10], max=list[0].
3- for(i=0;i<10;i=i+1)
   Read list[i]
4- for(i=0;i<10;i=i+1)
     if max < list [i] then max = list[i]

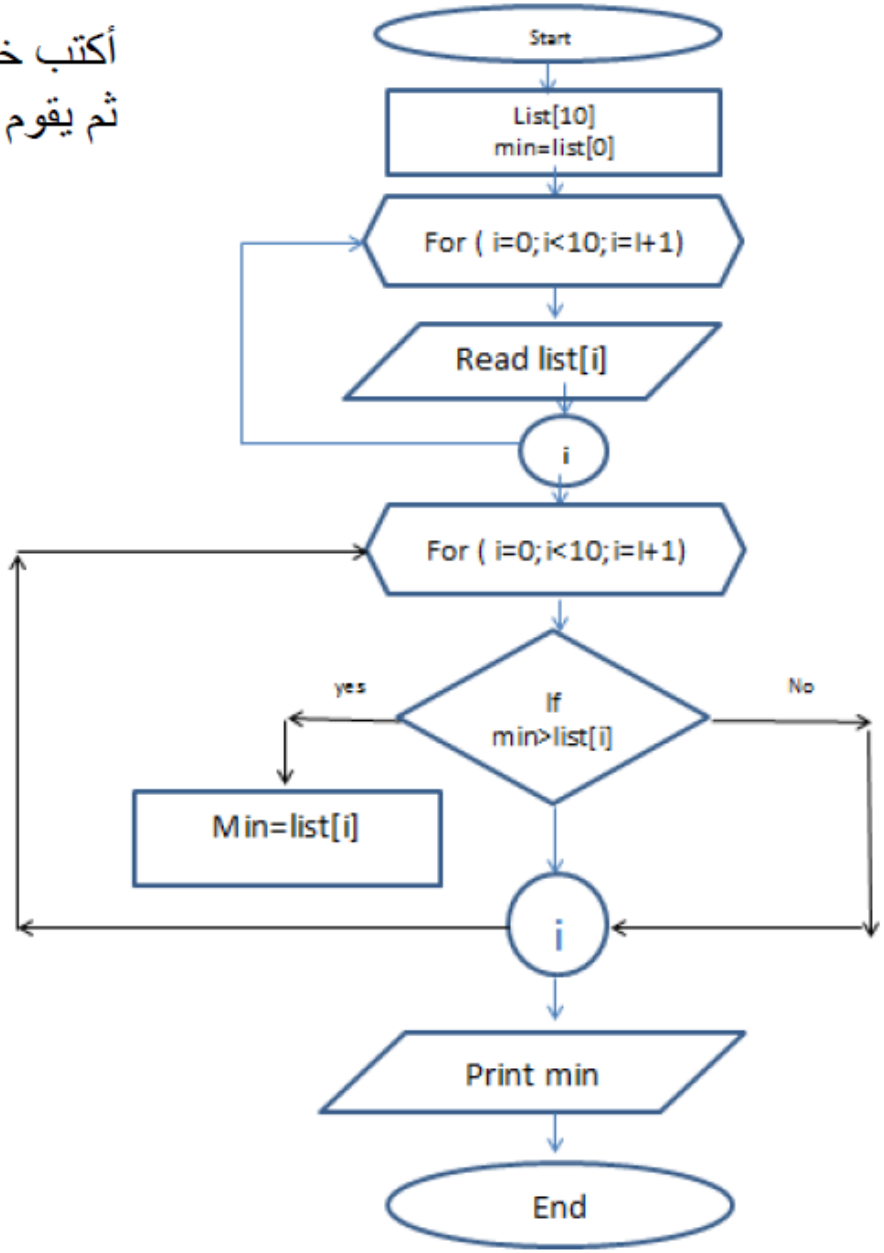
5- print max
6- End



أكتب خوارزميه تقوم بإدخال قيم مصفوفه ذات بعد واحد ثم يقوم بإيجاد القيمة الأصغر بينهم

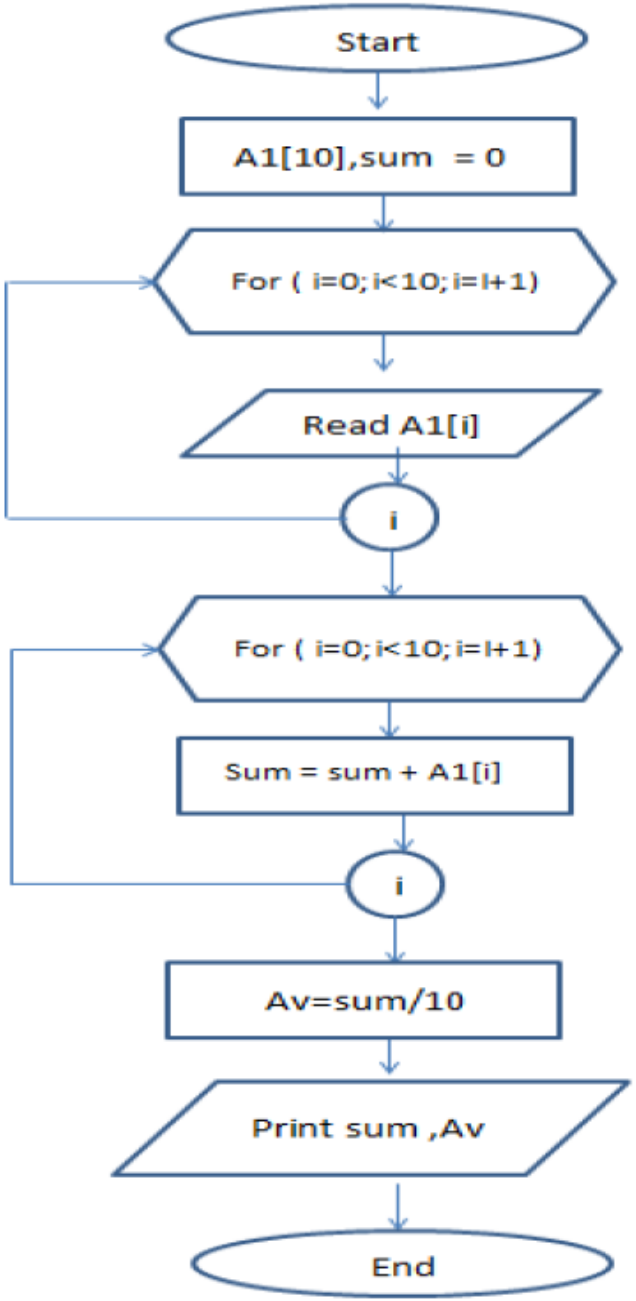
1- Start
2- list[10], min=list[0].
3- for(i=0;i<10;i=i+1)
   Read list[i]
4- for(i=0;i<10;i=i+1)
      if min > list [i] then min= list[i]

5- print min
6- End



أكتب خوارزميه تقوم بإدخال عناصر مصفوفه مكونة من ١٠ عناصر
ثم يقوم بطباعة حاصل جمع كل العناصر والمعدل .

```
1- start
2- A1[10],  sum=0
3- for(i=0,i<10;i=i+1)
   Read A1[i]
4- for (i=0;i<10;i=i+1)
       sum=sum+A1[i]
5- av =sum/10
6- print sum , av
8- End
```



Multidimensional arrays

Multidimensional arrays can be described as "arrays of arrays". For example, a bidimensional array can be imagined as a bidimensional table made of elements, all of them of a same uniform data type.

		0	1	2	3	4
	0					
B1	1					
	2					

B1 represents a bidimensional array of 3 per 5 elements of type int.

B1 [3][5];

and, for example, the way to reference the second element vertically and fourth horizontally in an expression would be:

B1[1][3]

		0	1	2	3	4
	0					
B1	1				X	
	2					

B1[1][3]

(remember that array indices always begin by zero).

Multidimensional arrays are not limited to two indices (i.e., two dimensions). They can contain as many indices as needed. But be careful! The amount of memory needed for an array rapidly increases with each dimension. For example:

```
century [100][365][24][60][60];
```

declares an array with a char element for each second in a century, that is more than 3 billion chars. So this declaration would consume more than 3 gigabytes of memory!

Multidimensional arrays are just an abstraction for programmers, since we can obtain the same results with a simple array just by putting a factor between its indices:

```
int B1[3][5]; // is equivalent to
int B1[15]; // (3 * 5 = 15)
```

Example

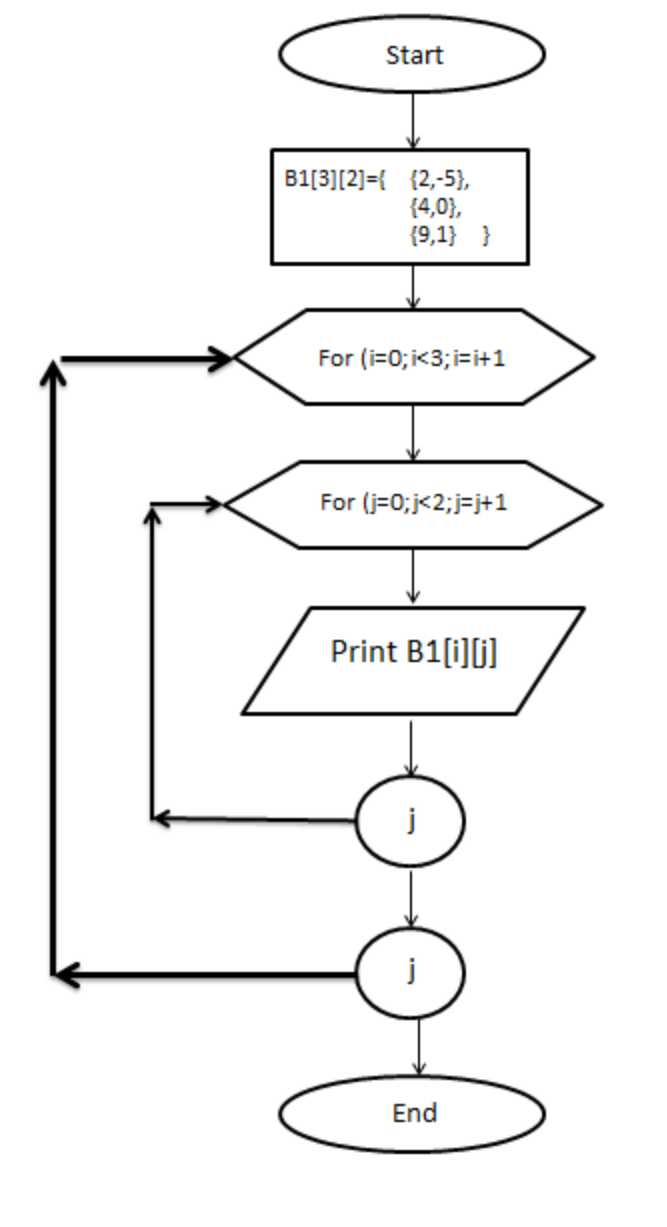
		0	1
B1	0	2	-5
	1	4	0
	2	9	1

1- start

2- B1[3][2]={ {2,-5},
{4,0},
{9,1} }

3- for (i=0;i<3;i=i+1)
For (j=0;j<2;j=j+1)
print B1[i][j]

4- End



Example 2 : Write an Algorithm that build **a matrix of 5 rows and 3 columns**. As the use to enter the values of all the matrix items, print out the sum of al matrix items and print out the sum of the diagonal items

Matrix

	0	1	2
0	1	2	3
1	4	5	6
2	7	8	9
3	10	11	12
4	13	14	15

Home work : the flowchart

```
1- start
2-  matrix [5][3], sum=0,
    sumdiagonal=0
3- for (r=0;r<5;r=r+1)
     for(c=0;c<3;c=c+1)
          Read matrix[r][c]
4-for(r=0;r<5;r=r+1)
      for(c=0;c<3;c=c+1)
        sum=sum + matrix[r][c]
5-for(r=0;r<5;r=r+1)
      for(c=0;c<3;c=c+1)
        if(r=c) then   sumdiagonal= sumdiagonal+ matrix[r][c]
6-Print sum, sumdiagonal
7- End
```

Example 3 :Write an algorithm that build **a matrix of 5 rows and 5 columns**. As the use to enter the values of all the matrix items, print out the sum of al matrix items and print out the main and sub diameter of array

A1

	0	1	2	3	4
0	10	11	12	13	14
1	15	16	17	18	19
2	20	21	2 2	23	24
3	25	26	27	28	29
4	30	31	32	33	34

Home work : the flowchart

```
1- start

2- A1[5][5], sum=0

3- for (i=0;i<5;i=i+1)
      for(j=0;j<5;j=j+1)
         Read A1[i][j]

4- for (i=0;i<5;i=i+1)
      for(j=0;j<5;j=j+1)
         sum=sum+A1[i][j]

5- for (i=0;i<5;i=i+1)
      for(j=0;j<5;j=j+1)
         if(i=j) then Print "The Main diameter is",A1[i][j]

6- for (i=0;i<5;i=i+1)
      for(j=0;j<5;j=j+1)
        if(i+j=4) then print "The sub diameter is ", A1[i][j]

7= End
```

Read the element of array

Sum all the element of array

Print the element of main diameter of array

Print the element of sub diameter of array

Square Matrix

Same number of rows and columns

$$B = \begin{bmatrix} 5 & 4 & 7 \\ 3 & 6 & 1 \\ 2 & 1 & 3 \end{bmatrix}$$

Transpose Matrix

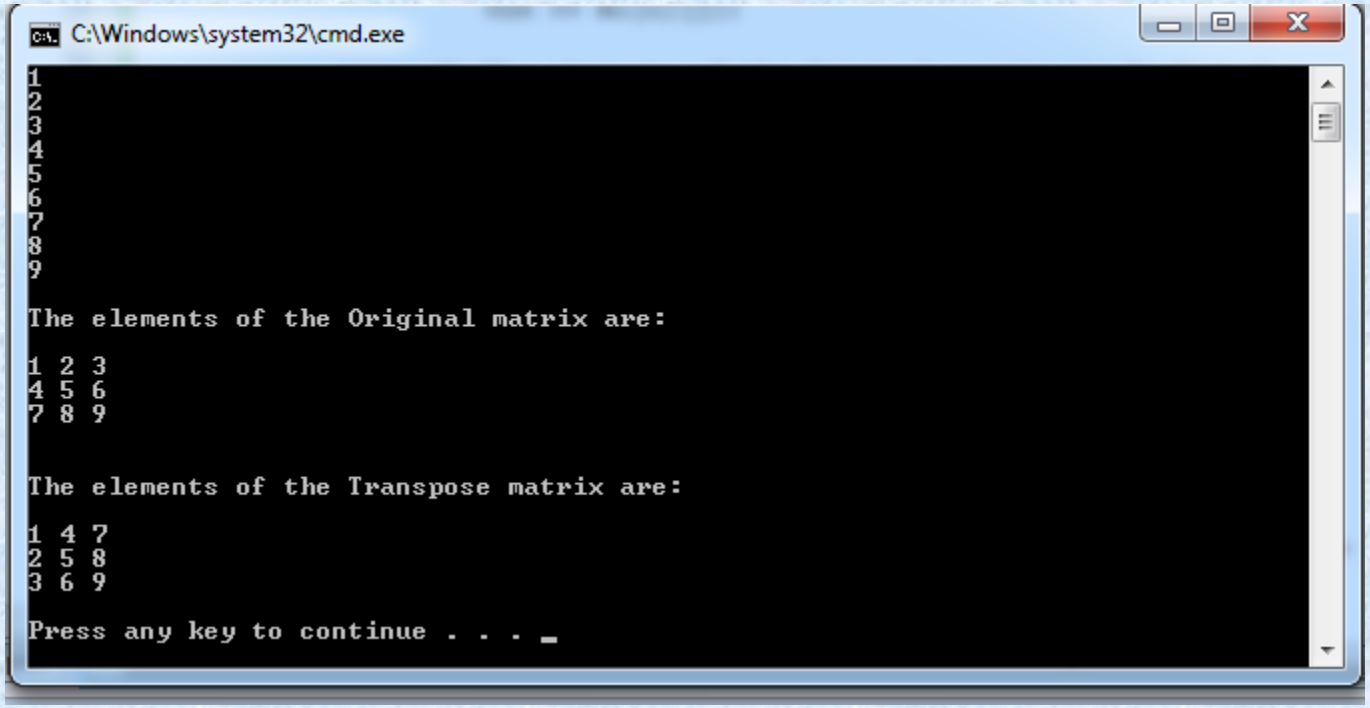
Rows become columns and columns become rows

$$A' = \begin{bmatrix} a_{11} & a_{21} & ,\ldots, & a_{m1} \\ a_{12} & a_{22} & ,\ldots, & a_{m2} \\ \ldots & \ldots & \ldots & \ldots \\ a_{1n} & a_{2n} & ,\ldots, & a_{mn} \end{bmatrix}$$

Example of a transpose

- Thus,

$$\boldsymbol{A} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ a_{31} & a_{32} \end{bmatrix} \qquad \boldsymbol{A}' = \begin{bmatrix} a_{11} & a_{21} & a_{31} \\ a_{12} & a_{22} & a_{32} \end{bmatrix}$$



Homework:
Write an algorithm and draw the flowchart to get these output

Special matrices

- There are a number of special matrices
  - Diagonal
  - Null
  - Identity

Diagonal Matrices

- A diagonal matrix is a square matrix that has values on the diagonal with all off-diagonal entities being zero.

$$\begin{bmatrix} a_{11} & 0 & 0 & 0 \\ 0 & a_{22} & 0 & 0 \\ 0 & 0 & a_{33} & 0 \\ 0 & 0 & 0 & a_{44} \end{bmatrix}$$

Null Matrix

- A square matrix where all elements equal zero.

$$\begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Identity Matrix

Square matrix with ones on the diagonal and zeros elsewhere.

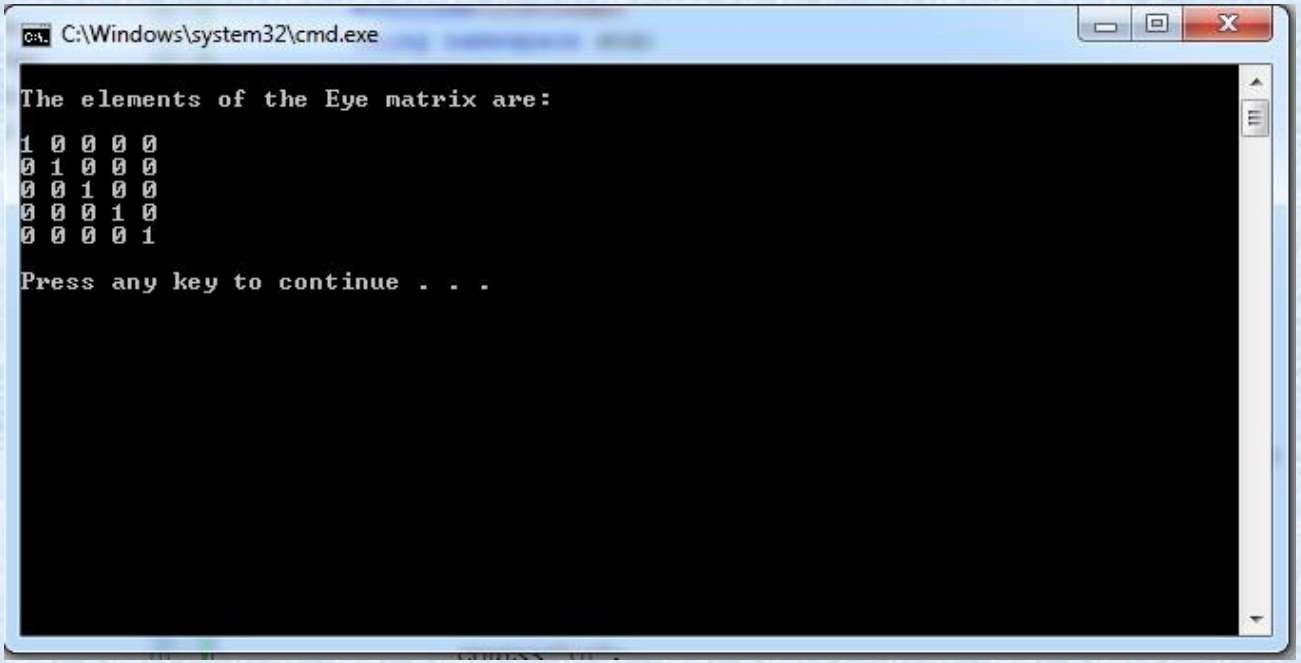
$$I = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Identity Matrix

- An identity matrix is a diagonal matrix where the diagonal elements all equal one.

$$I = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$A \times I = A$$



Homework:
Write an algorithm and draw the flowchart to get these output