المحاضرة الأولى:

1- مقدمة :

إن أهم مرحلة في حل مسألة ما باستخدام الحاسوب هي المرحلة المتعلقة بإيجاد خطة الحل , يجب أن تكون هذه الخطة قابلة للتنفيذ من قبل الآلة , وقابلة للتوصيف على وجهٍ لا يدعو الى اللبس أو التأويل , يطلق اسم الخوارزمية على هذه الخطة .

2- تعريف الخوارزمية :

مجموعة الخطوات المتسلسلة والمحدودة التي تؤدي إلى حل مسألة معينة والوصول إلى نتائج محددة اعتباراً من معطيات ابتدائية.

3- أنواع الخوارزميات :

- ✓ خوارزميات حسابية : تهتم بالمسائل الرياضية
  مثالها (حل معادلة من الدرجة الأولى)
- ✓ خوارزميات غير حسابية :لا تهتم بالمسائل الرياضية ولكنها تحتاج إلى حل منطقي
  مثالها (طريقة التدقيق الإملائي لنص ما ، اتخاذ قرار بالذهاب إلى مكان ما وتحديد الطريق الأمثل للوصول إليه)

4- طرق التعبير عن الخوارزمية :

- ✓ الطريقة الكلامية : كتابة الخوارزميات على شكل خطوات باستخدام اللغة المتداولة كاللغة العربية أو الإنكليزية .
- ✓ الطريقة الرمزية : كتابة الخوارزميات باستخدام الرموز.
- ✓ الطريقة التدفقية : كتابة الخوارزميات باستخدام المخططات البيانية(المخططات التدفقية).

مثال توضحي :

اكتب الخوارزمية التي تعطي نتيجة حل التعبير الرياضي الآتي باستخدام اللغة المتداولة (الطريقة الكلامية) :

$$Y=(x^2+7)/x(x+2)$$

علماً بأن x معلومة

الحل :

يمكن التعبير عن الخوارزمية باللغة المتداولة(العربية) على الشكل الآتي:

الخطوة الأولى : أدخل قيمة المتحول x.

الخطوة الثانية: احسب المقام

الخطوة الثالثة: إذا كان المقام مساوياً للصفر اطبع " المسألة ليس لها حل"

الخطوة الرابعة: احسب البسط

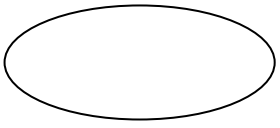
الخطوة الخامسة: احسب قيمة y .

الخطوة السادسة: اطبع قيمة y.

الخطوة السابعة: توقف .

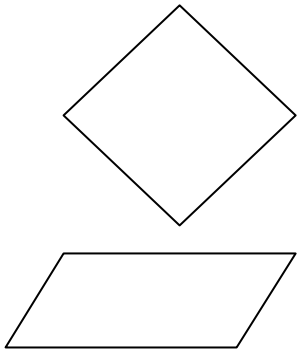
5- المخطط التدفقي (أو الهندسي) :

لتحديد بداية الخوارزمية ونهايتها :



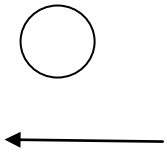
عمليات المعالجة:

العمليات التي ترتبط باختبار تحقق شرط ما وتتطلب قرارًا منطقيا :



عمليات الإدخال والإخراج :

عمليات الربط في حال تعدد الصفحات :

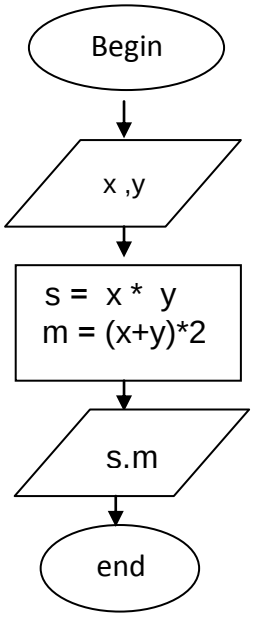


اتجاه تنفيذ الخوارزمية :

6- أمثلة :

اكتب الخوارزمية الكلامية والرمزية والمخطط التدفقي لإيجاد مساحة ومحيط المستطيل ؟

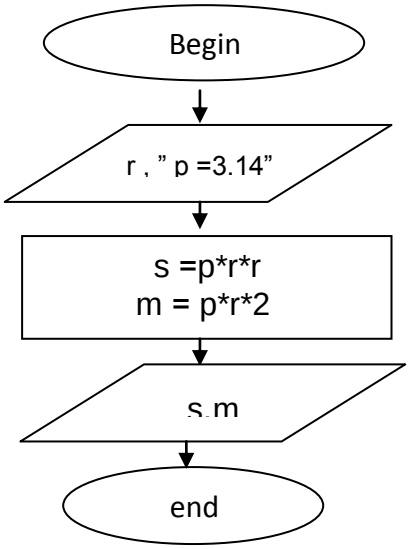
❖ **الخوارزمية الكلامية :**	❖ **الخوارزمية الرمزية :**
المدخلات : الطول والعرض **المعالجة** : المساحة (s)=الطول *العرض المحيط (m) =(الطول+ العرض)*2 **المخرجات** : المساحة والمحيط	**المدخلات :** x و y **المعالجة :** y * x = (s) 2 * (x + y) = (m) **المخرجات :** m, s



تمارين:

التمرين الأول: على نمط المثال السابق اكتب الخوارزمية الكلامية والرمزية والمخطط التدفقي لإيجاد مساحة ومحيط الدائرة ؟

الخوارزمية الكلامية :	**الخوارزمية الرمزية :**
المدخلات : نصف القطر و π المعالجة : المساحة (s)= π *نصف القطر تربيع المحيط(m) = نصف القطر *2*π المخرجات : المساحة والمحيط للدائرة	المدخلات : r و p=3.14 المعالجة : r * r * p = (s) p * 2 * r = (m) المخرجات : m, s



التمرين الثاني : اكتب الخوارزمية الرمزية والمخطط التدفقي لإدخال x(عدد) وإيجاد قيمة : y= (x-2)/x

الخوارزمية الرمزية :

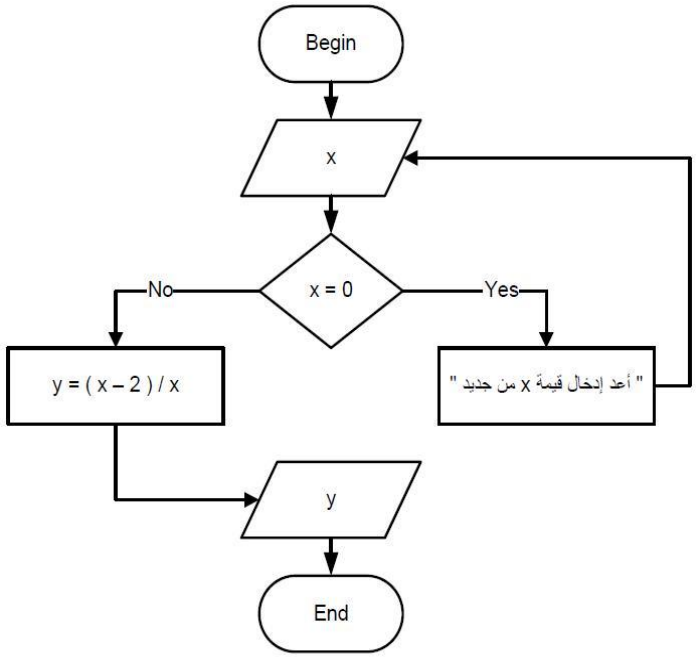
المدخلات : x

المعالجة :

إذا كانت (x = 0) عندئذ "أعد إدخال قيمة x من جديد لأنه لا يمكن القسمة على 0 "

وإلا فاحسب : y = (x – 2) / x

المخرجات : y



التمرين الثالث : اكتب الخوارزمية الرمزية والمخطط التدفقي لإيجاد y=x/(x-3)

الخوارزمية الرمزية :

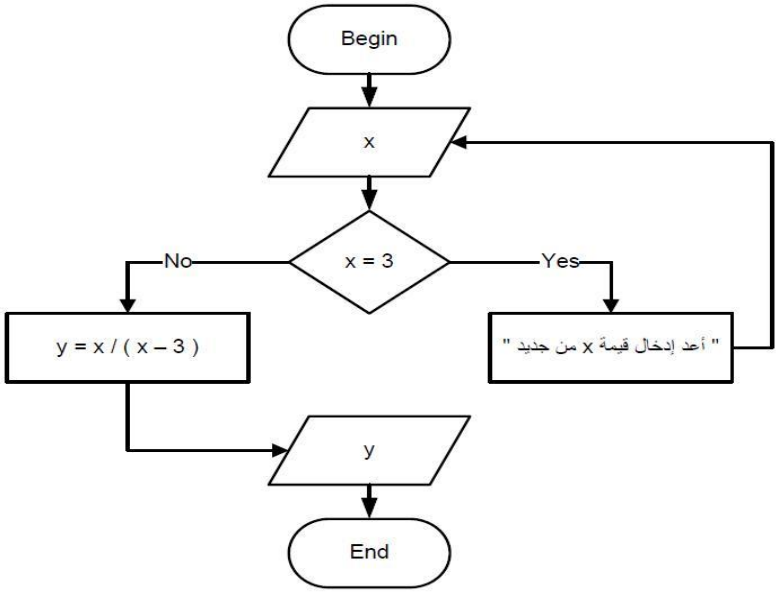
المدخلات : x

المعالجة :

إذا كانت (x = 3) عندئذ "أعد إدخال قيمة x من جديد لأنه لا يمكن القسمة على 0 "

وإلا فاحسب : y = (x – 3) / x

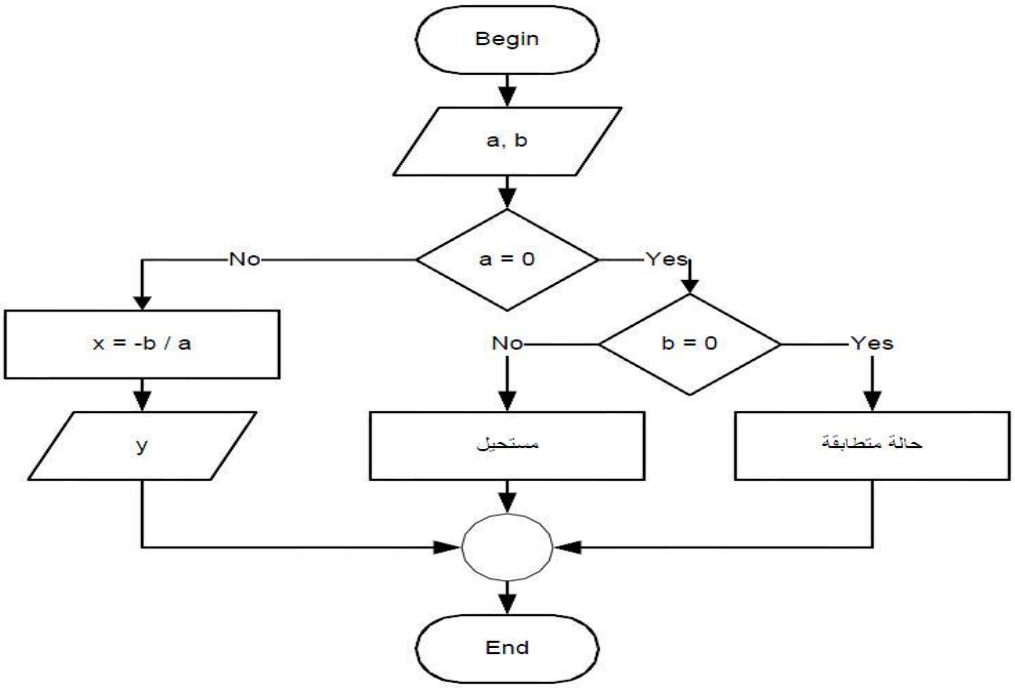
المخرجات : y



التمرين 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

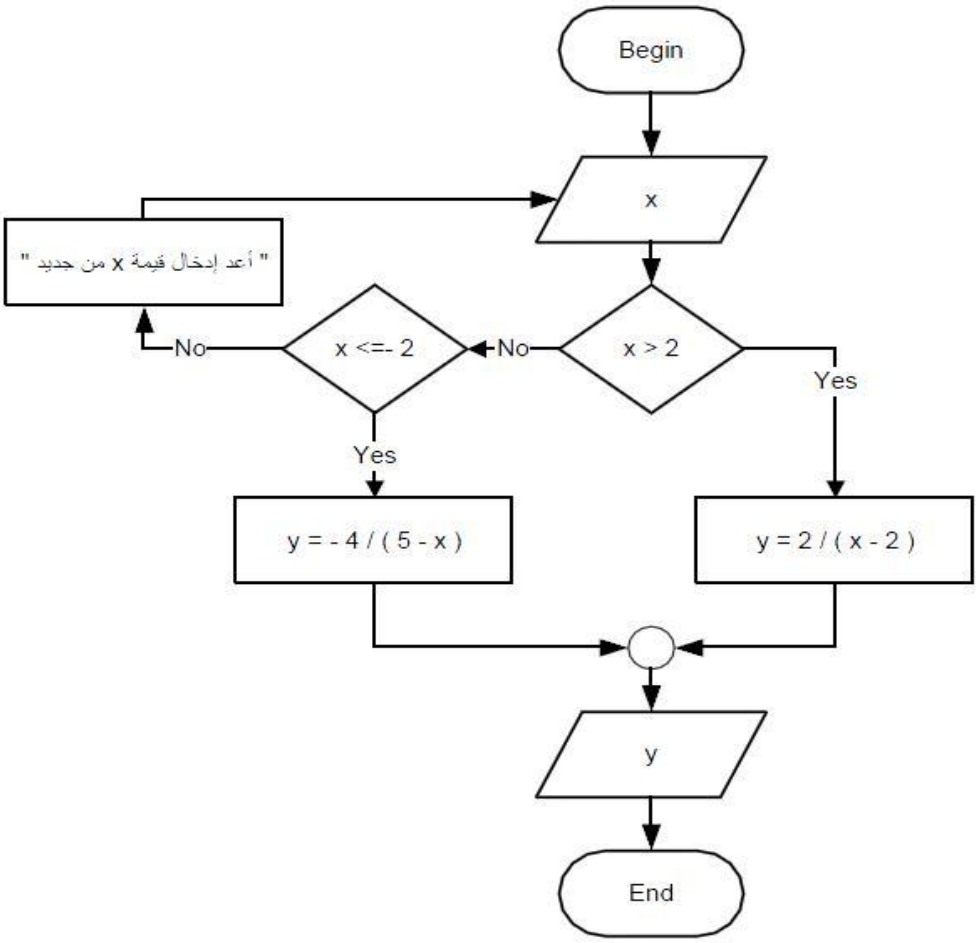


التمرين الخامس : اكتب الخوارزمية الرمزية والمخطط التدفقي لإيجاد قيمة y المعطاة بالشكل التالي :

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

الخوارزمية الرمزية :

المدخلات : x
المعالجة :
إذا كانت (x > 2) عندئذ : y = 2 / (x – 2)
وإلا إذا كانت (x <= -2) عندئذ : y = -4 / (x – 5)
وإلا أعد إدخال x
المخرجات : y



التمرين السادس :

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

الخوارزمية الرمزية :

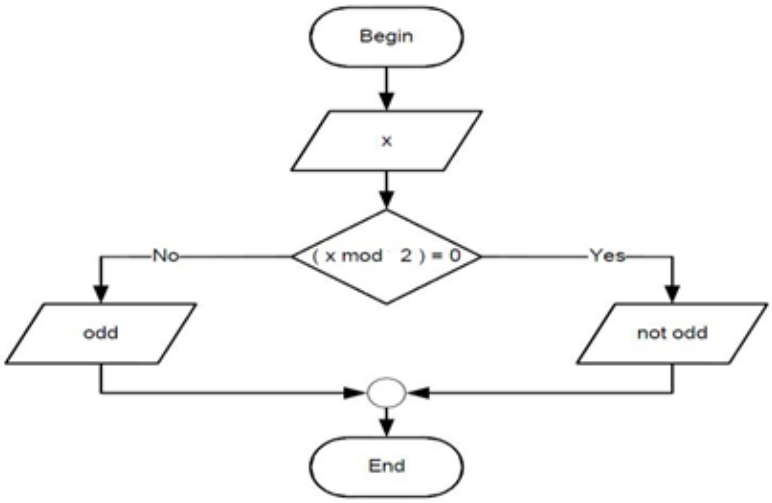
المدخلات : x

المعالجة والمخرجات :

إذا كان باقي قسمة العدد على 2 يساوي صفر (x mod 2 = 0) فإن

اطبع : " العدد زوجياً not odd"

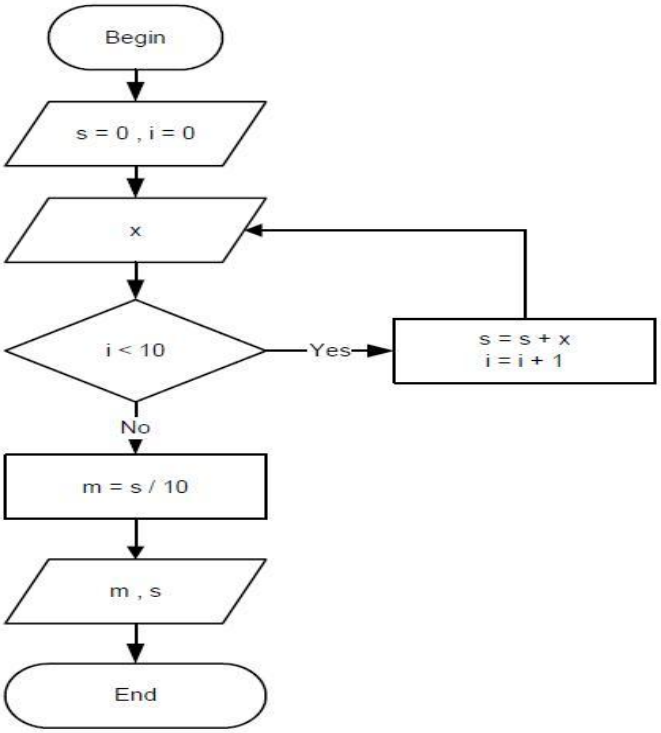
وإلا اطبع : " العدد فردي odd "



التمرين السابع :اكتب الخوارزمية الرمزية والمخطط التدفقي لإدخال عشرة أعداد مختلفة وإيجاد المتوسط الحسابي والمجموع .

الخوارزمية الرمزية :

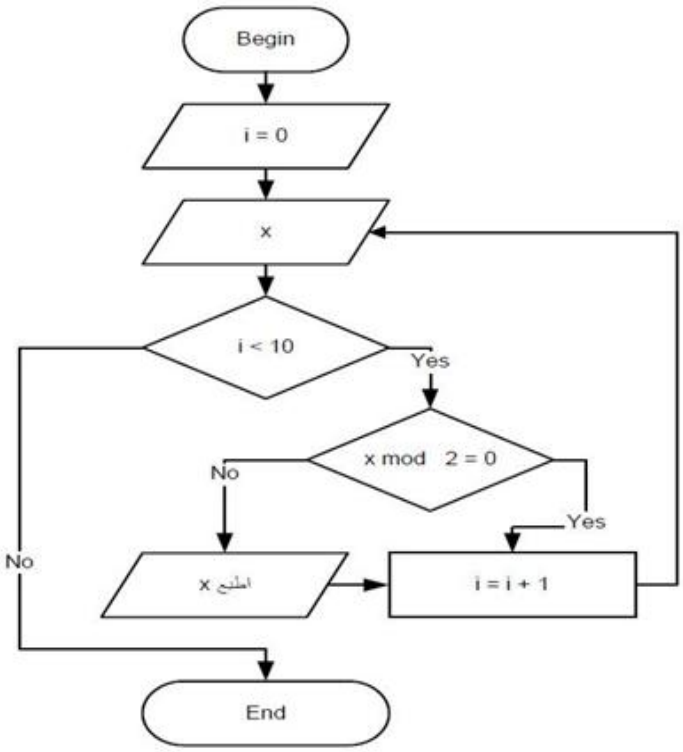
* المدخلات : x , i=0 , s=0
* المعالجة :
العداد (i=i+1) ، المجموع (s=s+x)
إذا كان i<10 عندئذ " أعد إدخال العدد "
وإلا i>=10 عندئذ : " توقف عن إدخال " واحسب :
m=s/10
* المخرجات :
المجموع (s) ، المتوسط (m)



التمرين الثامن : اكتب الخوارزمية الرمزية والمخطط التدفقي لإدخال عشرة أعداد وطباعة الفردي منها فقط ؟

الخوارزمية الرمزية :

* المدخلات : x , i=0
* المعالجة و المخرجات :
العداد (i=i+1)
إذا كان i<10 عندئذ " وإذا كان (x mod 2 = 0)
عندئذ i=i+1 و " أعد إدخال x "
وإلا اطبع قيمة x الحالية ثم أدخل قيمة جديدة لـ x
i=i+1 وشغل العداد
وإلا اخرج من البرنامج



أساسيات البرمجة بلغة C++

أولاً ..أنواع اللغات :

يمكن تقسيم اللغات المستخدمة في البرمجة الى ثلاثة أنواع :

1- لغة الآلة
2- لغة المجمع
3- اللغات عالية المستوى .

1- **لغة الآلة** :هي اللغة التي يستطيع الحاسب أن يفهمها مباشرة وهي معرفة من قبل البنية الصلبة للحاسب ،تتألف بشكل عام من سلاسل من الأعداد (مجموعات من الأصفار والواحدات) التي تعطي الأوامر للحاسب من اجل تنفيذ تعليماته الأولية كل تعليمة على حده .

2- **لغة المجمع** : هي لغة تستخدم مصطلحات قريبة من اللغة الانكليزية للتعبير عن العمليات الأولية للحاسب ،وقد تم تطوير مترجمات البرامج تسمى بالمجمعات assemblers تحويل البرامج من لغة المجمع إلى لغة الآلة .

3- **اللغات عالية المستوى** : هي اللغات التي ظهرت لتسريع عملية البرمجة وذلك باستخدام تعليمات تقوم بالعديد من المهام الجوهرية مثل لغة C++ .

تعتبر لغة C++ من أشهر اللغات التي تتمتع بطابع القوة والمرونة لإنتاج أسرع برامج وأداء أفضل . وعلى الرغم من وجود العديد من لغات البرمجة الأخرى إلا أنها تفتقر شمولية لغة C++ وقوتها . فاللغة C++ تتميز بقابليتها على معالجة التطبيقات الكبيرة والمعقدة، والقوة في صيانة البرامج المكتوبة بها مما يوفر وقتاً في تصميم البرامج وتطويرها .

ملاحظة :تدعى البرامج التي تقوم بتحويل النصوص من البرامج مكتوبة بلغات عالية المستوى الى لغة الآلة بالمترجمات

ثانياً.. كيفية كتابة برنامج بلغة (C ++):

الشكل العام للبرنامج هو :

```
#include<iostream.h>
main ( )
{
    هنا يتم تعريف البيانات ومن ثم يتم كتابة البرنامج
  return 0 ;
}
```

المكتبات الحاوية على العمليات الخاصة بالدخل والخرج .

التابع الرئيسي الذي يبدأ من عنده التنفيذ

القوس للدلالة على بداية البرنامج

القوس للدلالة على نهاية البرنامج

1 - طباعة نص مؤلف من سطر :

```
// First Program                    // كل الكتابات التي تلي هذه الإشارة ( // ) تسمى تعليق لا يتم تنفيذه
  #include <iostream.h>              //  ( .h ) توجيه ما قبل الترجمة حيث يتم ضم محتوى الملف الرأسي ذو
                                     الامتداد الحاوي على العمليات الخاصة بالدخل و الخرج لنص البرنامج

main( )                            //  التابع الرئيسي الذي يبدأ من عنده التنفيذ
{                                  //  بداية البرنامج
   cout << " welcome to c++ " ;    // تعليمة الطباعة
    return 0 ;                     // إحدى طرق الخروج من التابع
}                                  //نهاية البرنامج
```

2 – برنامج جمع عددين صحيحين:

```
   #include <iostream.h>
    main( )
    {
        int x1 , x2 , x3 ;                  //  تعريف المتحولات
        cout << " enter first number " ;    // تعليمة الطباعة
        cin  >> x1   ;                      // تعليمة قراءة متحول
        cout << " enter second number " ;
        cin  >> x2   ;
        x3=x1+x2   ;                        // x3 إجراء عملية الجمع والإسناد إلى المتحول الجديد
        cout  << "sum is " << x3   ;        // تعليمة الطباعة المتعددة
        return 0 ;
    }
```

ثالثاً.. المتحولات (المتغيرات) Variables:

كل اسم من أسماء المتحولات مثل .... , x1, x2 , x3 يتم وضعه في الذاكرة ويعرف باسم name ونمط type وحجم size وقيمة value .

وبالتالي فإن المتحول x1 يملك الاسم x1 والنمط int والحجم 4بايت والقيمة هي حسب القيمة المقروءة .

5	**x1**
10	**x2**
15	**x3**

مواضع المتحولات في الذاكرة مع ذكر الاسم والقيمة

✓ ***أنواع المتحولات :***

- *المتحول المحرفي* — *char*
- *المتحولات الصحيحة* — *short int , int , long int*
- *المتحولات الحقيقية* — *float , double , long double*
- *المتحول المنطقي* — *bool*
- *الثابت (ويضاف لنوع المتحول)* — *const*

رابعاً .. قواعد تسمية المتحولات :

بشكل عام يتكون اسم المتحول من مجموعة من الحروف (A ----> Z ,a ----> z) والأرقام ويخضع اسم المتحول للقواعد التالية :

- لا يمكن أن يبدأ برقم .
- لا يحتوي على سلسلة فارغة .
- يجب أن لا يكون من الأسماء المحجوزة في لغة C++ مثل:

true,false,break,puplic,private,static,new,……..,etc

يكتب تعريف المتحول وفق الصيغة :

Data-Type name = initial-value ;

نوع المعطيات اسم المتحول قيمة ابتدائية

إن ذكر اسم المتحول ونوع المعطيات الذي ينتمي إليه هو أمر اجباري في كل تعريف لأي متحول ،أما اسناد قيمة ابتدائية له عند تعريفه فهو أمر اختياري يمكن عدم ذكره في عملية التصريح عن المتحول .

ملاحظة :من الجدير بالذكر أن لغة C++ تفرق بين الحروف الأبجدية الصغيرة والكبيرة

يبين الجدول التالي أنواع المتحولات ومجالاتها وحجمها :

نوع المتحول	الحجم	المجال
char	1 byte	-128 to 127
int	4 bytes	-2147483648 to 2147483647
short int	2 bytes	-32768 to 32767
long int	4 bytes	-2147483648 to 2147483647
bool	1 byte	True or false
float	4 bytes	-38+E3.45 to 38-E3.4
double	8 bytes	-308+E1.7 to 308-E1.7
long double	8 bytes	-308+E1.7 to 308-E1.7

خامساً.. المعاملات في لغة C++ :

A.المعاملات الحسابية الأحادية :

المعامل	الاستخدام	الوصف
++	s++	زيادة قيمة المتحول s بمقدار 1، استخدام قيمة s قبل زيادته
++	++s	زيادة قيمة المتحول s بمقدار 1، استخدام قيمة s بعد زيادته
- -	s- -	انقاص قيمة المتحول s بمقدار 1، استخدام قيمة s قبل انقاصه
- -	- -s	انقاص قيمة المتحول s بمقدار 1، استخدام قيمة s بعد انقاصه

B. المعاملات الحسابية الثنائية :

العملية	الاستخدام	الوصف
+	x+y	جمع المتحول x مع y
-	x-y	طرح المتحول yمن x
*	x*y	ضرب المتحول x بالمتحول y
/	x/y	قسمة المتحول x على المتحول y
%	x%y	باقي قسمة المتحول x على المتحول y

C. المعاملات العلائقية :

تستخدم المعاملات العلائقية لمقارنة قيمتين وتحديد العلاقة بينهما والتي تكون النتيجة إما True أو False (يتم تمثيل True بالقيمة 1 و False بالقيمة 0)

العملية	الاستخدام	الوصف
>	x>y	إعادة True في حال كانت x أكبر تماماً من y وFalse إذا لم يكن xأكبر تماماً من y
>=	x>=y	إعادة True في حال كانت x أكبر أو تساوي y وFalse في الحالة غير ذلك.
<	x<y	إعادة True في حال كانت x أصغر تماما من y وFalse إذا لم يكن xأصغر تماماً من y
<=	x<=y	إعادة True في حال كانت x أصغر أو تساوي y وFalse في الحالة غير ذلك.
==	x==y	إعادة True في حال كانت x تساوي y وFalse في حال عدم المساواة .
!=	x!=y	إعادة True في حال كانت x لا تساوي y وFalse في حال المساواة .

D. معاملات الاسناد :

نستخدم معامل الاسناد لإسناد القيمة الموجودة على يمين المعامل الى المتحول الموجود على يسار المعامل .
x=5 أسندنا القيمة 5 الى المتحول x .

ملاحظة :

يمكننا في لغة C++ إجراء عمليات حسابية مع عمليات الاسناد مثل الجمع والضرب والطرح والقسمة كالتالي :

المعامل	شكل الاستخدام	مساوي لـ
+=	x+=y	x=x+y
-=	x-=y	x=x-y
*=	x*=y	x=x*y
/=	x/=y	x=x/y
%=	x%=y	x=x%y

E. العمليات المنطقية :

يتم استخدام العمليات المنطقية مع العمليات التي يكون لها إحدى قيم True أو False وهذه العمليات هي :

العملية	الرمز
AND	&&
OR	||
NOT	!

✓ **جداول الحقيقة لأدوات الربط المنطقية:**

- **النفي "Not":** لتكن A عبارة منطقية إن تطبيق أداة الربط نفي على هذه العبارة سيعطينا عبارة منطقية جديدة رمزها **!A** .

جدول الحقيقة للنفي:

A	!A
0	1
1	0

- **الوصل "And":** A And B= A&&B

جدول الحقيقة للوصل:

A	B	A&&B
0	0	0
0	1	0
1	0	0
1	1	1

إن العملية A AND B دوماً خاطئة (أي قيمة الحقيقة لها تساوي الصفر) إلا في حالة واحدة فهي صحيحة (أي قيمة الحقيقة لها تساوي الواحد) وذلك في حال كلا العبارتين A,B صحيحتين.

- **الفصل "OR":** A OR B= A||B

جدول الحقيقة للوصل:

A	B	A||B
0	0	0
0	1	1
1	0	1
1	1	1

إن العملية A OR B دوماً صحيحة (أي قيمة الحقيقة لها تساوي الواحد) إلا في حال كلا العبارتين A,B خاطئتين.

سادساً.. أولوية العمليات الحسابية :

عندما نطبق العمليات الحسابية في العبارات الحسابية فإنها تطبق حسب ترتيب معين تبعاً لقواعد الأولية بين العمليات وهي :

1) ( ) الأقواس :يتم تنفيذ الأقواس الداخلية أولاً أما إذا كان لدينا مجموعة من الأقواس جانب بعضها البعض وعلى نفس المستوى عندها يبدأ الحساب من اليسار إلى اليمين .
2) %,/,*: إذا وجدت على نفس المستوى فإنها تنفذ من اليسار الى اليمين .
3) -,+ : تنفذ في النهاية ،إذا وجدت على نفس المستوى فإنها أيضاً تنفذ من اليسار إلى اليمين .

سابعاً.. سلاسل الهروب :

سلسلة الهروب	المعنى
\n	سطر جديد أي وضع المؤشر في بداية السطر التالي
\t	تحريك المؤشر مسافة جدولية أفقية (tab)
\"	تستخدم لطباعة علامة الاقتباس "

```
# include <iostream.h>
main ( )
{
cout <<"welcome to c++ \ n Hello  World " ;
```

حرف الهروب

```
        return 0;
}
```

والخرج يكون :

```
welcome to c++
Hello  World
```

ثامناً.. أمثلة على العمليات :

✓ المعاملات الحسابية الأحادية (معاملات الزيادة والنقصان) :

```
 # include <iostream.h>
main ( )
{ int a=5;
cout<<a<<"\n";          ⟶  5
cout<<a++<<"\n" ;       ⟶  5
cout<<a<<"\n";          ⟶  6
return 0;
}
```

----------**********----------

```
# include <iostream.h>
main ( )
{ int a=5;
cout<<a<<"\n";          ⟶  5
cout<<++a<<"\n" ;       ⟶  6
cout<<a<<"\n";          ⟶  6
return 0;
}
```

```
# include <iostream.h>
main ( )
{ int a=3,b;
b= a++ *2;
cout<<b<<"\n";          ⟶   6
cout<<a<<"\n" ;         ⟶   4
return 0;
}
```

----------**********----------

```
# include <iostream.h>
main ( )
{ int a=3,b=5,c;
c= - -a *2+ b++;
cout<<c<<"\n";          ⟶   9
cout<<b<<"\n";          ⟶   6
cout<<a<<"\n" ;         ⟶   2
return 0;
}
```

✓ <u>أولوية العمليات :</u>

- (5+2)*3+(2*6)/4-1
  =7 * 3 + 12 / 4 – 1
  =21 +3 – 1 =23
- 4 * 8 – 5 + 2 * 6 / 3
  =32 – 5 + 12 / 3
  =32 – 5 + 4=31

✓ <u>المعاملات المنطقية :</u>

- cout<< ((4 > 3) && (3==2)); -----> 1 && 0 ----> 0 النتيجة المطبوعة النهائية
- cout<<(! (4 < 3) || (3==2)); -----> !(0) || 0---> 1 || 0

وبالتالي النتيجة النهائية المطبوعة 1 .

<u>تمارين :</u>

1. اكتب برنامج يأخذ كدخل ثلاث أعداد صحيحة من لوحة المفاتيح ثم يطبع مجموعها ومتوسطها وناتج جداؤها.

```
# include <iostream.h>
main( )
{
int a,b,c ;
cin >> a >> b >> c;
cout << " sum is " << a+b+c << " \n" ;
cout << " average is " << (a+b+c)/3 << " \n";
cout << " product is " << a*b*c;
return 0;}
```

2. اكتب برنامج يقرأ نصف قطر دائرة ثم يطبع قيمة قطر الدائرة ،محيطها ، مساحتها.

```
# include <iostream.h>
main( )
{
float  r ; // تعريف متحول حقيقي
float const p=3.14; // تعريف متحول حقيقي واسناد قيمة ابتدائية له
cin >> r ;
cout << r * 2 << " \n" ;
cout << 2*p*r << " \n";
cout <<p*r*r;
return 0;}
```

3. اكتب برنامج يطبع شكل مستطيل باستخدام سلاسل الهروب .

```
# include <iostream.h>
main()
{
cout << " **********  \n"<<" * \t" << " * \n" ;
cout << " * \t" <<" * \n";
cout << " * \t" <<" * \n";
cout << " * \t" <<" * \n";
cout << " * \t" <<" * \n";
cout << " **********  \n";
return 0;
}
```

4. اكتب برنامج يقوم بحساب مساحة ومحيط مستطيل (قيم الطول والعرض يتم ادخالها من لوحةالمفاتيح).

```
#include<iostream.h>
main()
{float   area,length,width,cir;      // أسماء للمتحولات
cout<<"Enter the length:  ";   //length :الطول
cin>>length;
cout<<"Enter the width:  ";    //width :العرض
cin>>width;
area=length*width;              //المساحة
cir= 2*( length+width);         //المحيط
cout<<"\n Area=  "<<area<<endl;      // تعني انتقل الى سطر جديد endl
cout<<" circumference= "<<cir<<endl;
return 0;}
```

بنى التحكم (الشروط)

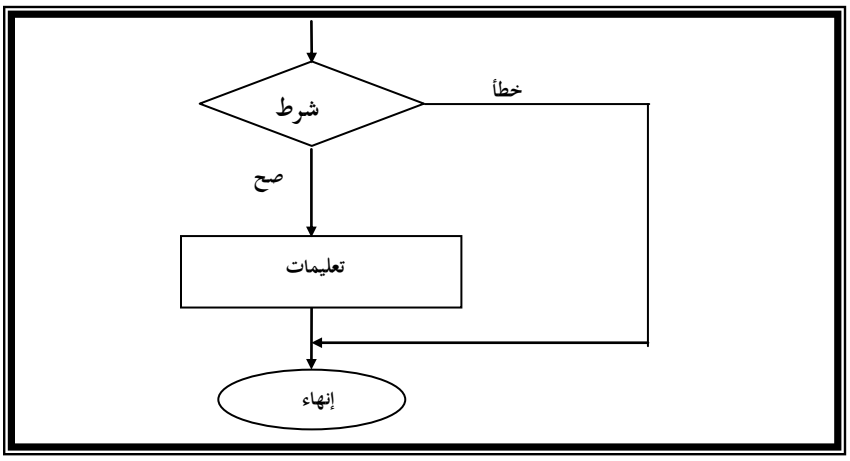
عادة يتم تنفيذ العبارات حسب تسلسل ورودها في البرنامج ويسمى هذا بالتنفيذ التتابعي (Sequential Execution). لكننا سنتعرض لبعض عبارات C++ والتي تجعل التنفيذ ينتقل لعبارة أخرى قد لا تكون التالية في تسلسل البرنامج، ويسمى هذا بنقل التحكم Transfer of control.
تنقسم بنيات التحكم في C++ إلى قسمين: بنى التحكم الشرطية والنوع الثاني وهو بنى التحكم التكرارية .

1. بنية الاختيار if:
تقوم بنية الاختيار if بتنفيذ فعل معين عندما يكون الشرط المرافق لها محققاً وإلا يتم تجاهله (أي في حال عدم تحقق الشرط لا يتم تنفيذ أي شيء) ولها الشكل العام التالي :

```
if(condition الشرط )
{
statements;
}
```

تعليمة أو مجموعة تعليمات في حال كانت تعليمة واحدة ليس هناك ضرورة للأقواس أما في حال كانوا أكثر من تعليمة فهنا يجب وضع هذه الأقواس { } عندئذ تسمى كتلة تعليمات (block)

ويبين الشكل التالي طريقة عمل العبارة if :



مثال 1: علامة النجاح في إحدى الامتحانات تساوي 60 عندها يكون الكود الخاص لمعرفة الطالب إذا كان ناجح هو كالتالي :

```
# include <iostream.h>
main()
 {  int grade;
       cin>>grade;
       if ( grad >= 60 )
       { cout << " passed " ;}
       return 0;}
```

مثال 2 : اكتب برنامج يطلب من المستخدم إدخال عددين صحيحين ثم يطبع العدد الأكبر بينهما متبوعاً بالجملة is larger أما اذا كان العددان متساويين عندها تطبع الكلمة Equals .

```
#include<iostream.h>
main()
 {int x,y;
  cin>>x>>y;
  if(x>y)    {cout<<x<<"is larger";}
  if(y>x)    {cout<<y<<"is larger";}
  if(x==y)   {cout<<"Equals";}
 return 0;
}
```

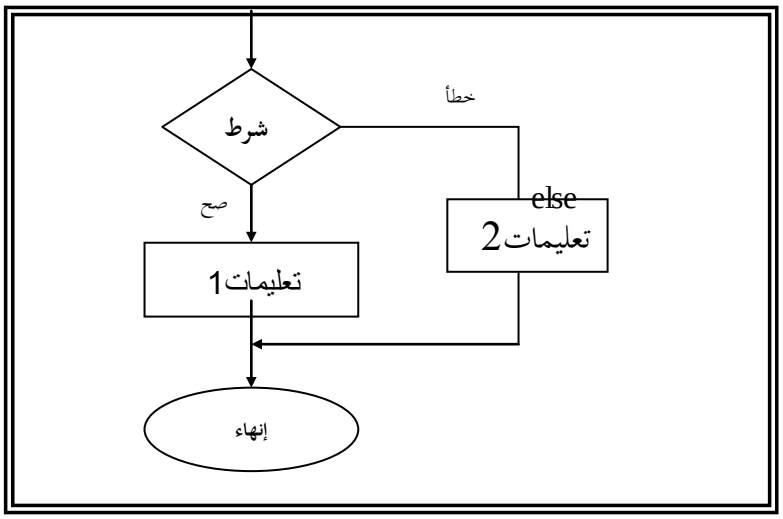
2.العبـارة if...else :

في العبارة if البسيطة يتم تنفيذ فعل معين إذا كان الشرط صحيحاً، لكن إذا لم يكن كذلك لا يحدث شيء على الإطلاق. لكن لنفترض أننا نريد حدوث شيء في الحالتين إذا كان الشرط صحيحاً وآخر إذا لم يكن كذلك، لتحقيق ذلك نستخدم العبارة if... else أي أن البنية **if...else** تسمح بتنفيذ مجموعة من الأفعال الممكن تنفيذها إذا كان الشرط المرافق لها محققاً وتنفيذ مجموعة من الأفعال الأخرى في حال عدم تحقق الشرط المرافق لها ولها الشكل التالي :

```
if(condition الشرط )
  {Statements  1;
  }
else
  {Statements  2;
  }
```

أي اذا تحقق الشرط فإنه يتم تنفيذ Statements 1
أما إذا لم يتحقق يتم تنفيذ Statements 2

ويبين الشكل التالي طريقة عمل البنية if..else :



مثال 3 : قراءة علامة طالب (أي ادخالها من لوحة المفاتيح) وطباعة passed في حال نجاحه وطباعة failed في حال رسوبه (علامة النجاح 60) .

```
#include<iostream.h>
main( )
{ float mark;
 cin>>mark;
 if(mark>=60)
   {cout<<"passed";}
 else
   {cout<<"failed";}
 return 0;
}
```

مثال 4 : برنامج لمعرفة العدد اذا كان زوجي أم فردي .

```
#include<iostream.h>
main()
{int x;
cin>>x;
if(x%2==0)    // شرط العدد الزوجي
  {cout<<"even";}
else
  {cout<<"odd";}
return 0;
}
```

ملاحظة :

هناك طريقة أخرى للتعبير عن البنية if/else في لغة C++وذلك باستخدام ما يسمى بالمعامل الشرطي(-:-?) والذي له الشكل التالي :

شرط ? تعليمة 1 : تعليمة 2 ;

أي في حال تحقق الشرط يتم تنفيذ تعليمة 1 وإلا يتم تنفيذ تعليمة 2

تستخدم هذه البنية عادة في دالة الإخراج وفي تعليمة الاسناد.

مثال 5: طباعة كلمة pass في حال كانت علامة طالب أكثر من 60 و طباعة fail في حال كانت العلامة أقل من 60.

```
# include <iostream.h>
main( )
{  int a;
   cin>>a;
   a>=60 ? cout<< "pass":cout<<"fail";
   return 0 ;
}
```

مثال 6 : طباعة العدد الأكبر بين عددين مدخلين من لوحة المفاتيح .

```
# include <iostream.h>
main( )
{int x,y,max;
  cin>>x>>y;
   max= x>y ? x:y;   // max تم اسناد القيمة المعادة من اختبار الشرط الى المتحول
   cout<<max<<"is greater \n";
 return 0 ;}
```

3.العبارات if ... else المتداخلة:

يمكن وضع العبارات if /else ضمن بعضها البعض ، من أجل القيام بفحص عدة حالات ولها الشكل :

```
if(1 الشرط )
  { Statements  1;      //1 التعليمات التي ستنفذ في حال تحقق الشرط
  }
else if(2 الشرط)
     {   Statements  2;   //2 التعليمات التي ستنفذ في حال عدم الشرط 1و تحقق الشرط
     }
      else
        {  Statements  3;   // التعليمات التي ستنفذ في حال عدم تحقق جميع الشروط السابقة
        }
```

مثال 7 : اكتب برنامج يطلب من المستخدم إدخال عددين صحيحين ثم يطبع العدد الأكبر بينهما متبوعاً بالجملة is larger أما اذا كان العددان متساويين عندها تطبع الكلمة Equals (باستخدام البنية if/else المتداخلة)

```
#include<iostream.h>
main( )
{int x,y;
cin>>x>>y;
if(x>y)     {cout<<x<<"is larger";}
 else  if(y>x)       {cout<<y<<"is larger";}
        else          {cout<<"Equals";}
return 0;}
```

4. العبـارة switch :

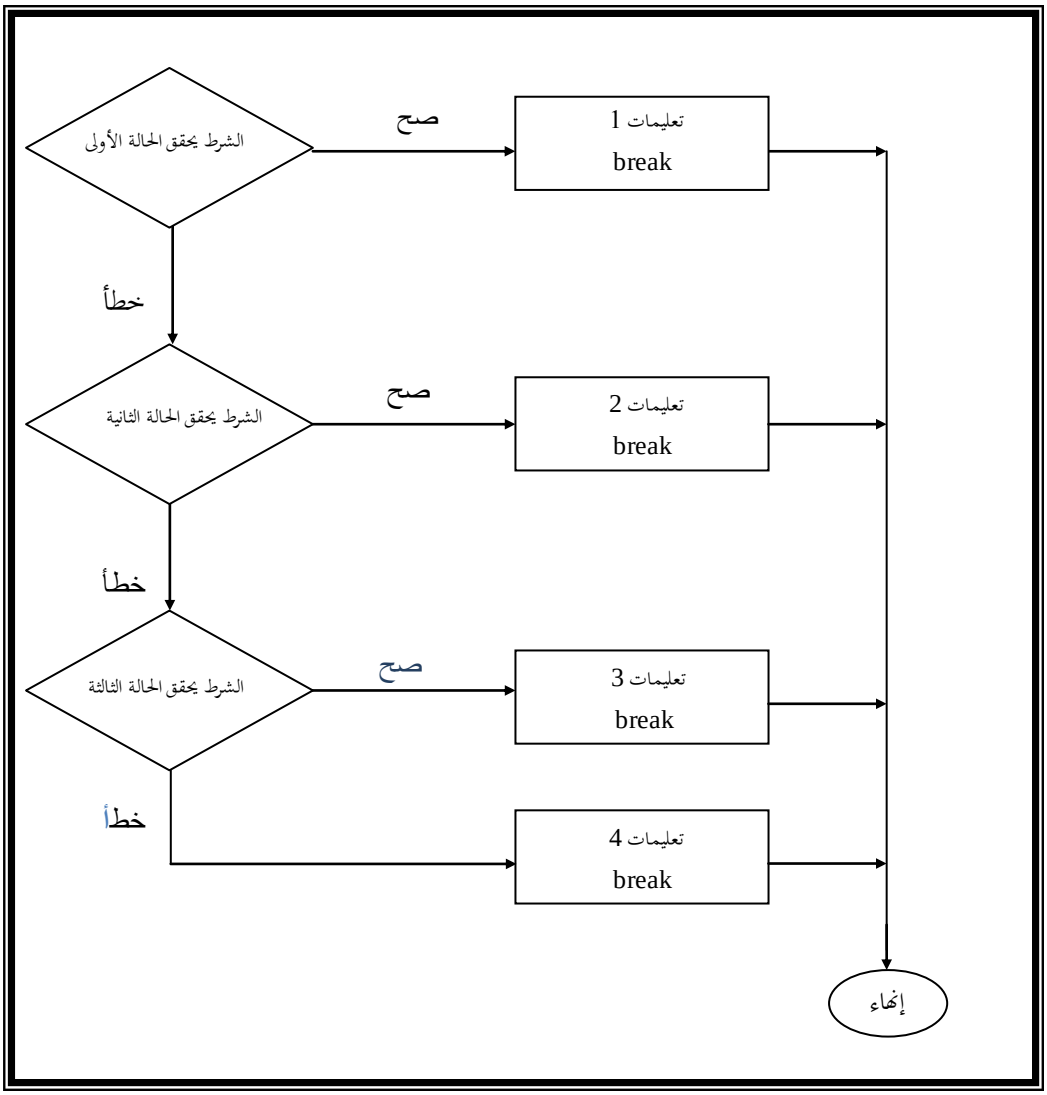
يمكن أن تصادفنا حالة خاصة في إحدى البرامج تحتوي على سلسلة من القرارات التي تتعلق بنتائج متعددة لفحص قيمة متحول أو تعبير ما ،ويمكن ان تؤدي كل نتيجة إلى القيام بفعل مختلف عن الآخر .لذلك توفر لغة C++ البنية switch من أجل التعامل مع حالات اتخاذ القرار المتعلقة بعدة خيارات ،ولها الشكل العام التالي :

```
switch (Variable name)     // المتحول قد يكون عدد أو محرف
{
case constant1 : statement1; break;
case constant2 : statement2; break;
.
.
case constant n : statement n; break;
default : last statement;
}
```

تتألف العبارة switch من الكلمة الأساسية switch يليها اسم متغير بين قوسين ، تفحص العبارة switch المتغير وتوجه البرنامج نحو أقسام مختلفة وفقاً لقيم ذلك المتغير.

يتضمن جسم العبارة switch عدداً من الوسوم وهي أسماء تليها نقطتان. تتألف هذه الوسوم من الكلمة الأساسية case ثم ثابت ثم نقطتين.

عندما تكون قيمة متغير العبارة switch مساوية للثابت المذكور في أحد وسوم case ينتقل التنفيذ إلى العبارات التي تلي ذلك الوسم وتؤدى العبارة break إلى منع تنفيذ بقية العبارة switch (أي الخروج الآمن والقسري من حالة الـcase الحالية وتضمن عدم الدخول الى حالات case الأخرى) ، وإذا لم تتطابق قيمة متغير العبارة switch مع أي وسم ينتقل التنفيذ إلى الوسم الافتراضي default (تعليمة default اختيارية للبنية switch فإذا تم تجاهلها فإن البنية ستعمل بالشكل المعتاد ،فإذا لم يكن هناك أي تطابق بين أي من ثوابت أقسام الـcase وقيمة صيغة التحكم فإن التحكم سينتقل إلى خارج البنية .



مثال 6: طباعة اسم اليوم من أيام الاسبوع عند إدخال رقمه (عند إدخال 1 يطبع Saturday وعند إدخال 2 يطبع Sunday وهكذا .....) .

```
#include<iostream.h>
main( )
{int x;
cin>>x;
switch(x)
        {case 1:cout<<"Saturday";break;
        case 2:cout<<"Sunday";break;
        case 3:cout<<"Monday";break;
        case 4:cout<<"Tuesday";break;
        case 5:cout<<"Wednday";break;
        case 6:cout<<"Thursday";break;
        case 7:cout<<"Friday";break;
        default :cout<<"Wrong Number";
        }
return 0;
}
```

تمارين :

1. برنامج اختبار عدد يتم إدخاله من لوحة المفاتيح إذا كان موجب أم سالب .

```
# include <iostream.h>
main( )
{int a;
cin>>a;
if(a>=0)
     {cout<<"positive";}
else
     {cout<<"negative";}
return 0 ;
}
```

2. اكتب برنامج لاختبار عدد إذا كان قاسم لآخر (يتم إدخال العددين من لوحة المفاتيح).

```
#include<iostream.h>
main( )
{int x,y;
cin>>x>>y;
if(x%y==0 && y!=00)
    {cout<<x<<"is divisible by "<<y;}
else
    {cout<<x<<"is not divisible by "<<y;}
return 0;
}
```

3. برنامج يأخذ كدخل ثلاثة أعداد صحيحة غير متساوية ومن ثم يطبع أصغر هذه الأعداد .

```
# include <iostream.h>
main()
{int a , b ,c ;
cin >> a >> b >> c ;
if ( a < b && a<c)
            {cout << "min is " << a ;}
 else if( b < a && b<c)
            {cout << "min is " << b ;}
    else if(c < a && c < b)
        {cout << "min is " << c ;}
          else   {cout<<" Two numbers are equally or more" ;}  // عددان متساويان أو أكثر
return 0 ;}
```

4. برنامج حل معادلة من الدرجة الأولى لها الشكل : ax+b=0 حيث a,b أعداد ثابتة .

```
# include <iostream.h>
main( )
{  int a,b;
   cin>>a>>b;
   cout<<a<<"X+"<<b<<"=0"<<"\n";
   if(a==0 && b!=0)
      {cout<<"Unsolvable";} // مستحيلة الحل
  else if(a==0 && b==0)
         {cout<<"State match";} //حالة تطابق
        else
           {cout<<"X="<<-1*b/a;}
return 0 ;}
```

5. اكتب برنامج لحساب المعادلة :

$$y = \begin{cases} x^2 + 1 & : x > 0 \\ x + 5 & : x = 0 \\ 2x^3 - 1 & : x < 0 \end{cases}$$

```
#include<iostream.h>
main( )
{int x,y;
 cout<<"Enter the x: ";
 cin>> x;
 if(x >0)
  {y=x*x+1;}
 else if(x = = 0)
       {y=x+5;}
     else
       {y=2*x*x*x-1;}
 cout<< y;
 return 0;
}
```

6. اكتب برنامج لحساب المعادلة :

$$y = \begin{cases} x^2 - 1 & : 10 < x < 100 \\ x^3 - 1 & : x > 100 \end{cases}$$

```
# include <iostream.h>
main( )
{
int x,y;
cout<<"Enter the x: ";
cin>> x;
if(x >10&&x<100)
{
     y=x*x-1;
     cout<< y;
}
else  if(x>100)
       {
          y=x*x*x-1;
        cout<< y;
        }
return 0;
}
```

7. اكتب برنامج لإدخال نتيجة طالب وطباعة تقديره .

```
# include <iostream.h>
 main( )
{
int grade;
cout<<"Enter the grade: ";
cin>>grade;
if(grade >=90)
{cout<< " : Excellent "<<endl;}     // ممتاز
else if(grade >=80)
     {cout<< " : Very Good "<<endl;}    //جيد جداً
        else  if(grade >=65)
        {cout<< " : Good "<<endl;}        //جيد
          else if(grade >=50)
              {cout<< " : Accepted "<<endl;}   //مقبول
                    else if(grade < 50)
                  {cout<< " : is failing  "<<endl;}  // راسب
return 0;
}
```

8. برنامج لإنجاز العمليات الحسابية (+,-,*,/)لعددين a,b على أن يتم إدخال رمز العملية والعددين من لوحة المفاتيح باستخدام البنية switch.

```
# include <iostream.h>
main( )
{
int x,y;
 char op;
 cout<<"Enter The Numbers :";
 cin>>x>>y;
 cout<<"Enter The Operation : ";
 cin>>op;
 switch(op)
 {case '+': cout<<x+y<<"\n";break;
 case '-': cout<<x-y<<"\n";break;
 case '*': cout<<x*y<<"\n";break;
 case '/': cout<<x/y<<"\n";break;
 default:cout<<"Error Operation\n";
 }
return 0 ;
}
```

الحلقات في لغة C++

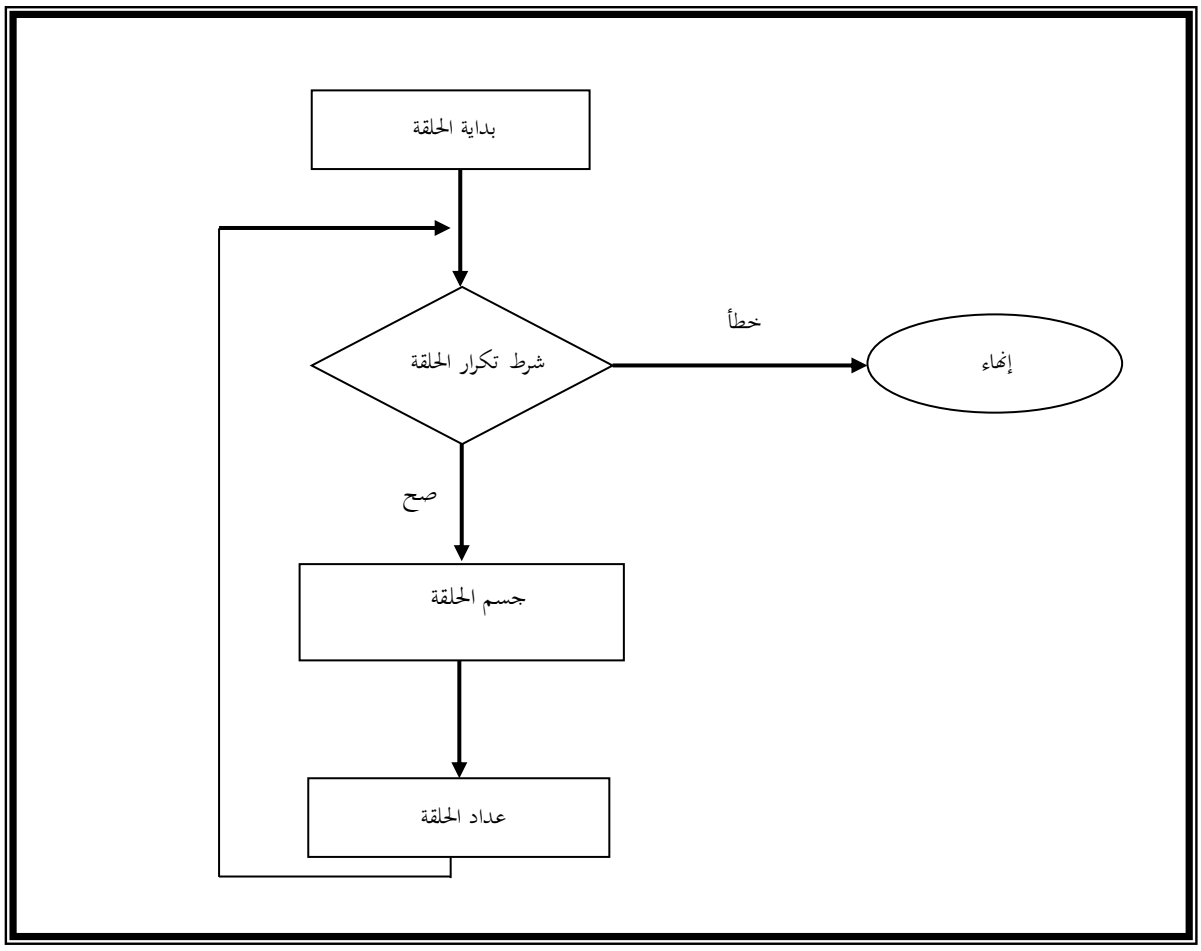
توفر C++ عدداً من أساليب التكرار (حلقات) التي تستخدم لتكرار أجزاء من البرنامج قدر ما تدعو الحاجة، لتحديد عدد مرات تكرار الحلقة تفحص كل حلقات C++ ما إذا كان تعبير ما يساوى صحيح (true) أو خطأ (false) يبلغها هذا ما إذا كان عليها التكرار مرة إضافية أخرى أو التوقف فوراً.
هنالك ثلاثة أنواع من الحلقات في C+ +:

1. الحلقة for

تأخذ الحلقة for الشكل العام التالي:

```
for( expression1; expression2; expression3)
    {statement;}
```

حيث يمثل:
expression1 : بداية الحلقة.
expression2 : شرط الحلقة الذي يفحص قيمة عداد الحلقة ويحدد ما إذا كان يجب تكرار الحلقة مرة أخرى أم لا.
expression3 : يمثل زيادة الحلقة الذي يقوم بزيادة أو إنقاص قيمة عداد الحلقة.
في الحلقة for يكون عدد مرات تنفيذ الحلقة مذكوراً عادة في بدايتها.
الشكل التالي يبين كيفية عمل الحلقة for.



طريقة عمل الحلقة for

مثال1:
المثال التالي يقوم بطباعة قيم المتغير i من 1 إلى 10 .

```
#include <iostream.h>
main( )
{
for ( int i = 1; i<= 10; i++)
{ cout << i <<"\n";}
return 0;
}
```

نتيجة التنفيذ:

```
1
2
3
4
5
6
7
8
9
10
```

تحتوي الأقواس التي تلي الكلمة الأساسية for على ثلاثة تعابير مختلفة تفصلها فاصلة منقوطة. تعمل هذه التعابير الثلاثة في أغلب الأوقات على متغير يدعى عداد الحلقة ،وهو المتغير i في المثال السابق.
هذه التعابير هي:
بداية الحلقة: يمهد قيمة عداد الحلقة عادة int i = 1; .
شرط الحلقة: يفحص قيمة العداد ليرى ما إذا كان يجب تكرار الحلقة مرة أخرى أوإيقافها i <=10;
زيادة الحلقة: يقوم عادة بزيادة (أو إنقاص) قيمة عداد الحلقة i++ .
المثال التالي يقوم بإنقاص عداد الحلقة بـ 1 كلما تكررت الحلقة :
مثال2:

```
#include <iostream.h>
 main ( )
 {
  for ( int j=10; j>0; j--)
   {cout <<j<<" ";}
 return 0;}
```

نتيجة التنفيذ: **10 9 8 7 6 5 4 3 2 1**

ويمكن أيضاً زيادة أو إنقاص عداد الحلقة بقيمة أخرى .
مثال 3:البرنامج التالي يوضح ذلك :

```
#include <iostream.h>
main ( )
{
for (int j=10; j<100; j+=10)
   { cout <<j<<"  ";}
return 0;}
```

نتيجة التنفيذ: 10 20 30 40 50 60 70 80 90

حلقات for المتداخلة:

تتكون الحلقات المتداخلة من حلقة خارجية وحلقة أخرى داخلية أو أكثر وفي كل مرة تتكرر الحلقة الخارجية يتم تكرار الحلقة الداخلية من بداية العداد إلى نهايته .

تأخذ الحلقات for المتداخلة الشكل العام التالي :

```
for (.....)
        for (.....)
  { statements;  }
```

مثال 4:

```
#include <iostream.h>
main( )
{  int i,j;
for (i=1 ; i<5;i++)
 {
    for (j=1 ; j<4;j++)
          {cout << i<<j<<" \t";}
    cout<<"\n";
 }
return 0; }
```

نلاحظ هنا أن الحلقة الداخلية تتكرر 3 مرات لكل قيمة من قيم i (عداد الحلقة الخارجية).

نتيجة التنفيذ:

11	12	13
21	22	23
31	32	33
41	42	43

يمكننا وضع أي نوع من الحلقات ضمن أي نوع آخر، ويمكن مداخلة الحلقات في حلقات متداخلة في حلقات أخرى وهكذا.

2. الحلقة while

تأخذ الحلقة while الشكل العام التالي:

```
while( condition)
  {statement}
```

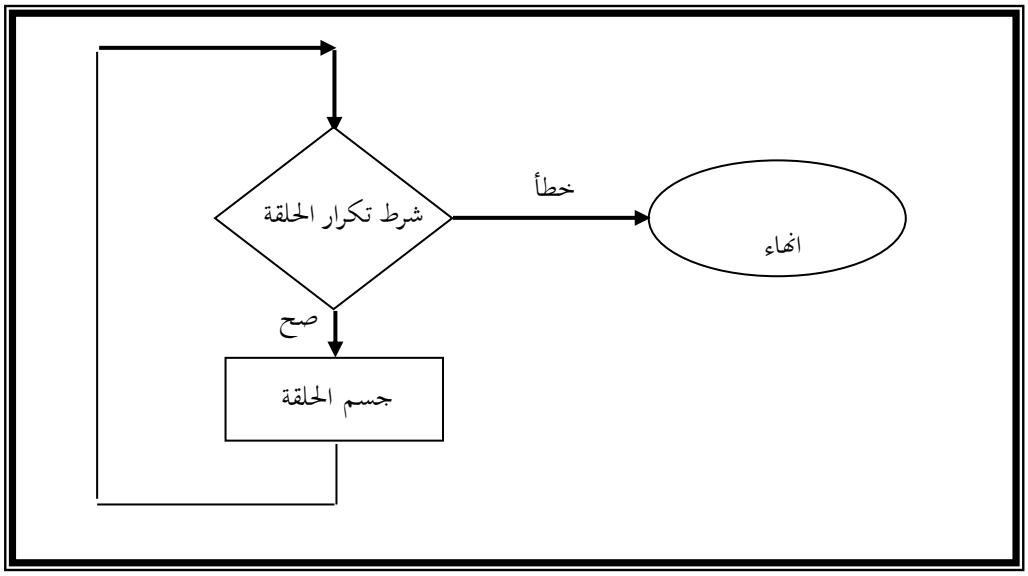
مثال 5 :

```
#include <iostream.h>
main( )
{  int n=3;
  while (n<30)
    { cout<<n<<"   ";
       n=n*2 ;
    }
return 0; }
```

والنتيجة هي :3 6 12 24

أي يقوم بطباعة القيم التي تأخذها n طالما هذه القيم أصغر تماماً من 30 أي ستستمر هذه الحلقة في مضاعفة المتغير n إلى أن تصبح قيمة n أكبر من 30 عندها تتوقف .

تتكون الحلقة من الكلمة الأساسية while يليها تعبير اختبار بين أقواس ويكون جسم الحلقة محصوراً بين أقواس حاصرة { } إلا إذا كان يتألف من عبارة واحدة. الشكل التالي يبين طريقة عمل الحلقة while:



طريقة عمل الحلقة while

مما يجدر التنويه إليه هنا أنه يتم فحص شرط الحلقة قبل تنفيذ جسم الحلقة، وعليه لن يتم تنفيذ جسم الحلقة أبداً إذا كان الشرط خطأ عند دخول الحلقة وعليه المتغير n في المثال السابق يجب تمهيده عند قيمة أقل من 100 .

مثال 6 :

برنامج حساب المتوسط الحسابي لعلامات 10 طلاب في امتحان .

```
#include<iostream.h>
 main ( ) {
 int i, grade, total ;
 float average;
 total = 0;
 i = 1;
 while (i <= 10)
  {  cin >>grade;
      total = total + grade;
      i = i + 1;
   }
 average = total /10;
 cout << " Class average is: " << average <<"\n";
 return 0; }
```

اذا كانت العلامات المدخلة كالتالي 75 65 51 89 71 54 80 79 81 90

فإن نتيجة التنفيذ:

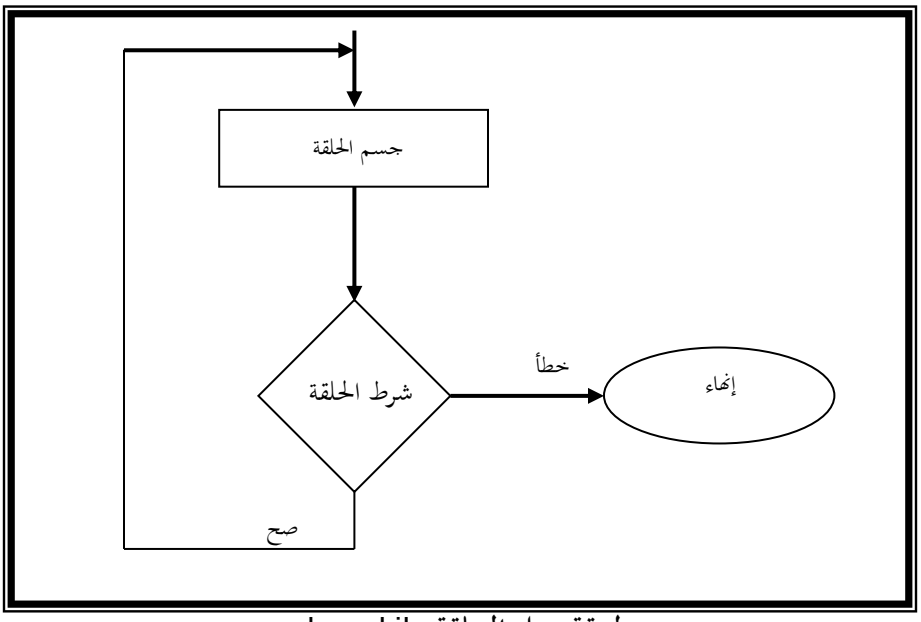
Class average is : 73.5

3. الحلقة do… while:

تعمل الحلقة do (غالباً تسمى ...do…while) كالحلقة while، إلا أنها تفحص شرط الحلقة بعد تنفيذ جسم الحلقة. وتستخدم أيضاً عندما نريد القيام بجزء من البرنامج مرة واحدة على الأقل.
الشكل التالي يبين كيفية عمل الحلقة do.
تأخذ الحلقة do الشكل التالي :

```
do
{statement;}
while( condition);
```

الحلقة do تفحص شرط الحلقة بعد تنفيذ جسم الحلقة ، وعليه يتم تكرار جسم الحلقة do مرة واحدة على الأقل حتى ولو كان الشرط غير متحقق وتفسير ذلك أن التحقق من الشرط يتم بعد التنفيذ وليس قبله كما في الحلقتين السابقتين..



طريقة عمل الحلقة do.. while

تبدأ الحلقة do بالكلمة الأساسية do يليها جسم الحلقة بين أقواس حاصرة { } ثم الكلمة الأساسية while ثم تعبير اختبار بين أقواس ثم فاصلة منقوطة.

مثال 6: البرنامج التالي يقوم بطباعة الأعداد من 1 إلى 10 .

```
#include <iostream.h>
main ( )
{ int i = 1;
 do {cout << i <<"  ";
   i++;
     } while (i <= 10);
return 0;}
```

تقوم ;" " >>cout بطباعة مسافة خالية بين كل رقم والآخر وعليه الخرج من البرنامج يكون كالتالي:

1 2 3 4 5 6 7 8 9 10

تمارين :

1. برنامج يطبع الأعداد الزوجية الموجودة في المجال [1..100] بشكل تصاعدي وبشكل تنازلي .

- بشكل تصاعدي : طريقة اولى

```
#include<iostream.h>
main()
{   for(int i=2;i<=100;i=i+2)
      { cout<<i<<"\t";}
   return 0;}
```

طريقة ثانية

```
#include<iostream.h>
main()
{
  for(int i=1;i<=100;i++)
  { if(i%2==0)
    { cout<<i<<"\t";}
  }
   return 0;}
```

- بشكل تنازلي

```
#include<iostream.h>
main()
{   for(int i=100;i>=1;i=i-2)
      { cout<<i<<"\t";}
   return 0;}
```

2. برنامج يحسب مجموع الأعداد من 1 الى 10 .

باستخدام **for**

```
#include<iostream.h>
main( )
{  int i,s=0;
   for (i=1;i<=10;i++)
     {s+=i; }
   cout<<s;
return 0; }
```

****

باستخدام **while**

```
#include<iostream.h>
main( )
{ int i=1,s=0;
  while(i<=10)
   {s=s+i;
     i++;}
  cout<<s;
  return 0; }
```

****

```
#include<iostream.h>
main( )
{ int i=1,s=0;
do  {s+=i;
       i++;
     }while(i<=10);
cout<<s;
return 0;}
```

do …..while باستخدام

3. برنامج لإدخال عشرة أحرف وطباعة حرف "A" إن وجد و كم مرة أدخل الحرف:

```
# include <iostream.h>
main( )
{
  char letter ;
  int  count=0;
    for (int i=1;i<=5;i++)
      {
          cin >> letter ;
              if(letter=='A')
                    {count++;}
      }
    cout<<"Letter A appears:"<<count<<"\n";
    return 0;
}
```

4. برنامج يطبع جدول الضرب للرقم المدخل فقط.

```
#include<iostream.h>
main( )
{
  int i,x;
 cout<<"Enter  the number : ";
 cin>> x;
   for (i=1;i<=10;i++)
      {cout<<x<<"*"<<i<<"="<<x*i<<"   ";
       cout<<"\n";
      }
return 0; }
```

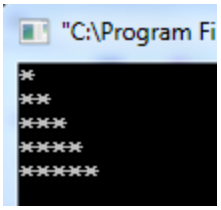
5. برنامج يطبع جدول الضرب الى العدد 10 .

```
#include<iostream.h>
main()
{
   for (int i=1;i<=10;i++)
       {
           cout<<"Multiply Table for "<<i<<"\n\n";
           for(int j=1;j<=10;j++)
                {
                    cout<<i<<"*"<<j<<"="<<i*j<<"\n";
                }
          cout<<"\n";
      }
return 0;}
```

6. برنامج يطبع الشكل التالي :

```
#include<iostream.h>
main()
{   int i,j;
   for (i=1;i<=5;i++)
    {
        for (j=1;j<=i;j++)
          {
           cout<<  "*";
          }

       cout<<"\n";
    }
cout<<"\n";
return 0; }
```



7. برنامج يعمل إدخال علامات عشرة طلاب وطباعة المعدل العام للطلاب العشرة:

```
#include <iostream.h>
 main()
{  float mark,sum ;
     for(int i=1;i<=10;i++)
       {
           cin >> mark ;
           sum = sum + mark ;
       }
     cout << " average is : " << sum / 10 ;
     return 0; }
```

8. طباعة مجموع الأعداد 3.5 , 4 , 4.5,……,9.5

```
#include<iostream.h>
main()
{float s=0,i;
  for(i=3.5;i<=9.5;i=i+0.5)
     { s=s+i;}
  cout<<s<<"\n";
 return 0;}
```

9. برنامج ايجاد قواسم عدد ما:

```
    # include <iostream.h>
     main( )
    { int x ;
      cout << " enter number:  " ;
      cin >> x;
      for ( int i=1 ;i<=x ; i++ )
        { if(x%i==0)
          {cout << i <<"\n";}
        }
      return 0 ; }
```

المصفوفات

1- تعريف المصفوفة:

المصفوفة هي نوع من أنواع بنية البيانات، لها عدد محدود ومرتب من العناصر التي تكون جميعها من نفس النوع type، فمثلاً يمكن أن تكون جميعها أعداد صحيحة int أو حقيقية float أو محارف char ولكن لا يمكن الجمع بين نوعين مختلفين في نفس المصفوفة .

وعند تعريف المصفوفة وإنشاؤها يتم حجز عدد محدد من المواقع المتجاورة في الذاكرة لتخزين البيانات فيها ،حيث يتم الوصول الى البيانات المخزنة في هذه المواقع عن طريق اسم المصفوفة ورقم الموقع (index) والغاية من استخدام المصفوفات هي تخزين عدد من القيم تحت اسم واحد فقط (يكون لها النمط نفسه) دون الحاجة الى تخزين كل قيمة في متحول منفصل .

2- ميزات المصفوفات :

أ- تقليل حجم البرنامج .
ب- سهولة اسناد القيم واسترجاعها.
ت- استخدام تقنيات البحث والترتيب .
ث- الوصول المباشر إلى البيانات المخزنة فيها.

3- عيوب المصفوفات :

أ- لا يمكن تحديد حجمها أثناء التنفيذ(فقط يمكن تحديد حجمها عند التصريح عنها).
ب- جميع عناصرها يجب أن تكون من نوع واحد للبيانات (أي لا تحتوي على عناصر مختلفة من انواع البيانات).

4- أنواع المصفوفات :

أولاً .. المصفوفات أحادية البعد :

وتكون عبارة عن صف واحد من العناصر .

وهي تتكون من مواقع متجاورة لتخزين بيانات عددية فيها على شكل صف واحد ولكل موقع (دليل index) رقم مخصص له ويتم ترقيم هذه المواقع (الأدلة) بالتتالي (0,1,2,……,n-1) (n طول المصفوفة أي عدد عناصرها).

لتعريفها وحجز موقع لها :

```
Data Type     name of array [n];
```

حيث:

Data Type :نوع بيانات المصفوفة .

name of array : اسم للمصفوفة .

n : عدد عناصر المصفوفة .

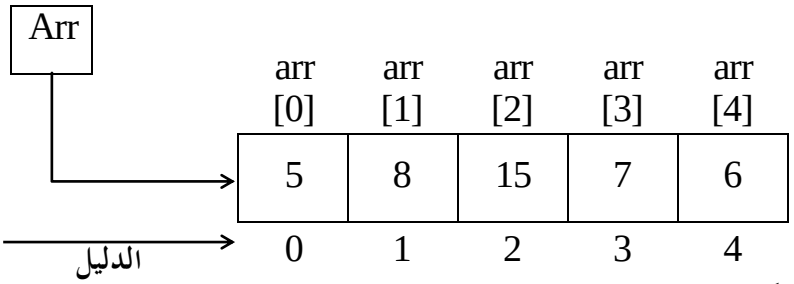
مثال :مصفوفة أعداد صحيحة عدد عناصرها 5 .

```
int  arr [ 5];
```

أما لإعطاء قيم ابتدائية لهذه المصفوفة:

```
arr [0]=5;
arr [1]=8;
arr [2]=15;
arr [3]=7;
arr [4]=6;
```

حيث أن دليل أول عنصر في المصفوفة هو 0 أما آخر عنصر فدليله n-1 .



أما لطباعة عنصر معين مثلاً العنصر الذي دليله 2 من المصفوفة السابقة يكون كالآتي: cout<<arr[2];

ولتعديل القيمة المخزنة في الموقع 1 لتصبح 18 بدل من 8 نكتب: arr [1]=18;

ولإدخال عناصر المصفوفة arr من لوحة المفاتيح:

```
cin>>arr[0];
cin>>arr[1];
cin>>arr[2];
cin>>arr[3];
cin>>arr[4];
```

وبالتالي يمكن استخدام حلقة for يبدأ العداد فيها من الـ 0 دليل أول عنصر وينتهي بأصغر تماماً من n عدد عناصر المصفوفة.

```
for(int i=0;i<5;i++)
    {cin>>arr[i];}
```

وكذلك الأمر من أجل طباعة عناصر مصفوفة نستخدم حلقة for:

```
for(int i=0;i<5;i++)
    {cout<<arr[i];}
```

- أمثلة عن طريق إعطاء قيم ابتدائية لعناصر المصفوفة:

1. يمكن إعطاء قيمة الصفر لكل عناصر المصفوفة على الشكل التالي:
   مثلاً int A[10]={0};
2. يمكن تحديد القيم الابتدائية للمصفوفة أثناء التصريح عنها بالطريقة التالية:

```
int  arr [5]={5,8,15,7,6};
```

3. يمكن إعطاء قيم ابتدائية لعناصر المصفوفة كقيمة معينة واحدة لكل العناصر بالشكل التالي:

```
# include <iostream.h>
main( )
{  int A[5] ;
    for (int I =0 ; I <5 ; I ++ )
        {A[I ] = 5 ;}
return 0 ;}
```

ملاحظات:

1. يسبب التصريح التالي: int n[5] = {1 , 2 , 34 , 56 , 24 , 14} ; خطأً قواعدياً لأننا أعطينا ستة قيم لمصفوفة مؤلفة من خمسة عناصر فقط.
2. يسبب التصريح التالي: int n[5] = {1 , 2 , 4 , 6 , 2 } ; إعطاء قيمة الصفر للعنصر الخامس من قبل المترجم.
3. إذا تم حذف حجم المصفوفة أثناء التصريح عنها فإن عدد عناصر هذه المصفوفة يصبح مساوياً لعدد القيم الابتدائية المعطاة ضمن القائمة الملحقة بالتصريح .لذلك يقوم التصريح التالي:

```
int n[ ] = { 1,2,3,4,5,6 } ;
```

بإنشاء مصفوفة مؤلفة من ستة عناصر.

مثال 1 :
اكتب برنامج لإدخال مصفوفة أعداد صحيحة عدد عناصرها 8 من لوحة المفاتيح ومن ثم طباعة هذه المصفوفة بالشكل النظامي وأيضاً طباعتها بالشكل العكسي (معكوس المصفوفة أي ابتداءً من آخر عنصر الى أول عنصر).

```
#include<iostream.h>
main( )
{int a[8];
  for(int i=0;i<8;i++)
       {cin>>a[i];}
 for(int i=0;i<8;i++)
       {cout<<a[i]<<"\t";}
 cout<<"\n";
 for(int i=7;i>=0;i--)
       {cout<<a[i]<<"\t";}
return 0;
}
```

ملاحظة :
تحدثنا في المحاضرة الأولى عن المتحول الثابت وهو المتحول الذي لا يمكن تغيير قيمته بعد التصريح عنه .
ويمكن استخدام المتحول الثابت في تحديد حجم المصفوفة .

```
const int size =10 ;
int s [size] ;
```

تفيد التعليمات السابقة في تحديد حجم مصفوفة s باستخدام الثابت size .
ويفيد استخدام المتحولات الثابتة لتحديد حجم المصفوفات في جعل البرامج اكثر قابلية لتغيير الحجم. فمثلاً حلقة for تقوم بتعبئة 10 عناصر يمكن تعديلها لتقوم بتعبئة 1000 عنصر وذلك بتغيير قيمة الثابت المرتبطة به أما في حالة عدم استخدام الثوابت فيتطلب التعديل السابق عدة تعديلات في اماكن مختلفة من البرنامج .

تمارين :
1. اكتب برنامج لإدخال مصفوفة أعداد صحيحة طولها 10 و طباعة الأعداد الزوجية فقط الموجودة في المصفوفة

```
# include <iostream.h>
main( )
{int  num[10];
for(int i=0;i<10;i++)
{cin>>num[i];}
for(int i=0;i<10;i++)
  {
       if(num[i]%2==0)
            {cout<<num[i]<<"\n";}
   }
return 0 ;
}
```

2. اكتب برنامج لإدخال مصفوفة أعداد صحيحة عدد عناصرها 10 ومن ثم طباعة مجموع أعداد هذه المصفوفة .

```
# include <iostream.h>
main( )
{
  int  num[10];
  int s=0;
  for(int i=0;i<10;i++)
    {
       cin>>num[i];
       s=s+num[i];
    }
 cout<<"The Sum:"<<s;
return 0;}
```

3. اكتب برنامج جدول الضرب للعدد 4 وتخزينه في مصفوفة وطباعة قيم الجدول هذا(أي طباعة المصفوفة) .

```
# include <iostream.h>
main()
{
int  mult[11];
for(int i=0;i<11;i++)
   {
        mult[i]=4*i;
        cout<<"4 *"<<i<<"="<<mult[i]<< "\n";
   }
return 0;}
```

4. لدينا المصفوفتان الأحاديتان a[ ]={2,4,1,7} b[ ]={3,1,7,8}

اكتب برنامج لتصريح وطباعة ناتج جمع هاتين المصفوفتين

```
# include <iostream.h>
main( )
{
int a[ ]={2,4,1,7};
int b[ ]={3,1,7,8};
int c[4];
   for(int i=0;i<4;i++)
      {
        c[i]=a[i]+b[i];
        cout<<"c["<<i<<"]="<<c[i]<<endl;
      }
return 0 ;
}
```

5. اكتب برنامج لتخزين مربعات الأعداد من 0 إلى 10 في مصفوفة وطباعتها .

```
# include <iostream.h>
main()
{int  sqr[11];
 for(int i=0;i<=10;i++)
      {
           sqr[i]=i*i;
           cout<<"Square "<<i<<"="<<sqr[i]<< "\n";
      }
return 0;}
```

6. اكتب برنامج لإدخال مصفوفة أعداد صحيحة طولها 15 ومن ثم طباعة أكبر عنصر وأصغر عنصر فيها .

```
#include <iostream.h>
main ( )
{   int x[15] , i, max, min ;
   for (i=0 ; i<15 ; i++)
       {cin >> x[i] ;  }
   max=x[0];
   min=x[0];
    for (i=0 ; i<15 ; i++)
      {
          if(x[i]>max)      {max=x[i];}
          if(x[i]<min)      {min=x[i];}
      }
cout<<"The max :"<<max <<"The min:"<<min;
return 0; }
```

7. اكتب برنامج لإنشاء مصفوفة أعداد صحيحة عدد عناصرها 7 قيم هذه المصفوفة عبارة عن العدد 1 اذا كان الدليل عدد فردي والعدد 0 اذا كان الدليل عدد زوجي .

```
#include <iostream.h>
 main ( )
 {   int num[15] ;
     for (int i=0 ; i<7 ; i++)
        {
          if(i%2!=0)
               {num[i]=1;}
           else
               {num[i]=0;}
        }
     for (int i=0 ; i<7 ; i++)
           {cout<<num[i]<< "\t";}
   return 0;
 }
```

مهارات حاسوب ٢

Computer Skills 2

صياغة البرامج ولغات البرمجة

Programming & programming languages

المحاضرة الأولى

صياغة البرامج ولغات البرمجة

بعد أن تكلمنا سابقا عن برامج نظم التشغيل و برامج التطبيقات يبقى سؤال

من كتب هذه البرامج؟

وكيف كتبت؟

وبأي لغة؟

وما هي الخطوات التي اتبعت لبناء هذه البرامج؟

صياغة البرامج ولغات البرمجة

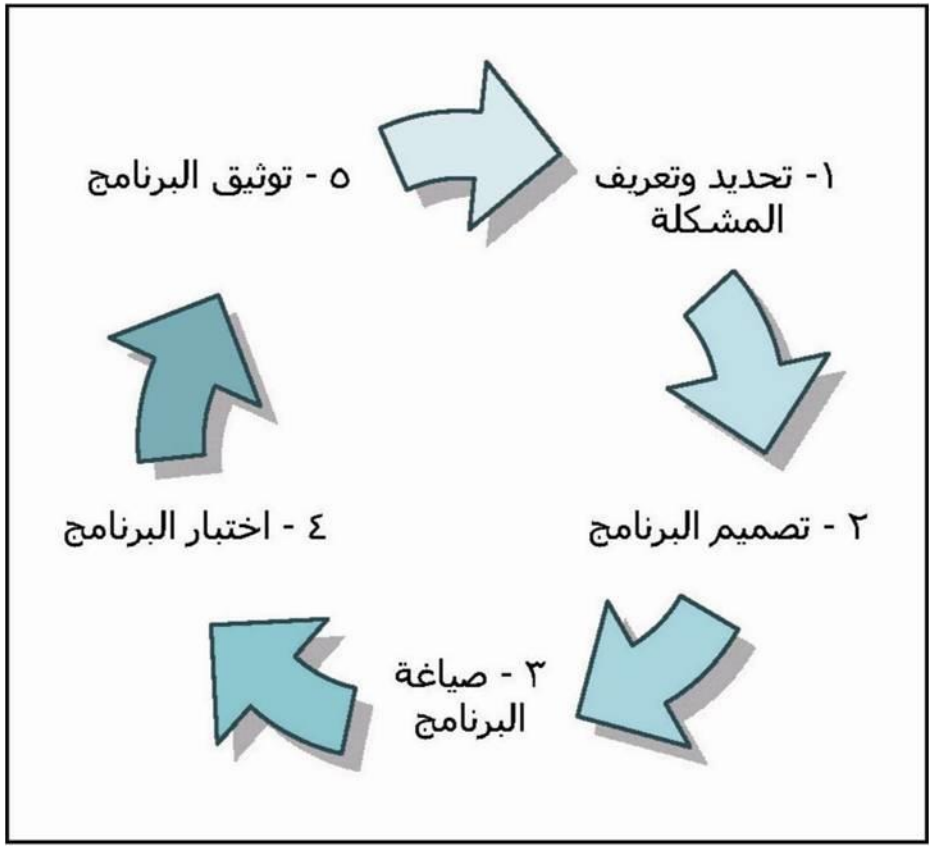
- **البرنامج Program** هو مجموعة التعليمات المرتبة منطقياً التي توجه الحاسوب لأداء عمل معين على البيانات بهدف الحصول على معلومات مفيدة.
- **لغات البرمجة Programming Languages** هي مجموعة القواعد التي توفر طريقة صياغة تعليمات البرنامج.

عمل صائغ البرامج Programmer

يقوم صائغ البرامج (المبرمج) بعدة خطوات لحل مشكلة ما، فيقوم

- أولاً بصياغة المشكلة في صورة تعليمات للحاسوب لحلها،
- ثم ينفذ التعليمات على الحاسوب ويختبر البرنامج لمعرفة مدى صحة النتائج،
- وفي النهاية يكتب تقريراً عن البرنامج.

خطوات صياغة وتطوير البرامج
Program Development Steps



مخطط يبين خطوات صياغة وتطوير البرامج

خطوات صياغة وتطوير البرامج
Program Development Steps

١. تحديد وتعريف المشكلة (Defining the Problem)

- في هذه الخطوة يقوم المبرمج بتحديد وتعريف المشكلة وتتضمن هذه الخطوة تحديد التالي بالترتيب:
  - الهدف من البرنامج (حساب ارباح، فواتير استهلاك الماء والكهرباء، أو حساب معدل الطالب التراكمي)
  - نوع وحجم المخرجات ووسائل الإخراج (تقارير – فواتير – شيكات – نقود ...)
  - نوع وحجم البيانات المدخلة ووسائل الإدخال.
  - مستخدمي البرامج والمستفيدين منه.

خطوات صياغة وتطوير البرامج
Program Development Steps

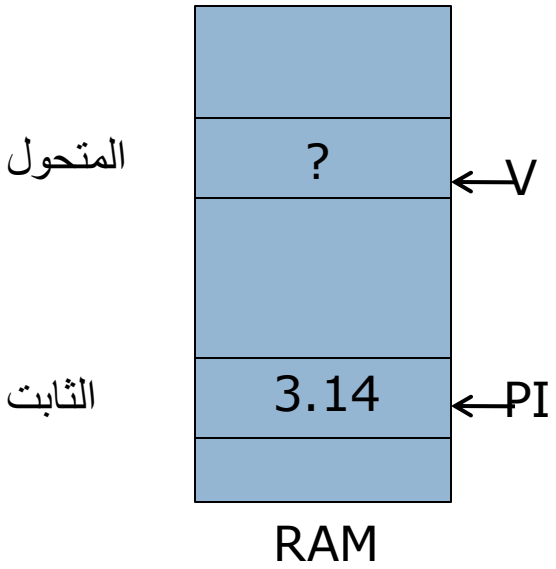
٢. تصميم البرنامج Design the Program

- يتم هنا تحديد المواصفات والخطوات الدقيقة والمرتبة منطقيا والتي تم فهمها ودراستها في الخطوة الأولى.
- ويتم ذلك بإستخدام عدة طرق منها:
  - الطريقة النصية: **لغة الخوارزمية Pseudo–code شبه الترميز**
  - الطريقة البيانية: **خرائط التدفق Flowchart** ويطلق عليها أيضاً خرائط سير العمليات وهي مجموعة من الرموز المتعارف عليها تستخدم لتوضيح الخطوات المنطقية اللازمة لحل مشكلة ما.

خطوات صياغة وتطوير البرامج
Program Development Steps

❖ ***الطريقة النصية: لغة الخوارزمية Pseudo-code شبه الترميز.***

- المتحول Variable غرض تتغير قيمته ضمن الخوارزمية
- الثابت Constantغرض لا تتغير قيمته ضمن الخوارزمية
- الصيغة تتألف من متحولات وثوابت وعمليات حسابية أو منطقية
- الشرط :صيغة نتيجتها صح أو خطأ



الصيغة $X + 4 * Y$

الشرط $X > 0$

خطوات صياغة وتطوير البرامج
Program Development Steps

لغة الخوارزمية Pseudo–code شبه الترميز.

التعليمات :

١. تعليمة القراءة
٢. تعليمة الكتابة
٣. تعليمة الإسناد
٤. التعليمة الشرطية
٥. التعليمة التكرارية

خطوات صياغة وتطوير البرامج
Program Development Steps

التعليمات :

١. تعليمة القراءة

اقرأ < اسم المتحول> : خذ قيمة واحدة من الدخل (لوحة المفاتيح) وضعها في الذاكرة المسماة بـ < اسم المتحول >

أمثلة:

اقرأ V ضع القيمة المدخلة في الذاكرة V.

اقرأ A,B ضع القيمتين المدخلتين في الذاكرتين A,B على الترتيب.

خطوات صياغة وتطوير البرامج
Program Development Steps

التعليمات :

٢. تعليمة الكتابة

كتابة قيمة معينة على وحدة الخرج (الشاشة) اكتب <صيغة>

أمثلة:

اكتب "The result is"

اكتب V+3

اكتب "The result is", V

خطوات صياغة وتطوير البرامج
Program Development Steps

التعليمات :

٣. تعليمة الإسناد

تعليمة الإسناد:<صيغة>=<اسم المتحول>

أمثلة:

A=70; B=A

حساب قيمة التابع $y=x^2+5x+2$ من أجل x=4

حساب قيمة التابع $y=x^2+5x+2$ من أجل x قيمة مدخلة من قبل المستثمر.

اقرأ x

$y=x^2+5x+2$

اكتب "x=",x,"y=",y

خطوات صياغة وتطوير البرامج
Program Development Steps

التعليمات :

٤. التعليمة الشرطية

أ- إذا <شرط> نفذ

<مجموعة التعليمات>

مثال

إذا (N>0) نفذ

a=S/N

ب- إذا <شرط> نفذ

<مجموعة التعليمات١>

وإلا

<مجموعة التعليمات٢>

مثال: نريد أن يقوم المستثمر بإدخال قيمة ما ، يحدد البرنامج هل القيمة تقع ضمن المجال [0,10] أم لا.

اقرأ a

إذا (a<=10) and (a>=0) نفذ

اكتب "a is inside the interval [0,10]"

وإلا

اكتب "a is outside the interval [0,10]"

خطوات صياغة وتطوير البرامج
Program Development Steps

التعليمات :

٥. التعليمة التكرارية

مادام <شرط> كرر

< مجموعة التعليمات>

اختبر الشرط إذا كان محققاً ننفذ < مجموعة التعليمات> ثم نختبر من جديد إذا كان محققاً ننفذ < مجموعة التعليمات> ... عندما يصبح الشرط غير محققاً ننتقل إلى التعليمة التالية لـ مادام

ملاحظة: يجب أن تغير < مجموعة التعليمات> متحولات الشرط ليصبح الشرط فيما بعد غير محققاً وإلا سندخل في حلقة لا منتهية .

خطوات صياغة وتطوير البرامج
Program Development Steps

مثال ١

- حساب مجموع الأعداد من 1 إلى L حيث L قيمة مدخلة من قبل المستثمر.
  - المعطيات L
  - الخرج S= 1+2+...+L
- **الخوارزمية:**

S=0 المجموع
i=1 عدد عمليات الجمع
اقرأ L
مادام (i<=L) كرر
s=s+i
i=i+1
اكتب s,"the sum is"

خطوات صياغة وتطوير البرامج
Program Development Steps

مثال ٢

- نريد حساب مربعات الأعداد من 10 إلى 25 ، الخرج جدول يحوي الأعداد ومربعاتها مسبوقاً بترويسة

```
اكتب "قيمة x      قيمة y"
x=10
مادام (x<=25) كرر
    y=x*x
    اكتب x," ",y
    x=x+1
```

خطوات صياغة وتطوير البرامج
Program Development Steps

مثال ٣

- حساب القاسم المشترك الأعظم GCD لعددين مدخلين.
- الحل:

١٥ 20 القاسم المشترك الأعظم =٥
15 5
10 5
٥ 5 القاسم المشترك الأعظم

- **الخوارزمية:**
  - قراءة العددين A,B
  - إيجاد القاسم المشترك الأعظم
  - كتابة النتيجة

خطوات صياغة وتطوير البرامج
Program Development Steps

اقرأ A,B
مادام (A≠B) كرر
إذا(A>B) نفذ
A=A-B
وإلا
B=B-A
GCD=A
اكتب GCD, "The Greater Common Divider ="

تمارين

- تمرين ١ : محيط دائرة نصف قطرها 5cm
- تمرين 2: حساب محيط ومساحة مستطيل، طوله و عرضه يتم إدخالهما من قبل المستثمر.
- تمرين ٣: حساب العاملة لعدد صحيح n حيث n قيمة مدخلة

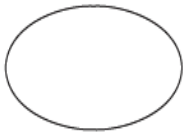
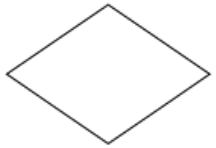
$$n!=n*(n-1)*(n-2)*....2*1$$

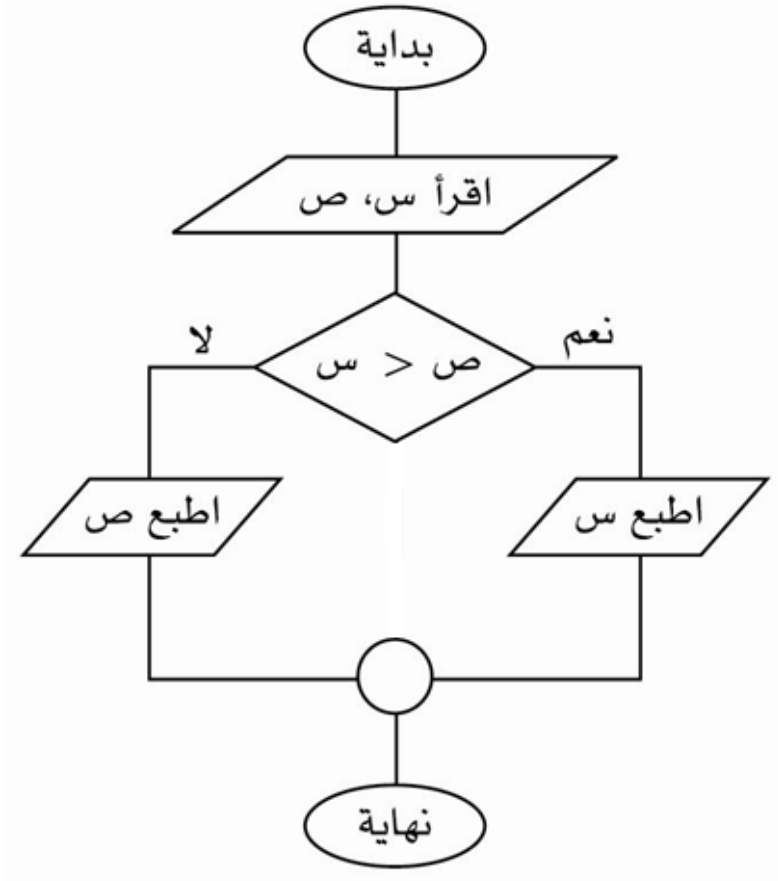
- تمرين ٤:حساب مجموع الأعداد الزوجية من 1 إلى L حيث L قيمة مدخلة من قبل المستثمر.
- تمرين ٥: نريد حساب مكعبات الأعداد من ٥ إلى ٣٥، الخرج جدول يحوي الأعداد ومكعباتها مسبوقاً بترويسة

خطوات صياغة وتطوير البرامج
Program Development Steps

❖الطريقة البيانية:مخطط التدفق flowchart

أهم الرموز المستخدمة في خرائط التدفق

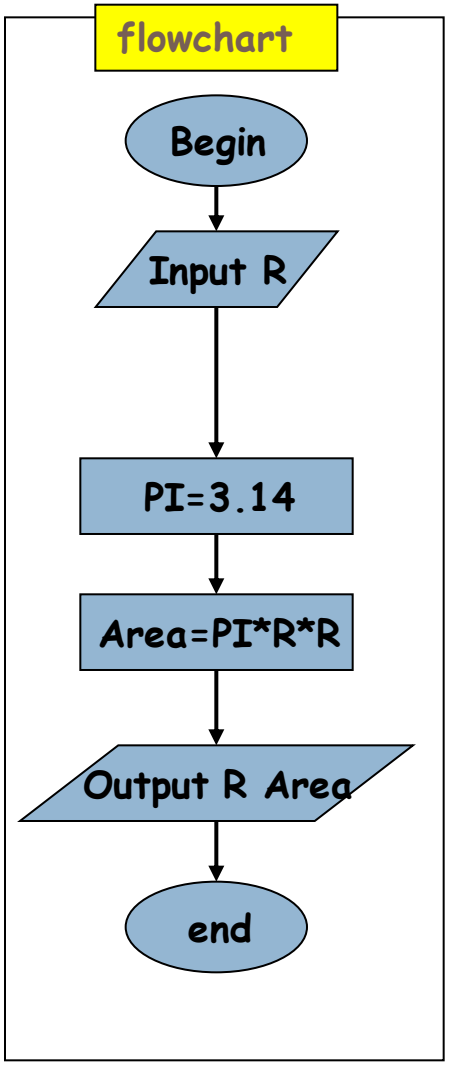
الاسم	الرمز
بداية – نهاية Start - End	
مدخلات – مخرجات Input - Output	
معالجة Processing	
قرار Decision	



الأسهم تربط بين الأشكال السابقة

اتجاه الأسهم يدل على طريقة تنفيذ التعليمات

حساب مساحة دائرة نصف قطرها R ، حيث R قيمة مدخلة من قبل المستثمر



لغة الخوارزمية

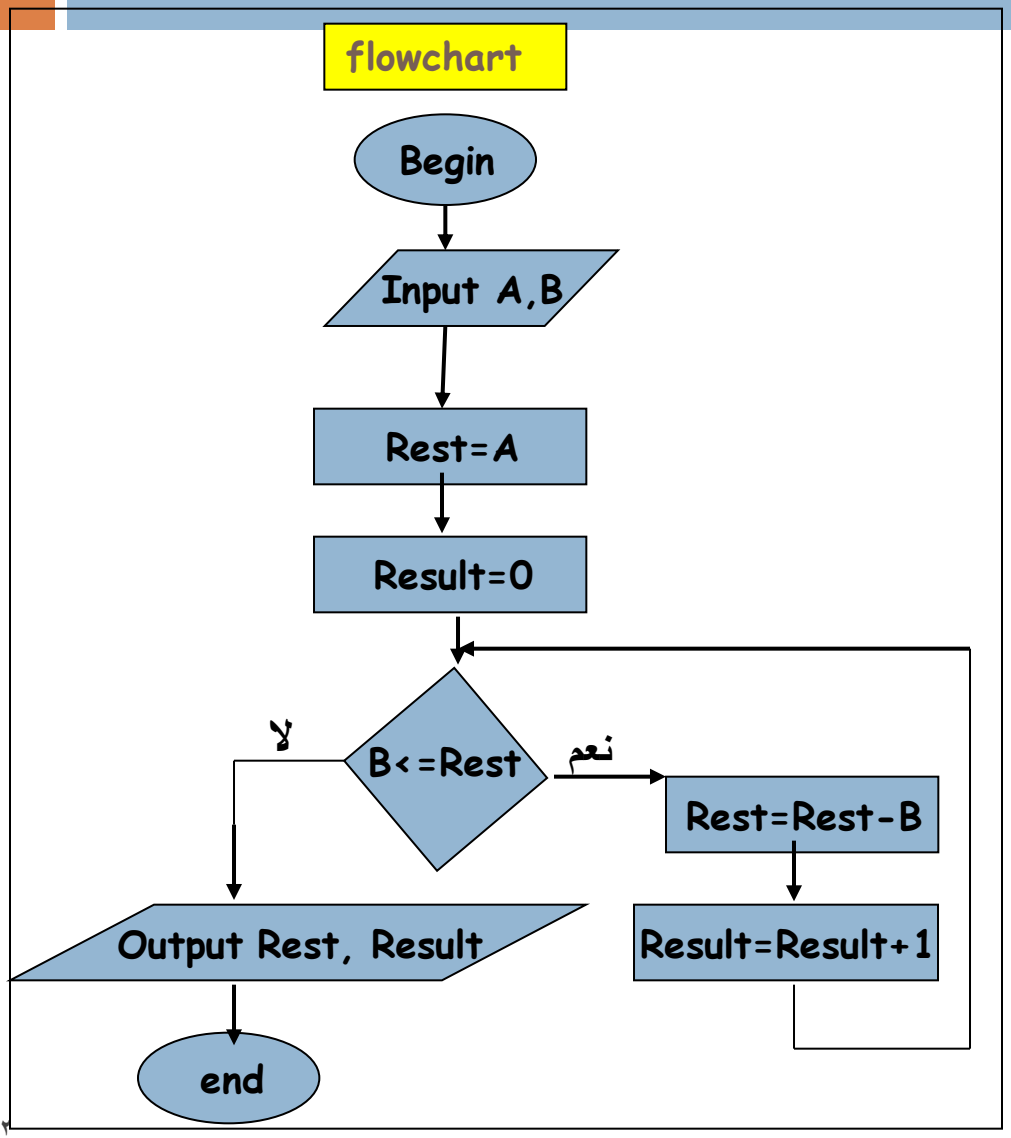
اقرأ R

Pi=3.14

area=Pi*R*R

اكتب area , R

ناتج وباقي قسمة عددين صحيحين مدخلين من قبل المستثمر



لغة الخوارزمية

اقرأ A,B

Rest=A

Result=0

مادام (B<=Rest) كرر

Rest=Rest-B

Result=Result+1

اكتب Result , Rest

خطوات صياغة وتطوير البرامج
Program Development Steps

٣. صياغة البرنامج Coding the Program

- بعد الانتهاء من تصميم البرنامج يتم اختيار إحدى لغات البرمجة المناسبة لصياغة أوامر البرنامج Coding وذلك بالاستعانة بخريطة التدفق Flow Chart أو غيرها.
- يجب عند صياغة البرنامج اتباع قواعد صيانة لغة البرمجة المستخدمة حيث ان لكل لغة برمجة قواعد خاصة بها ولا يعمل البرنامج اذا كان هنالك اخطاء املائية او اخطاء في قواعد اللغة Syntax Errors.

خطوات صياغة وتطوير البرامج
Program Development Steps

٤. اختبار البرنامج وتصحيح الأخطاء Program Debugging and Testing

- خلال عملية الترجمة Compilation قد تظهر اخطاء في صياغة البرنامج المصدر ينبغي على المبرمج تصحيحها.
- هناك ثلاث انواع من الأخطاء:

١. **اخطاء في قواعد اللغة Syntax Errors:** اخطاء املائية في كتابة الأوامر.

٢. **خطاء منطقية Logical Errors:** لا يكتشفها الحاسوب وتظهر عند تنفيذ البرنامج على عينه من البيانات فنحصل على نتائج خاطئه او غير متوقعة، ويقوم المبرمج بتتبع خطوات البرنامج لمعرفة مصدر الخطأ وتصحيحه وتسمى هذه العملية Tracing.

٣. **اخطاء اثناء التشغيل Run-Time Errors:** تظهر عند تنفيذ البرنامج مثل عدم حجز مساحة كافية للمدخلات او الدخول في دوران بلا نهاية، وتظهر رسالة بنوع الخطاء.

خطوات صياغة وتطوير البرامج
Program Development Steps

٥. توثيق البرنامج Documenting the Program

- في هذه المرحلة تتم كتابة وصف تفصيلي لصياغة البرنامج، ويشمل هذا التوثيق أصل المشكلة وخطوات الحل وخرائط الحل وتعليمات التشغيل ومتطلبات التشغيل والمدخلات والمخرجات وكيفية التحكم في البرنامج في المواقف المختلفة

تصنيف لغات البرمجة

تصنف لغات البرمجة إلى ثلاثة أنواع هي:

١. لغات برمجة ذات مستوى منخفض Low Level Languages

٢. لغات برمجة ذات مستوى عال High Level Languages

٣. لغات الجيل الرابع Fourth Generation Languages

تصنيف لغات البرمجة

١. لغات البرمجة ذات المستوى المنخفض Low Level Languages

- تعتبر لغات البرمجة ذات المستوى المنخفض من أوائل لغات البرمجة ومنها:
  - لغة الآلة Machine Language
  - لغة التجميع Assembly language
- ميت باللغات المنخفضة المستوى نظراً لأن المبرمجين يكتبون أوامر البرنامج بمستوى قريب من مستوى فهم الآلة (الحاسوب)، حيث تستخدم هذه اللغة (٠ ، ١) في كتابة البرامج.

تصنيف لغات البرمجة

٢. لغات البرمجة ذات المستوى العالي High Level Languages

- سميت بهذا الاسم لأنه اصبح بامكان المبرمج كتابة البرامج دون معرفة تفاصيل كيفية قيام الحاسب بهذه العمليات، كمواقع التخزين و تفاصيل الجهاز الدقيقه.
- تعبيرات اللغات ذات المستوى العالي شبيهه الى درجة كبيرة باللغة التي يستخدمها الانسان في التخاطب و التواصل مع الاخرين
- تتميز بسهولة اكتشاف الاخطاء و تصحيحها و يمكن تشغيلها على اكثر من جهاز. كما يمكن استخدام أكثر من لغة برمجه على جهاز واحد.

تصنيف لغات البرمجة

٣. لغات الجيل الرابع Fourth Generation Languages

- تسمى هذه اللغات أيضاً باللغات عالية المستوى بصورة كبيرة جداً Very High Level Languages حيث إنها لغات سهلة الاستخدام والفهم وقريبة جداً من لغة الإنسان.
- يستطيع المبرمج القيام بكثير من العمليات بسهولة تغنيه عن صياغة Coding صفحات عديدة من أوامر البرنامج. ويهتم المبرمج بماذا يريده من الكومبيوتر دون ان يوجهه بكيفية القيام بذلك.
- من لغات الجيل الرابع: SQL , DBase

أنواع لغات البرمجة

١. لغة البيسك BASIC Language ولغة فيجوال بيسك Visual Basic

٢. لغة الجافا Java Language

٣. لغة الكوبل COBOL Language

٤. لغة الباسكال PASCAL Language

٥. لغة اللوجو LOGO Language

٦. لغات الذكاء الاصطناعي Artificial Intelligence Languages

٧. لغة سي ولغة سي بلس بلس C & C++ Language

أنواع لغات البرمجة

١. لغة البيسك BASIC Language ولغة فيجوال بيسك Visual Basic

- وهي لغات بسيطة عامة الأغراض وسهلة التعلم ويستخدمها المبتدئون في جميع الأعمال، وخاصة في التطبيقات العلمية، وهي اختصار للمعنى Beginners All-Purpose Symbolic Instruction Code.

- لبساطة هذه اللغة واستخدامها في التعليم ظهرت لها عدة إصدارات منها:
  - BASICA - GWBASIC - TURBO BASIC - QUICK BASIC.

- كما ظهرت أيضاً لغة فيجوال بيسك (البيسك المرئي) Visual Basic وهي لغة برمجة مرئية وتعتبر لغة مطورة من لغة البيسك وهي خاصة لإنتاج برمجيات ذات قدرة عالية وتتناسب مع بيئة برنامج نظام ويندوز Windows.

أنواع لغات البرمجة

٢. لغة الجافا Java Language

- تعتبر لغة الجافا من اللغات عالية المستوى وتعرف بأنها من اللغات المرئية Visualوالشيئية Objects، وهي من اللغات العامة الأغراض والتي تستخدم لإنتاج برمجيات متنوعة.

- تشبه لغة الجافا لغة C++ إلا أنها تتسم بالسهولة.

أنواع لغات البرمجة

٣. لغة الكوبل COBOL Language

- تستخدم هذه اللغة بصفة رئيسية في الأعمال التجارية مثل البنوك والشركات.

- هي لغة واسعة الانتشار، وكلمة كوبل مشتقة من الكلمة

  Common **B**usiness **O**riented **L**anguage

- وبدأ ظهور هذه اللغة سنة ١٩٥٩ وقد أجري عليها عدة تعديلات لزيادة كفاءتها وكان آخرها سنة ١٩٧٤.

أنواع لغات البرمجة

٤. لغة الباسكال PASCAL Language

- سميت نسبة إلى العالم الفرنسي في علم الحاسوب Blaise Pascal و يرجع تاريخها إلى ١٩٧٣ وتستخدم للأغراض العامة وكلغة تعليمية.

- على الرغم من وضوح بنائها إلا أنها أصعب في التعلم من لغة البيسك.

- تعتبر لغة باسكال من لغات البرمجة الرئيسية التي تدرس لطلبة المدارس والكليات نظرا لوضوح السمات الأساسية لتخطيط البرامج البنائية بها Structured Programming.

أنواع لغات البرمجة

٥. لغة اللوجو LOGO Language

- هي لغة تطبيقات علمية تتميز ببساطة وسهولة تعلمها وقد صممت خصيصا ليستخدمها الأطفال فهي تشجع على الإستخدام المنطقي والتركيبي.

- تعتمد هذه اللغة على استخدام روبوت صغير يسمى بالسلحفاة Turtle من أجل ابراز استعمالاتها كتعلم الأفكار الحسابية مثل الزوايا والقياسات.

أنواع لغات البرمجة

٦. لغات الذكاء الاصطناعي Artificial Intelligence Languages

- هي لغات خاصة بإنتاج حاسبات ذكية تحاكي الإنسان في قدراته الحركية والبصرية والتحليل والاستنتاج واتخاذ القرارات بناء على نظم الخبرة التي ستغذي بها الحاسبات.

- من أهم هذه اللغات:
  - لغة برولوج Prolog: يطلق عليها اسم لغة البرمجة المنطقية **Pro**gramming in **Log**ic.
  - لغة ليسب Lisp: يطلق عليها اسم لغة برمجة القوائم **L**ist **P**rogramming Language.

أنواع لغات البرمجة

٧. لغة سي ولغة سي بلس بلس C & C++ Language

- تتميز هذه اللغة بالقوة والمرونة والقدرة على إنتاج برمجيات متعددة وذات كفاءة عالية.

- وقد ظهرت نسخة حديثة من لغة C ذات بيئة مرئية وهي لغة C++ تتميز بكونها لغة برمجة مرئية Visual.