جَامعة المَنارة
MANARA UNIVERSITY

كلية الهندسة المعلوماتية

مدخل إلى الخوارزميات والبرمجة
Introduction to Algorithms and Programming

ا. د. علي عمران سليمان

محاضرات الأسبوع السابع
الفصل الثاني 2022-2023

بنى التحكم

3-1- بنى التحكم	3-3-3- بنية الاختيار if \ else المتداخلة
3-2- المؤثرات العلائقية والمنطقية.	3-3-4- بنية الاختيار switch
3-3- أنواع بنى التحكم .	3-3-5- البنية التكرارية while (حلقة while)
أ- البنية التسلسلية :	3-3-6- البنية التكرارية do \ while
ب- بنية الاختيار :	3-3-7- البنية التكرارية for
ت- بنية الشرط:	3-3-8- الحلقات المتداخلة
3-3-1- بنية الاختيار if :	3-4- التعليمتان continue, break
3-3-2- بنية الاختيار if \else :	

المحاضرة من المراجع :

[1]- Deitel & Deitel, C++ How to Program, Pearson; 10th Edition (February 29, 2016)

[2] - د.علي سليمان, مدخل إلى الحاسوب والخوارزميات, جامعة تشرين 2005-2006

البنية التكرارية

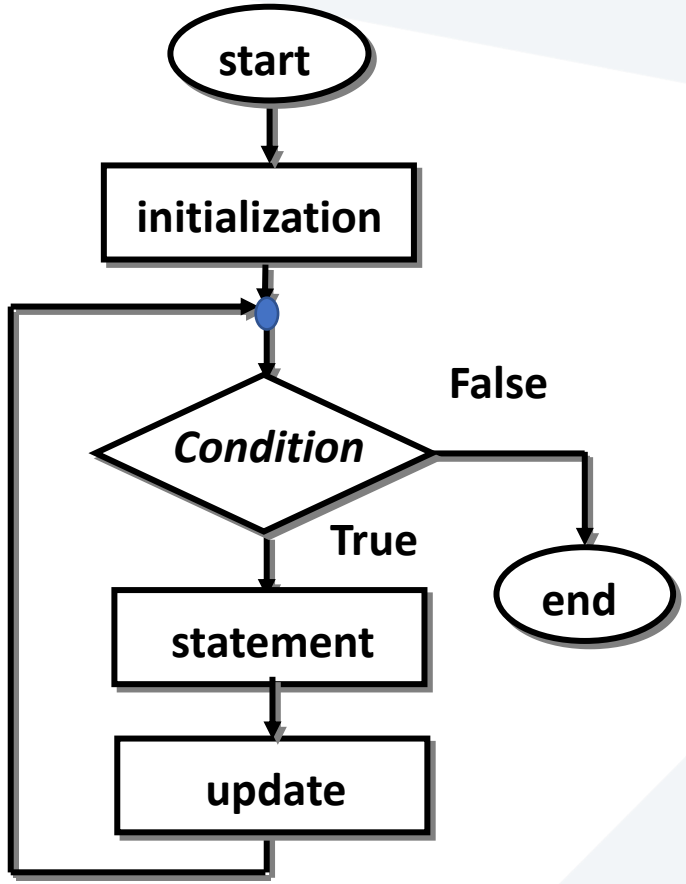
3-3-5- البنية التكرارية repetition structure

نحتاج في بعض المسائل إلى تكرار تعليمة أو عدة تعليمات عدد من المرات مما تطلب وجود بنى تكرارية.

- يوجد نوعين من البنى التكرارية ذات الشرط المسبق وذات الشرط التالي.
  - موقع الشرط يحدد نوعها:
    - إذا سبق الشرط الجسم كانت سابقة للشرط.
    - إذا تلى الشرط الجسم كانت لاحقة للشرط.
- تتكون الحلقة من الجسم الذي يعالج المهمة التي تقوم بها الحلقة وشرطأ ينهي أو يكرر الجسم.
- قد تملك الحلقة عداد يتم الاعتماد عليه في إختبار الشرط وبالتالي يجب أن يتغير كي يغير حالة الشرط، قد تكون لعدد من المرات معلوم سابقاً، وقد يتحقق أمر يؤدي إلى طلب إنهائها من جسمها.

5-3-3- البنية التكرارية while ذات الشرط السابق

The while repetition structure



تملك الصيغة العامة:

```
initialization;
while ( loopContinuationCondition ) {
    statement;
    update;
}
```

آلية التنفيذ :

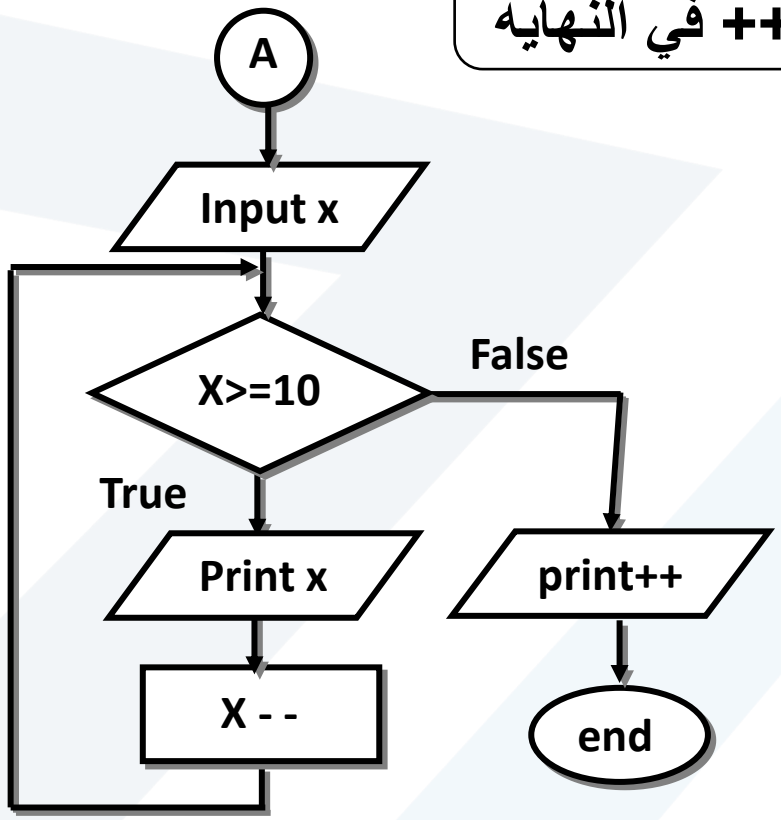
يتم التجهيز ***initialization*** بقيمة عادةً تكون متعلقة بالشرط، وقبل الجسم ولمرة واحده. يقيم شرط استمرار الحلقة ***loop Continuation Condition*** فإذا كان **true** يتم تنفيذ الجسم المكون من (***statement*** والذي قد يكون تعليمة مفردة أو مركبة، والتحديث ***update***) ثم يعاد تقيم قيمة ***Condition*** مرة أخرى، فإذا كان **true** يعاد تنفيذ الجسم مرة أخرى وهكذا تتكرر هاتان الخطوتان حتى يصبح قيمة ***Condition*** مساوية **false**، بمعنى أنه يتم تكرار تنفيذ الجسم طالما أن قيمة ***Condition*** مساوية **true** وعندما تصبح تلك القيمة **false** يتم الخروج من جسم الحلقة **while** والمتابعه بعد جسم الحلقة.

ملاحظة: قد يغيب ***initialization*** و ***update*** وتنهى الحلقة باسلوب آخر

طباعة الاعداد مابين

مثال16: طباعة الاعداد مابين العدد الصحيح المدخل الأكبر من 10 و10 ويطبع ++ في النهايه

```
void main()
{       int x;
cout<<"enter number:";
cin>>x;
while(x>=10)
      {cout<<x <<" ";  x--;}
cout<<"++"<<endl; }
```



Start
Input x
While x value great then or equal to ten
print x
decrement x by one
Print ++
End

إذا تم إدخال x = 13 ثم x=2 نجد

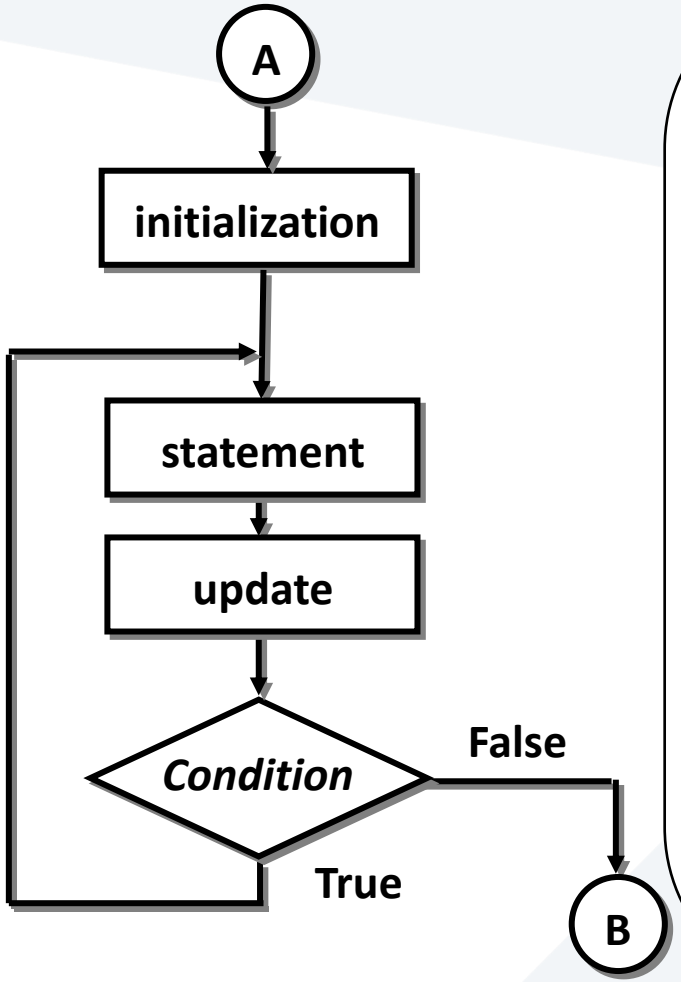
```
enter number : 13
13 12 11 10
++
enter number : 2
++
```

لعدم طباعة 10 يكتب الشرط x>10

سيتم حذف الاسطر ماقبل التابع الرئيس لعدم التكرار في ما يلي من التمارين

3-3-5- البنية التكرارية do while ذات الشرط الاحق

The do while repetition structure



تملك الصيغة العامة:

```
initialization;
do {
    statement;
    update;
} while ( loopContinuationCondition);
```

عند الرغبة بعرض قائمة قبل الادخال مثلاً

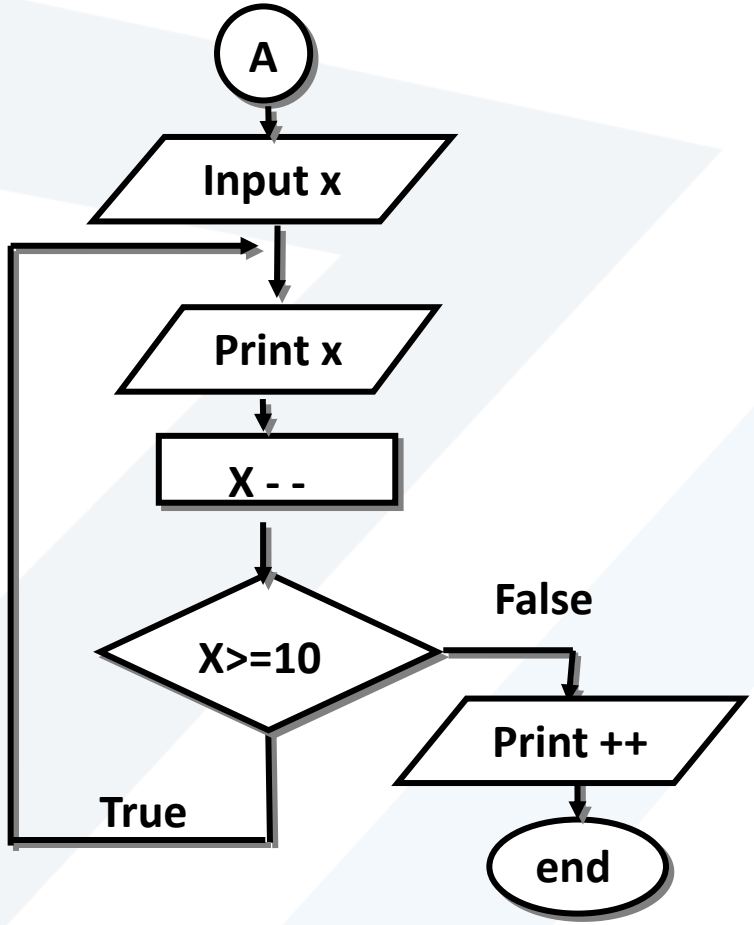
آلية التنفيذ :

يتم التجهيز ***initialization*** بقيمة عادةً تكون متعلقة بالشرط ولمره واحده فقط.
يتم تنفيذ الجسم (المؤلف من ***statement*** والذي قد يكون تعليمة مفردة أو مركبة والتعليمة ***update***)، ثم يقيم شرط استمرار الحلقة ***loop Continuation Condition*** فإذا كان **true** تتم العوده لتنفيذ جسم *الحلقة من جديد*، ثم يعاد تقدير قيمة ***Condition*** مرة أخرى، فإذا كان **true** يعاد تنفذ الجسم مرة أخرى وهكذا تتكرر هاتان الخطوتان حتى يصبح قيمة ***Condition*** مساوية **false**،عندها نتابع بعد جسم الحلقة.

ملاحظة: هذه الحلقة تنفذ مرة واحدة على الأقل لإن الشرط بعد الجسم.

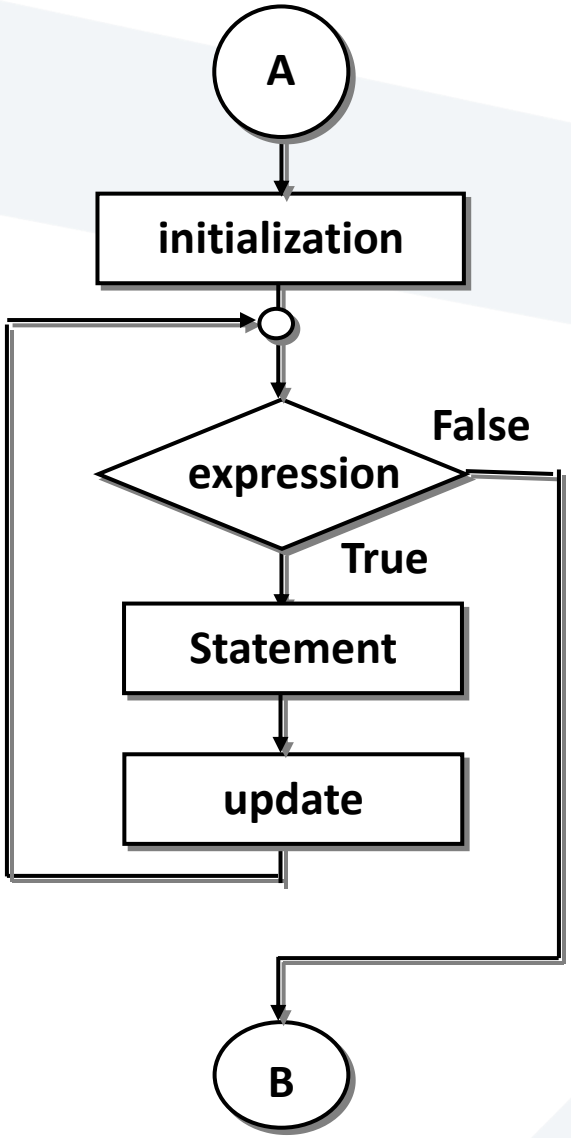
مثال 17: تطبيق المثال 16 مستخدم do while.

```
void main()
{
int x;
cout<<"enter number:”;
cin>>x;
do
{ cout<<x <<" "; x--;}
while(x>=10);
cout<<"++"<<endl;
 }
```



Start
Input x
do
print x
decrement x by one
While x value great then or equal to ten
Print ++
End

3-3-5- البنية التكرارية for
The for repetition structure



من البنى التكرارية ذات الشرط السابق.
تملك الصيغة العامة التالية :

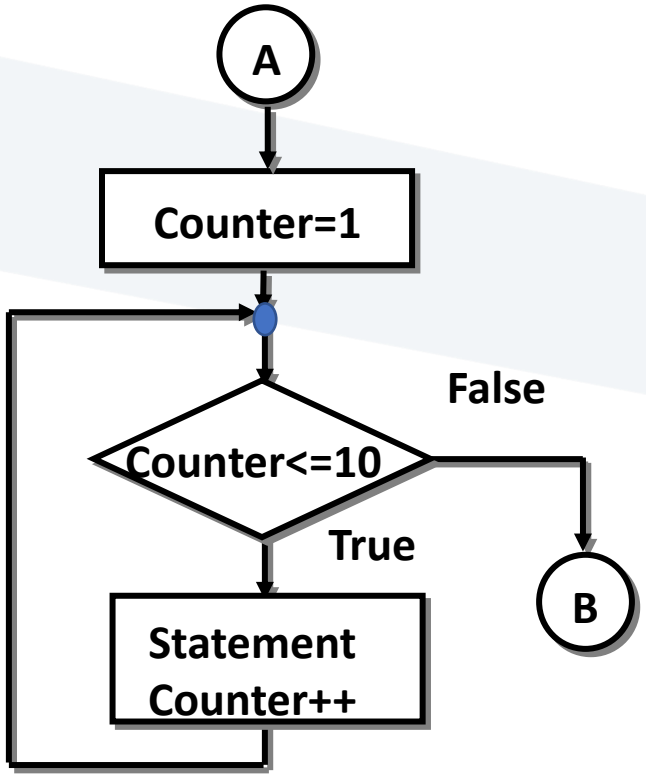
for(initialization ;loopContinuationCondition ;update) statement ;

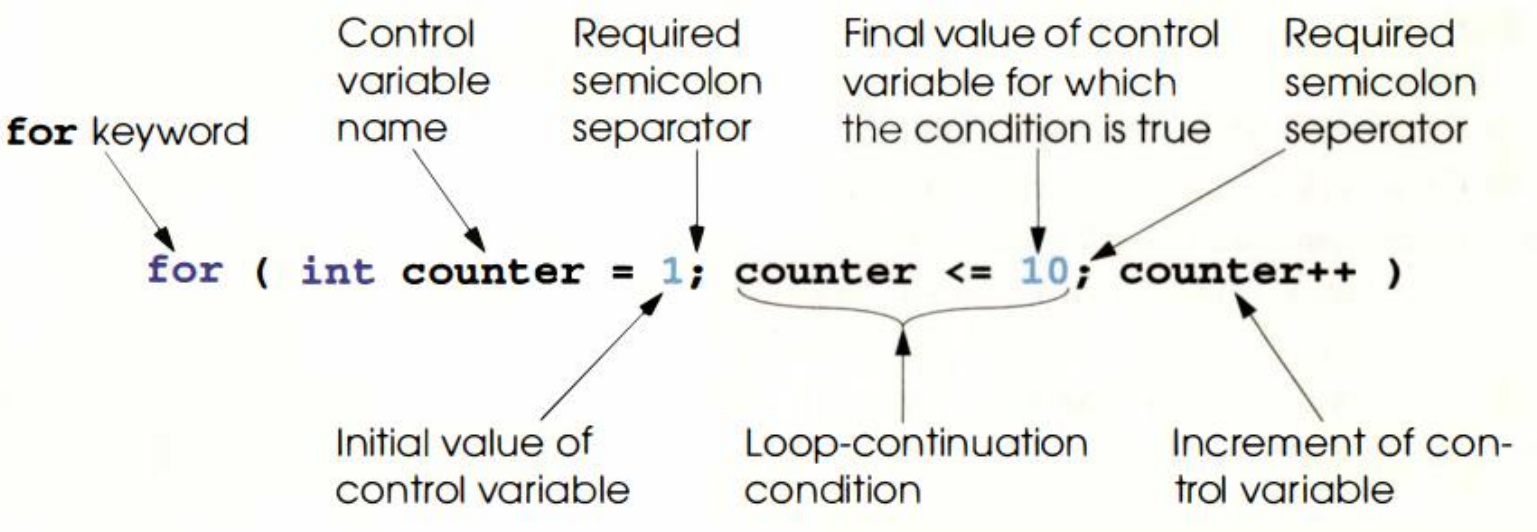
يتم التحكم في هذه البنية التكرارية بثلاثة أجزاء منفصلة:
القيمة الابتدائية (التهيئة) initialization.
شرط الستمرار الحلقة loopContinuationCondition.
تحديث القيمة update.

آلية تنفيذ الحلقة :

يأخذ متحول الحلقة قيمة هي القيمة الابتدائية initialization وهى معرفة فقط ضمن الحلقة إذا عرفت داخل القوس)، يتم اختبار شرط استمرار الحلقة, فإن كان محققاً تنفذ التعليمة statement، ثم يعطى متحول الحلقة قيمة جديدة ويعاد اختبار شرط استمرار الحلقة فإن كان محققاً تنفذ statement، ثم يعطى متحول الحلقة قيمة جديدة هكذا يستمر تنفيذ statement حتى يختل شرط الحلقة،عندها يتم الخروج من جسم البنية for.

مثال: حلقة الشرط السابق for





آلية التنفيذ:يتم التجهيز للـ ***initialization*** بقيمة عادةً تكون متعلقة بالشرط. وهنا counter =1; تنفذ لمره واحده فقط وهي معرفة فقط ضمن جسم الحلقة . تقدر قيمة شرط استمرار الحلقة ***loop Continuation Condition*** وهو هنا counter<=10 فإذا كانت true يتم تنفيذ جسم الحلقة (***statement*** والذي قد يكون تعليمة مفردة أو مركبة)، وتنفذ ***increment*** وهنا counter++ ثم يعاد تقدير قيمة ***Condition*** مرة أخرى، فإذا كان true يعاد تنفذ الجسم مرة أخرى وهكذا تتكرر هاتان الخطوتان حتى يصبح قيمة ***Condition*** مساوية false، وعندها يتم المتابعة ما بعد جسم الحلقة.

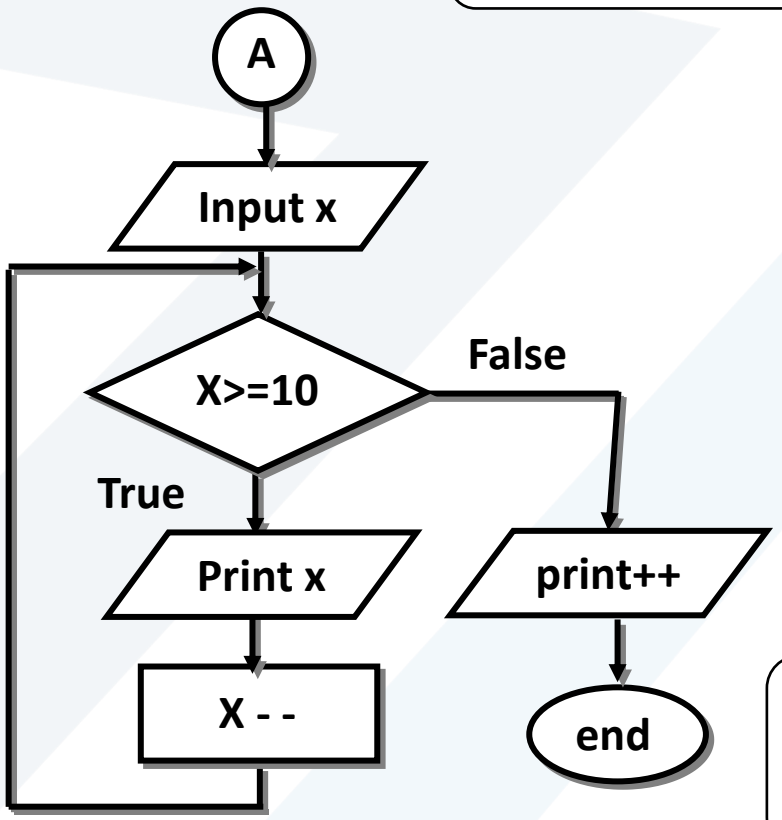
ملاحظة: قد يغيب ***initialization*** و***increment*** وتنهى باسلوب آخر.

مثال 18

مثال 18: تطوير المثال 16 مستخدم for.

```
void main()
{          int x;
cout<<"enter number:”;
cin>>x;
for( x ; x>=10; x - -)
   { cout<<x <<" ";}

cout<<"++"<<endl; }
```



Start
Input x
for x value great then or equal to ten
print x
decrement x by one
Print ++
End

ملاحظه: عرفت x خارج رأس الحلقة وهنا معرفة من مكان تواجدها حتى نهاية جسم التابع الرئيس main() {...}

مقارنة مابين الحلقات الثلاث

```
void main()
{int x;
cout<<"enter number:”;
cin>>x;
do
{ cout<<x <<" ";
 x--;} while (x>=10) ;
cout<<"++"<<endl;
 }
```

```
void main()
{int x;
cout<<"enter number:”;
cin>>x;

for( x ; x>=10; x - -)
{ cout<<x <<" ";}
cout<<"++"<<endl;
}
```

```
void main()
{int x;
cout<<"enter number:”;
cin>>x;

while (x>=10)
{ cout<<x <<" "; x--;}
cout<<"++"<<endl;
}
```

مقارنة مابين الحلقات الثلاث

مثال 19: طباعة الاعداد

مثال 19_1 طباعة الاعداد مابين العدد الصحيح المدخل الأصغرمن 10 و10 ويطبع ++ في النهايه.
نفس خوارزمية وكود المثال 16مع استبدال الشرط x>=10 بـ x<=10 وزيادة العداد بدل نقصانه.

```
void main()
{          int x;
cout<<"enter number:”;
cin>>x;
If(x>=10)
while(x>=10)
      { cout<<x <<" ";  x--;}
else
 while(x<=10)
      { cout<<x <<" ";  x++;}
cout<<"++"<<endl; }
```

مثال 19_2: طباعة الاعداد مابين العدد الصحيح المدخل و10 مهما تكن قيمته ويطبع ++ في النهايه .

الحل:

1- بعد إدخال العدد x

2- إذا كان x>10

2-1 – نعم نفذ كود المثال 16وانتقل إلى 3

2-2- لا نفذ كود المثال 19_1

3- اطبع ++

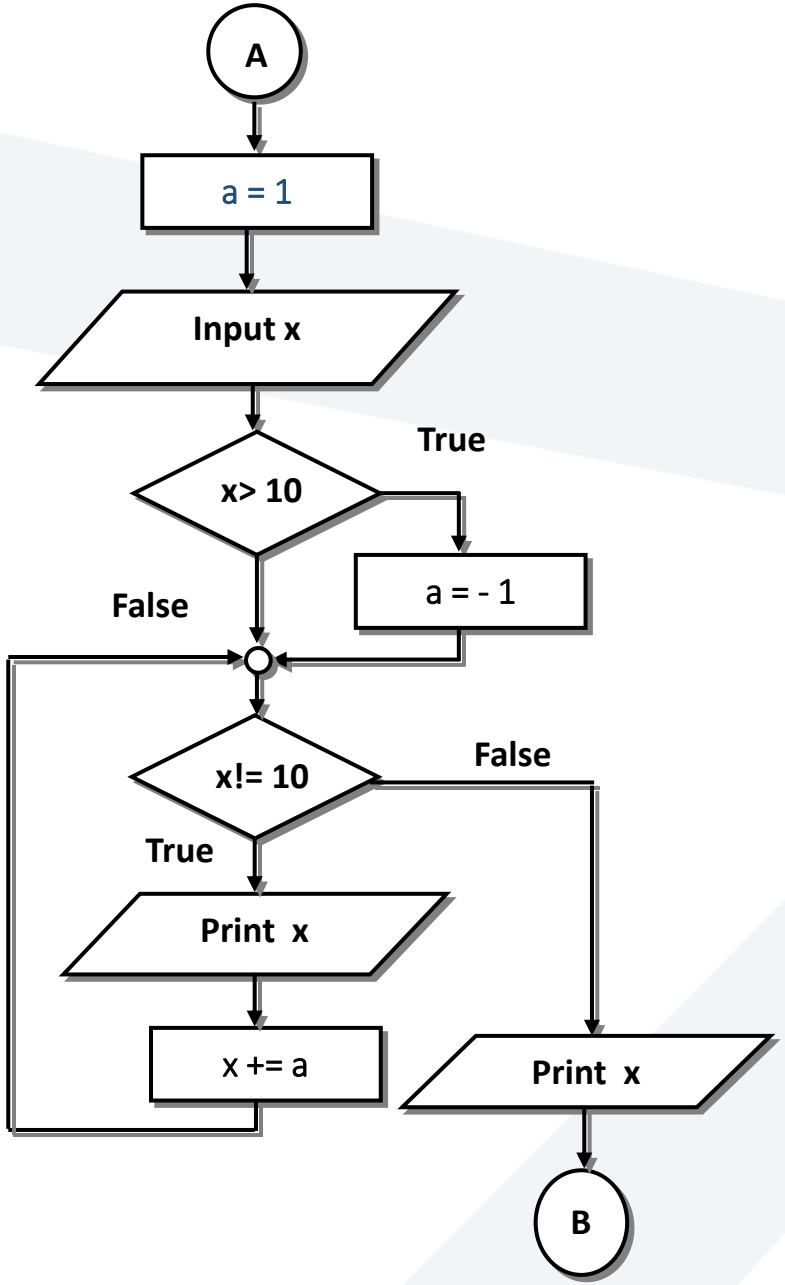
4- توقف

مثال 20: طباعة الاعداد مابين العدد المدخل و10 مستخدم حلقة واحدة فقط

مقارنة مابين الحلقتين السابقتين لكتابة الخوارزمية
1- شرط الحلقة =< أو <=
2- تغير الدليل - -x أو x++
الحل: جمع الشرطين بشرط وحيد هو =!
دمج البند الثاني يكون بتعريف متغير a=1
مفترضين x <10 وإذا كانت x >10 يكون a=-1
والدخول للحلقة طالما الشرط محقق، وفي النهاية لابد من طباعة x والتي تساوي 10 المخطط الانسيابي والكود البرمجي في الشرية التالية.

```
Start
Set a to one
Input  x
If (x>10) Set a to -1
While x value not equal to ten then
    Output   x
    Set  x= x+a
Print  x
End
```

مابين العدد مثال 20: طباعة الاعداد المدخل و10 مستخدم حلقة واحدة فقط




```
void main()
{
int x, a = 1 ;
cout<<"enter number : ";  cin >> x;
cout<<"the number between "<<x
     <<" and 10 are :"<<endl;
if (x> 10) a = -1 ;

while (x!= 10)
       {  cout << x<<"  ";x += a;}
cout << x<<endl;
}
```

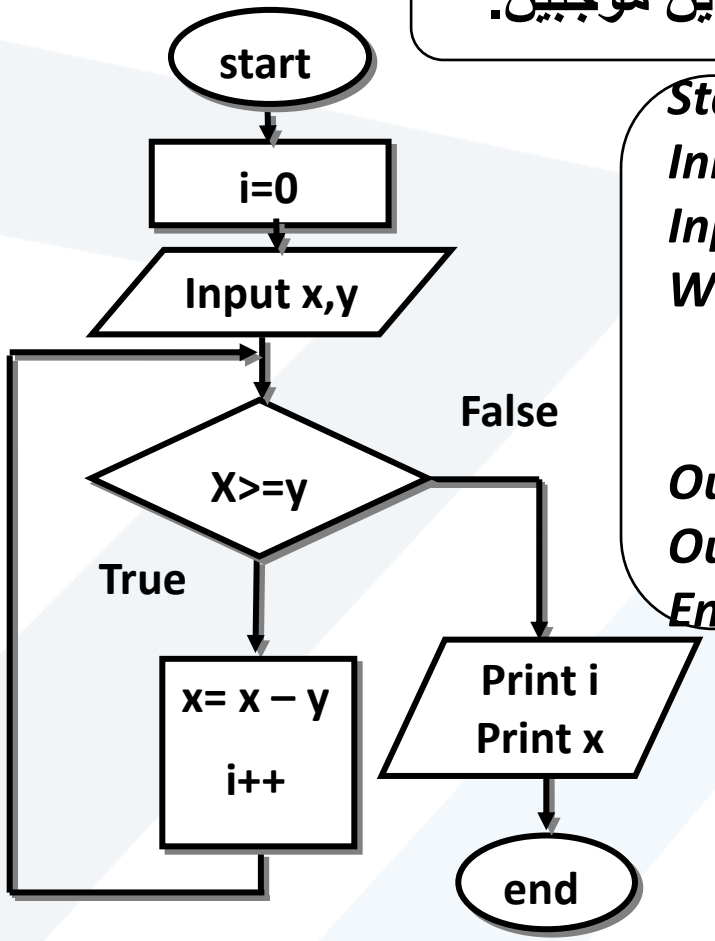
ناتج قسمة عددين

مثال 21: برنامج لإيجاد ناتج وباقي قسمة عددين int بدون استخدام معامل القسمة حيث x , y عددين موجبين.

```
void main()
{
   int x,y,i=0 ;
   cout<<"enter 2 int: ";
   cin>>x >>y;
   while(x >=y)
            {x=x-y ; i++ ;}
   cout<<"x /y= "<<i<<endl;
   cout<<"remainder="  <<x;
}
```

If input x=7, y=2 ➔ i=3,x=1

```
enter 2 int: 7 2
x / y = 3
remainder=1
```



Start
Initialize i to zero
Input x and y
While x value great then or equal to y
set x equal to x-y
add one to i
Output "x /y= " i
Output " remainder " x
End

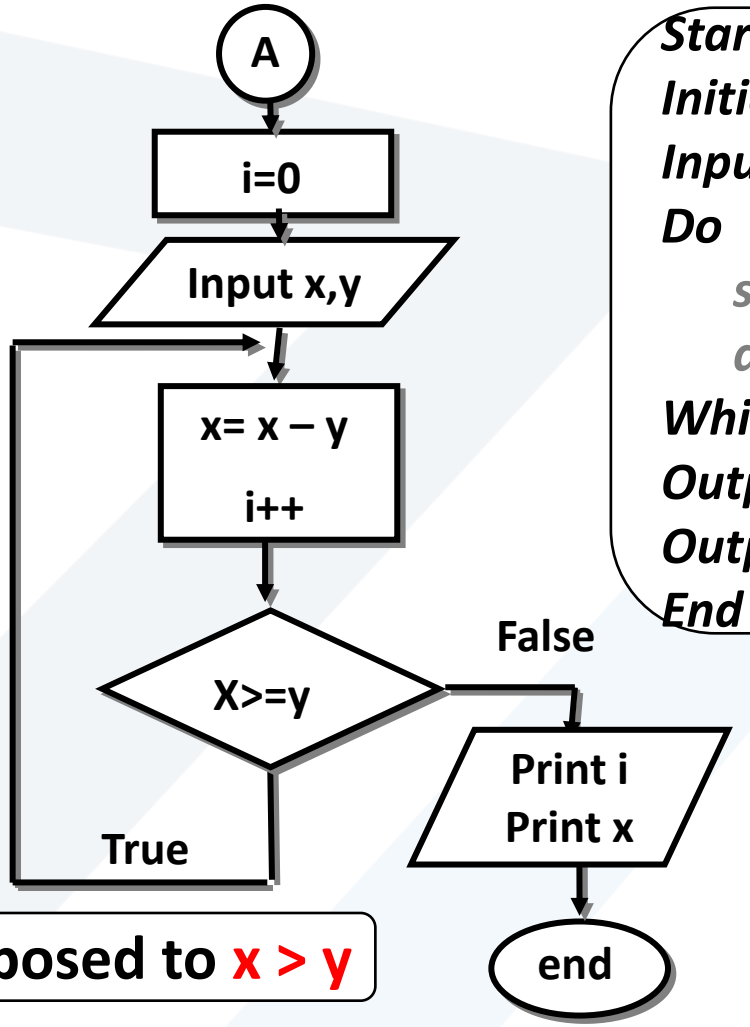
x >=y ✓	X=7-2=5	i=1
x >=y ✓	X=5-2=3	i=2
x >=y ✓	X=3-2=1	i=3
x >=y X		

Output tracking

مثال 22: تطوير المثال 15 برنامج ناتج وباقي القسمه مستخدم do while.

```
void main()
{
int x,y,i=0 ;
cout<<"enter 2 int: ";
cin>>x >>y;
do
   {x=x-y ; i++ ;}
while(x >=y);
cout<<"x / y="<<i<<endl;
cout<<"remainder"  <<x;
}
```

Start
Initialize i to zero
Input x and y
Do
set x equal to x-y
add one to i
While x value great then or equal to y
Output "x /y= " i
Output " remainder " x
End



If input x=7, y=2 ➔ i=3,x=1

enter 2 int: 7
2
x / y = 3
remainder=1

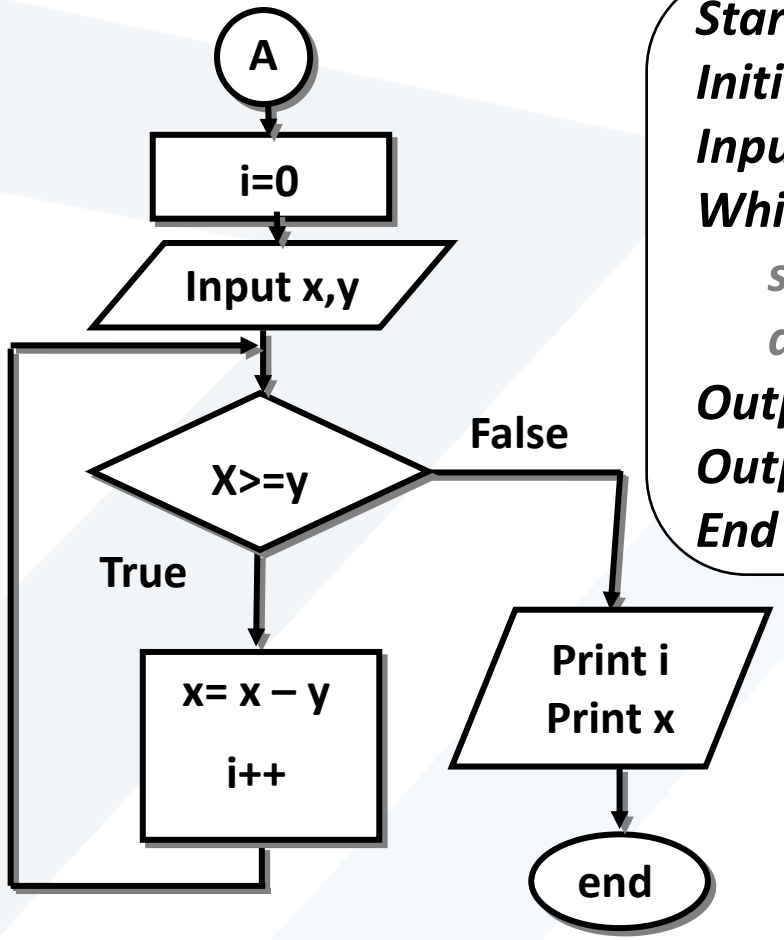
supposed to x > y

X=7-2=5	i=1	x >=y ✓
X=5-2=3	i=2	x >=y ✓
X=3-2=1	i=3	x >=y ✓
		x >=y X

Output tracking

مثال 23: تطوير المثال 15 برنامج ناتج وباقي القسمه مستخدم for.

```
void main()
{
int x,y,i=0 ;
cout<<"enter 2 int: ";
cin>>x >>y;
for( ; x >=y;  ) {x=x-y ; i++ ;}
cout<<"x / y="<<i<<endl;
cout<<"remainder"  <<x;
}
```



Start
Initialize i to zero
Input x and y
While x value great then or equal to y
set x equal to x-y
add one to i
Output "x /y= " i
Output " remainder " x
End

تم حذف القيمة البدائية وقيمة زيادة دليل الحلقة نظراً لإن القيمة البدائية موجوده قبل رأس الحلقة والتحديث موجود ضمن جسم الحلقة i++

x >=y ✓	X=7-2=5	i=1
x >=y ✓	X=5-2=3	i=2
x >=y ✓	X=3-2=1	i=3
x >=y X		

مثال 24 بفرض أننا نرغب بحساب معدل عشر قيم يتم إدخالها من لوحة المفاتيح

```
void main()
{
int  counter, total , average;
total = 0;counter = 1;

while (counter <= 10)
 {
   cout << "Enter grade: "; cin >> grade;
   total = total + grade;
   counter = counter + 1;
 }
int average = total / 10;
cout << "Class average is ";
cout<< average << endl;
}
```

1- start
2- initialize total to 0
3- initialize counter to 1
4- while counter <= 10 do
    i- print "Enter grade: "
    ii= input grade
    iii- add grade to total
    iv- increment counter by 1
5- set average equal to total / 10
6- print "Class average is "
7- print average
8- move to next line
9- end

الناتج قيمة صحيحة ولجعلها حقيقية نبدل int average = total / 10; بـ double average =(double) total / 10; وتعرف بعملية القسر

بعض الملاحظات

بعض الملاحظات على المثال 24

1. من الضروري إعطاء المتغيرين total , counter قيم ابتدائية وإلا سيتم إعطاؤها قيم عشوائية أو غير متوقعة وبالتالي نحصل على نتائج خاطئة.
2. المتغير counter يزيد بمقدار واحد من أجل كل عملية إدخال وجمع وعندما تصبح قيمته 11 يختل شرط حلقة while ويتم الخروج منها والمتابعه بعد جسمها.
3. المتغير average الذي يحسب المعدل أعلن عنه كمتغير صحيح، لذلك عند طباعته ستعطي القسم الصحيح من المعدل المحسوب أي قسر الناتج بشكل Implicit ولطباعة القيمه الحقيقية يجب قسر عملية القسمة وكتابة double average =(double) total / 10.
4. إن كافة القيم العددية الصحيحة المدخلة تقبل ضمن هذا التمرين مما يجعله غير دقيق أي من الممكن إدخال قيم لا تتناسب والعلامات (قيم سالبة أو قيم كبيرة جداً) ويجب معالجتها .

مثال 25 تطوير مثال 42 حساب معدل علامات كيفي مدخله من لوحة المفاتيح

1. start
2. initialize total to 0
3. initialize counter to 0
4. print "Enter grade: "
5. input grade
6. while ((grade greater than or equal 0) and (grade less than or equal 100)) do
   - add grade to total
   - increment counter by 1
   - print "Enter grade: "
   - input grade
7. if counter not equal to 0 then
   - set average equal to total / 10
   - print "Class average is "
   - print average
8. else
   - print "No grades were entered"
9. move to next line
10. end

```
void main()
{float average; int counter,grade,total;
 total=0;counter=0;
cout << "Enter grade [0..100]  out of range to end:";
 cin >> grade;
 while ( (grade >= 0) && (grade <= 100 ) )
{total = total + grade;  counter = counter + 1;
cout<<" Enter grade (0..100)  out of range to end: ";
cin >> grade; }
if (counter != 0)
{ average = (float) total / counter;
cout<<"Class average is"<<average; }
else cout<<"No grades were entered \n";
}
```

مثال 26 تطوير مثال 25 لحساب متوسط علامات الطلاب الناجحين في مقرر ما وعدد الطلاب الراسبين في ذلك المقرر.

1. start
2. initialize total to 0
3. initialize counter to 0
4. initialize count1 to 0
5. print "Enter grade: " input grade
6. while grade not equal to –1 do
   - if grade greater 10 or equal to 60 then
     1. add grade to total
     2. increment counter by 1
   - else
     1. increment count1 by 1
   - print "Enter grade: " input grade
7. if counter not equal to 0 then
   - set average equal to total / counter
   - print "Class average is “ average
8. else
   - print "No grades were entered GE 60"
9. if count1 not equal to 0 then
   - print "the numer of the failed stu is "
   - print count1
10. else print "No grades were entered LT 60"
11. move to next line
12. end

```
void main()
{float average; int counter,grade,total,count1;
 total=0;counter=0;count1=0;
cout << "Enter grade,-1 to end"; cin >> grade;
 while (grade != -1)
{if (grade >= 60)
{ total = total + grade; counter = counter + 1; }
else {count1 = count1+1;}
cout << "Enter grade, -1 to end ";cin >> grade; }
 if (counter != 0) {average = (float) total / counter;
cout << "Class average is " <<average << endl;}
else cout << "No grades were entered GE 60\n";
if (count1 != 0) {cout << "the num of the failed stu " ;
cout<<count1<< endl;}
else cout << "No grades were entered LT 60\n ";
}
```

بعض الأسئلة المطلوب

بعض الأسئلة المطلوب الإجابة عن التالي:

1- أكتب برنامجاً لحساب المجموع التالي: $SUM = x + x^2 + x^3 + \ldots + x^n$

2- طور البرنامج في التمرين 1 بحيث يقوم بحاسب المجموع التالي:

$$SUM = \frac{x}{1} + \frac{x^2}{2} + \frac{x^3}{3} + \ldots + \frac{x^n}{n}$$

3- طور البرنامج في التمرين 2 بحيث يقوم بحاسب المجموع التالي:

$$SUM = \frac{x}{1} - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} \ldots + \frac{x^n}{n}$$

بعض الأسئلة المطلوب

4- نفذ الكود التالي وفسر الناتج: ASCII table

```
int m,m1=118;
  char ch,ch1 = 'a';
  m=ch1; cout<<m<<endl;
  for (m1=256;m1>=0;m1--){ch=m1; cout<<m1<<"= "<<ch<<"  ";}
cout<<endl;
```

5-العدد السعيد هو عدد صحيح موجب يحقق الشرط التالي: إذا تم جمع مربع أرقامه، وأعيد القيام بتلك العملية للعدد الناتج مرة أو أكثر، يبقى في النهاية 1 (ويستقر عندها). أما الأعداد التي تظل تتكرر نتائجها في حلقة ولا تصبح أبداً 1 فتسمى أعدادا تعيسة مثل العدد 9. والمطلوب كتابة برنامج لمعرفة فيما إذا كان العدد المدخل سعيداً أم تعسياً.

6- عدد هَرشَد (Harshad number أو **عدد نيفين** Niven number) أو **متعدد الأرقام** multidigital number هو عدد قابل للقسمة على مجموع أرقامه والمطلوب كتابة برنامج لمعرفة فيما إذا كان العدد المدخل ينتمي لهذه الاعداد.

بعض الأسئلة المطلوب
لمن لديه رغبة بالمزيد

7- في نظرية الاعداد، **عدد مونشهاوزن** (Münchhausen number) هو كل عدد يساوي مجموع كل رقم له مرفوعا إلى قوة نفسه x^x على حدة (في العادة باستعمال الأساس 10). وهو يختلف عن العدد النرجسي لأن ارقامه لا ترفع إلى قوة ثابتة. مثلا: $3435 = 3125 + 27 + 256 + 27 = 5^5+3^3+4^4+3^3$.

الأعداد المونشهاوزن الأوائل هي 0, 1, 3435, 438579088..

سميت بالأعداد مونشهاوزن لأن الأرقام ترفع نفسها كما يزعم أن البارون فون هيرونيموس مونشهاوزن رفع نفسه بركوب كرة مدفع حديدية كما وصف في فيلم 1943 fantasy comedy film Münchhausen والمطلوب ادخال عدد واثبات أنه يتمي لها أو لاينيمي.

8- لعدد النرجسي أو **عدد ارمسترونغ** أو (perfect digital invariant أو **مثالي زائد** plus perfect number) عدد أرقامه n هو عدد يساوي مجموع أرقامه مرفوعة إلى n على حدة مثلا :

$3^3+5^3+1^3 = 153$ ، $0^3+7^3+3^3 = 370$، $1^3+7^3+3^3 = 371$، $7^3+0^3+4^3 = 407$

والمطلوب ادخال عدد واثبات أنه يتمي لها أو لاينيمي.