**قسم الرياضيات والحاسوب / مرحلة اولى**

# برمجة مهيكلة Structured Programming

## المحاضرة (4)

**الهدف من المحاضرة** : التعرف على المعاملات المنطقية والحلقات التكرارية وكيفية تنفيذها بلغة C++.

**مثال**: عبر عن الشروط التالية بصيغة تناسب لغة **C++.**

1. إذا كانت x أكبر من y فإن d=5.
2. إذا كانت الدرجة d أكبر أو يساوي 60 اطبع ناجح "pass" وإلا اطبع راسب "fail".
3. إذا كانت قيمة الجذر التربيعي sr أقل من 0 اطبع عدد تخيلي"imaginary number".
4. إذا كانت a لا تساوي 5 فإن b=2a+3، وc=3a-4b، وإلا فإنb=2a-11 ، وc=3a+b.

الحل:

```
1. if(x>y)
   d=5;
2. if(d>=60)
  cout<<"pass";
  else
  zcout<<"fail";
3. if(sr<0)
 cout<<" imaginary number";
4. if(a!=5)
   {
    b=2a+3;
    c=3a-4b;
   }
 else
 {
   b=2a-11;
   c=3a+b;
   }
```

## المعاملات المنطقية Logical Operators

ويتم عن طريقها الربط بين الشروط وهي ثلاثة معاملات مبينه في الجدول التالي:

| الأداة | المعنى | اتجاه التنفيذ |
|---|---|---|
| **&&** | تعني **and** "و" | كلا الاتجاهين |
| \|\| | تعني **or** "أو" | كلا الاتجاهين |
| ! | تعني **not** "لا" | اليسار لليمين |

**ملاحظة:**

أولوية التنفيذ لهذه المعاملات للتي تكون داخل أقواس وإلا فإن الأولوية تكون من اليسار لليمين.

## جداول الحقيقة Truth Tables

كما هو معلوم أن نتيجة الشرط إما صواب أو خطأ true or false، وفي حالة الجملة الشرطية المركبة فإن الناتج يعتمد على الشرطين المكونين للجملة ونوع الرابط المنطقي، وهذا موضح فيما يعرف في علم الرياضيات المتقطعة discrete mathematics بجداول الخطأ والصواب أو جداول الحقيقية truth tables والموضحه أدناه. "سنرمز للشرطين بالحرفين A, B وللصواب بالحرفT والخطأ بالحرفF"

**أولاً جدول الحقيقة للرابط المنطقي &&**

| A | B | Result |
|---|---|---|
| T | T | T |
| T | F | F |
| F | T | F |
| F | F | F |

**ثانياً جدول الحقيقه للرابط المنطقي ||**

| A | B | Result |
|---|---|---|
| T | T | T |
| T | F | T |
| F | T | T |
| F | F | F |

**ثالثاً جدول الحقيقه للرابط المنطقي!**

| A | !B |
|---|---|
| T | F |
| F | T |

**مثال**: عبر عن الشروط التالية بصيغة تناسب لغة **C++.**

1. إذا كانت x أكبر من y و y تساوي 0 فإن d=5.
2. إذا كانت الدرجة d أكبر أو يساوي 80 أوd أقل من 90 اطبع "B".
3. إذا لم تكن( x أكبر من y و y تساوي 0) أو x تساوي 3 فإن d تساوي10 وإلا فإن d تساوي 20.

**الحل:**

```
1.   if(x>y&&y==0)
     d=5;
2.   if(d>=80||d<90)
    cout<<"B";
3.   if(!(x>y&&y==10)||x==3)
    d=10;
    else
    d=20;
```

**الحلقات التكرارية** Loops

تمثل هذه الحلقات الجزء الثاني من بنى التحكم فعند الرغبة في تكرار أمر" أو أوامر" معينه تستخدم هذه الحلقات، هنالك ثلاثة أنواع من هذه الحلقات. يجب عند تصميم الحلقة تحديد الآتي:

- ♦ الأوامر التي تحتاج إلى تكرار لتوضع داخل جسم الحلقة.
- ♦ عدد مرات تنفيذ الحلقة، وذلك عن طريق متغير من نوع **int** يسمى عداد الحلقة وصياغة شرط استمرار الحلقة أو توقفها.

**1- حلقة for**

**الصيغه العامة**

```
for(counter_initial_value;condition;counter_incrementation/decrementatin)
{
.       statements " loop body"
```

.

**}**

الصيغة العامة تحتوي على :

**for** وهي كلمة محجوزة بين القوسين المستديرين توجد ثلاثة جمل برمجية:

- counter_initial_value وهي الجملة المحتوية على إعطاء عداد الحلقة counter القيمه الإبتدائية، ويمكن أن تحتوي هذه الخطوة على تعريف العداد إذا لم يكن أعلن عن تعريفه من قبل، ويمكن أن تتم عمليتي التعريف وإعطاء القيمة الإبتدائية قبل الحلقة وبالتالي يمكن ان نستغني عن هذا التعبير.
- condition هو شرط استمرار الحلقة أي أن الحلقه تستمر إذا كان الشرط صحيح وتتوقف إذا انتفى الشرط، وهذا الشرط يكتب باستخدام المعاملات العلائقيه آنفة الذكر، كما يمكن أن يكون الشرط مركباً.
- counter_incrimintation/decrimintation وهو معدل الزيادة incrimination أو النقصان decrimintation في قيمة عداد الحلقه، ويمكن أن تتم هذه الخطوة ضمن الأوامر داخل جسم الحلقة وبالتالي يمكن ان نستغني عن هذا التعبير، ويستخدم لذلك معاملات خاصة كما هو موضح:

x++⟺x=x+1

x-- ⟺x=x-1

x=variable "mathematical_operator" " value"

x"mathematical_operator"= " value"

**أمثله توضيحية**

تم تعريف العداد وإعطاء القيمه الإبتدائية له داخل تركيبة for

- for(int i=3;i<=8;i++) {…….}

تم تعريف العداد في الدالة الرئيسية وإعطاء القيمه الإبتدائية له داخل تركيبة for

- int i=3;
  for( ;i<=8;i--) {……}

معدل التغير في العداد موضوع داخل جسم الحلقه

- for(int i=3; i<=8;) { i*=3;…}

تم تعريف العداد وإعطاء القيمه الإبتدائية له في الدالة الرئيسية ووضع معدل التغير فيه في

- int i=3;

جسم الحلقه

for( ;i<=8;) { i/=3;…}

حلقه مرتبطه بمتغيرين

- for(int i=3,int j=9;i<=8||j>0;i=i+2,j--) {……}

**مثال**: أكتب برنامجاً بلغة C++ يطبع الأرقام الفردية odd number المحصورة بين 2 و 20

**الحل**: هذه الحلقه تحتاج لعداد يتغير بمعدل اثنين "الفرق بين آل عدد فردي والذي يليه" والعداد قيمته الإبتدائية 3 وينتهي بـ 20 "شرط الحلقه" كما هو موضح في البرنامج:

```
#include<iostream.h>
void main()
{
for(int counter=3;counter<=20;counter+=2)
cout<<"\n"<<counter;
}
```

**2- حلقة while**

**الصيغه العامه**

**while**(condition)

{

. statements " loop body"

.

}

نفس الوضع القائم في الحلقه السابقة غير أن معدل التغير في قيمة العداد يكون داخل جسم الحلقه، وتعريف العداد وإعلان قيمته الإبتدائية قبل الحلقه، يمكن إعادة كتابة البرنامج السابق باستخدام حلقة **while**:

```
#include<iostream.h>
void main()
{
    int counter=3;
while(counter<=20)
{
cout<<"\n"<<counter;
counter+=2;
}
```

**3- حلقة do-while**

**الصيغه العامه**

**do**

{

. statements " loop body"

.

}

**while**(condition);

نفس الوضع القائم في الحلقه السابقة غير أن معدل التغير في قيمة العداد يكون داخل جسم الحلقه، وتعريف العداد وإعلان قيمته الإبتدائية قبل الحلقه، نجد أن هذه الحلقه عكس حلقة **while** حيث اختبار الشرط مؤخر بعد جسم الحلقه مما يعني أن حلقة **do-while** لابد أن تنفذ مره على الأقل حتى ولو انتفى شرط الحلقه، يمكن إعادة كتابة

```
#include<iostream.h>
void main()
{
     int counter=3;
     do
       {
cout<<"\n"<<counter;
counter+=2;
}
while(counter<=20);
}
```