مدخل الى الخوارزميات والبرمجة

عنوان المحاضرة: بنى التحكم – بنى الاختبار
المحاضر: د. حيدر خليل
الكلية: الهندسة
رقم المحاضرة 5+6.

أهداف المحاضرة:
معرفة الطالب ببنى التحكم و بنى الاختبار الأساسية في لغة C++ لحل المسائل البرمجية التي تحتاج اختبارات منطقية.

- بنية الاختبار IF.
- بنية الاختبار IF/else.
- بنية الاختبار IF/else المتداخلة.
- البنية متعددة الاختبار **switch**.

1- بنى التحكم :

يتم عادة تنفيذ تعليمات برنامج ما واحدة تلو الأخرى حسب ترتيب ورودها في نص البرنامج من الأعلى باتجاه الأسفل، نسمي أسلوب التنفيذ هذا بالتنفيذ التسلسلي Sequential execution.

سوف نقوم في الفقرات التالية بعرض العديد من تعليمات C++ التي تساعد المبرمج على تحديد التعليمة التي ستنفذ بعد التعليمة الحالية والتي يمكن أن تأتي في موضع آخر من نص البرنامج مختلف عن موضع التعليمة التالية لها مباشرة، نسمي هذه العملية بعملية نقل التحكم transfer of control ، عملية نقل التحكم هذه ضرورية في حل المشاكل التي تتطلب إجراء اختبارات منطقية لتنفيذ تعليمة معينة وتجاهل أخرى أو العكس، كذلك في تلك التي تتطلب تكرار تعليمة عدداً من المرات طالما أن الشرط محقق أو حتى يتحقق شرط ما (الحلقات).

وقبل الخوض في أنواع بنى التحكم لا بد من التذكير بالمؤثرات العلائقية والمنطقية.

2- أنواع بنى التحكم :

من الممكن كتابة البرنامج في لغة C++ باستخدام ثلاث بنى للتحكم هي :

1- البنية التسلسلية :

حيث يقوم الحاسب بتنفيذ تعليمات C++ واحدة بعد الأخرى حسب ترتيب ورودها ضمن نص البرنامج.

2- **بنية الإختيار** :

توفر لغة C++ ثلاثة أنواع من بنى الإختيار:

أ- **البنية if** : وتسمى بالبنية وحيدة الإختيار.

ب- **البنية if/else** : وتسمى بالبنية مضاعفة الإختيار كما يمكن أن نجد البنية if/else المتداخلة.

ج- **البنية switch** وتسمى بالبنية متعددة الإختيار.

2- **بنية التكرار** :

توفر لغة C++ أيضا ثلاثة أنواع من البنى التكرارية :

أ - **البنية while** : وتسمى بالبنية ذات الشرط المسبق(حلقة while).

ب- **البنية do\ while** :و تسمى بالبنية ذات الشرط الملحق(حلقة do\while).

ج- **البنية for :** وتسمى بالبنية ذات العدد المعروف من مرات التكرار(حلقة for).

3- بنية الإختيار if :

The if selection structure

تسمى بالبنية وحيدة الإختيار Single selection structure لأنـها تسمح باختيـار أو تجاهل فعل وحيد.

تملك الصيغة العامة التالية :

if (expression) statement;

حيث أن:

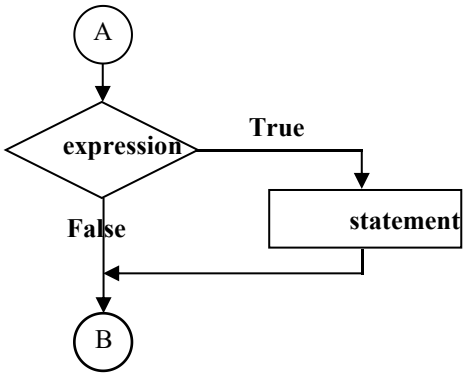
expressio: عبارة عن تعبير منطقي أو شرط نتيجة اختباره true أو false.

statement :عبارة عن تعليمة قد تكون بسيطة أو مركبة.

آلية تنفيذ التعليمة :

يتم اختبار expression فإذا كانت نتيجة الاختبار true أو لا تساوي الصفر سيتم تنفيذ statement، أما إذا كانت نتيجة الإختبار false فسيتم تجاهل statement.

يبين الشكل(3-1) المخطط النهجي للبنية if.



الشكل (3-1) المخطط الصندوقي لبنية الإختيار if

مثال 3 -1 :

```
#include<iostream>
using namespace std;
int main()
{
int x;
cout<<"enter number:";
cin>>x;
if(x>5)
cout<<"**"<<endl;
cout<<"++"<<endl;
cout<<"--"<<endl;
return 0;}
```

في هذا المقطع البرمجي أعلن عن x بأنه متغير من النوع الصحيح، فإذا تم تنفيذ البرنامج وإدخال قيمة لـ x سنميز حالتين :

أ- إذا كانت قيمة x أكبر من 5 مثلا 6 :

هذا معناه أن شرط if محقق وبالتالي ستنفذ التعليمة التي بعد if والتي هي في هذا المثال تعليمة بسيطة وسيتم طباعة **، ثم ينتقل الحاسب لتنفيذ التعليمات التالية بشكل تسلسلي حيث يطبع ++ ثم --.

خرج البرنامج :

```
enter number : 6
* *
+ +
- -
```

ب- إذا كانت قيمة x أصغر من 5 مثلا 1- :
هذا معناه أن شرط if غير محقق، وبالتالي سيتم تجاهل التعليمة التي بعد if ومتابعة تنفيذ التعليمات التالية بشكل تسلسلي.
خرج البرنامج :

```
enter number : -1
++
--
```

مثال 3 -2 :

```
#include<iostream>
using namespace std;
int main()
{
int x;
cout<<"enter number : ";
cin>>x;
if(x>5){
cout<<"**"<<endl;
cout<<"++"<<endl;
}
cout<<"--"<<endl;
return 0}
```

إذا تم إدخال x=7 سيكون خرج البرنامج :

```
enter number : 7
**
++
--
```

ولكن إذا تم إدخال x=3 سيكون الخرج:

```
enter number : 3
--
```

حيث سيتم تجاهل التعليمة المركبة بعد if والتي تضم تعليمتي طباعة ** و++ لعدم تحقق شرط if.

مثال 3-3 :

هـذا البرنـامج يقـوم بطباعة أكبر عدد بين ثلاثة أعداد يتم إدخالها إلى البرنـامج:

```
#include<iostream>
using namespace std;
int main()
{
int n1,n2,n3;
cout<<"enter three integer:";
cin>>n1>>n2>>n3;
int max=n1;
if(n2>max)max=n2;
if(n3>max)max=n3;
cout<<"the maximum is:"<<max;
cout<<endl;
return 0};
```

خرج البرنـامج :

```
enter three integer : 8  7  9
the maximum is : 9
```

تم إدخال القيم:

n1=8,n2=7,n3=9

في البداية تم تخصيص قيمة n1 للمتغيرmax أي(max =8) ثم تم اختبار كون n2>max والنتيجة false وبالتالي تم تجاهل التعليمةmax=n2، بعد ذلك تم اختبار كون n3>max وبما أن النتيجة كانت true تم تخصيص قيمة n3 للمتغير max (max=9) ومن ثم طبعت القيمة العظمى.

4 - بنية الاختيار if \else :

The if else selection structure

تسمى هذه البنية بالبنية مضاعفة الإختيار Double – selection – structure أنها تسمح بالإختيار بين فعلين مختلفين.

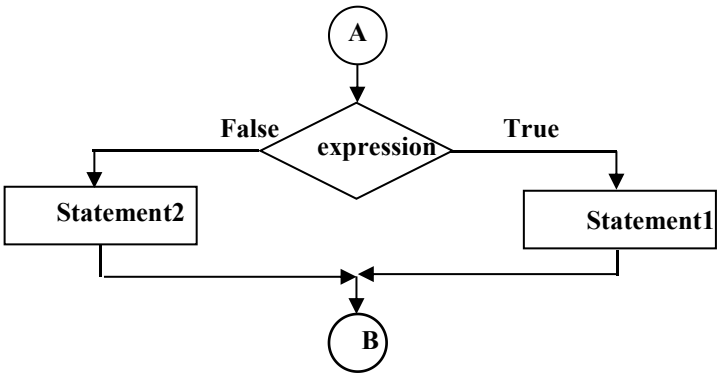
تملك الصيغة العامة التالية :

if (expression) statement1 ; else statement 2;

آلية تنفيذ التعليمة :

يتم اختبار expression فإذا كانت نتيجة الاختبار true أو لا تساوي الصفر يتم تنفيذ statement1 وتجاهل statement2، أما إذا كانت نتيجة الاختبار false أو تساوي الصفر يتم تجاهل statement1 وتنفيذ statement2.

ويبين الشكل(4-1) المخطط الصندوقي للبنية if/else.



الشكل (4-1) المخطط الصندوقي لبنية الإختيار if /else

هذا ويمكن أن نكتب التعليمة if\else بالشكل التالي :

Expression ? statement1:statement2;

آلية التنفيذ : إذا كان الشرط محققاً تنفذ statement1 وإلا تنفذ statement2.

مثال :

إذا كان grade =< 48 اطبع " ناجح " وإلا إطبع " راسب ".

grade >=48 ?cout <<"passed":cout<<"failed";

مثال 4-1

```
#include<iostream>
Using name space std;
int main()
{
int x;
cout<<"enter number : ";cin>>x;
if(x>5)
cout<<"**"<<endl;
else  cout<<"++"<<endl;
cout<<"--"<<endl;
cout<<"# #"<<endl;
return 0 };
```

إذا تم تنفيذ البرنامج وأدخلت قيمة لـ x سنميز حالتين:
أ- إذا كان x>5 مثلاً x=7 هذا يعنـي أن شرط if محقق وبالتالي سيكون خرج البرنامج:

```
enter number : 7
**
--
##
```

ب- إذا كان x<5 مثلاً x = -1 فهذا يعني أن شرط if غير محقق وبالتالي سيكون خرج البرنامج :

```
enter number : -1
++
--
##
```

مثال 4-2:

```
#include<iostream>
using namespace std;
int main()
{
int x;
cout<<"enter number : ";
cin>>x;
if(x>5){
cout<<"**"<<endl;
cout<<"++"<<endl;}
else
cout<<"--"<<endl;
cout<<"##"<<endl;
return 0};
```

إذا كان x>5 فسيكون خرج البرنامج :

```
enter number : 7
* *
++
##
```

وذلك لتنفيذ التعليمة المركبة بعد if وإهمال تلك التي بعد else.

إذا كان x<5 سيكون خرج البرنامج :

```
enter number : -1
- -
# #
```

وذلك بسبب اختلال شرط if وتنفيذ التعليمة بعد else وتجاهل التعليمة المركبة بعد if

.

5- بنية الاختيار if \ else المتداخلة

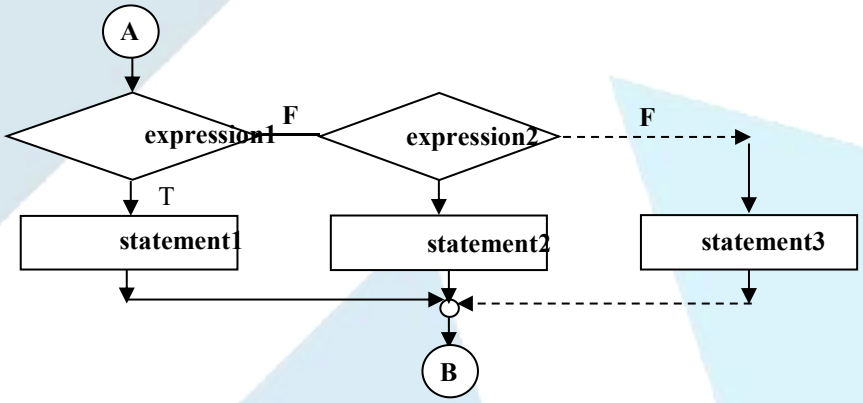
Nested if else selection structure

يتم اختبار عدة تعابير منطقية أو شروط وينتهي العمل عند إحداها.
تملك الصيغة العامة التالية :

if (expression 1) statement 1; else
if (expression 2) statement 2 ; else
if (expression 3) statement 3……..

آلية تنفيذ التعليمة :

يتم اختبار expression1 فإذا كانت نتيجة الاختبار true أو لا تساوي الصفر يتم تنفيذ statement1 وتجاهل باقي البنية if else \ المتداخلة ولكن إذا كانت نتيجة الاختبار false أو تساوي الصفر يتم تجاهل statement1 وتنفيذ ما بعد else حيث يتم اختبار expression2 وهكذا دواليك.
ويبين الشكل (3-3) المخطط الصندوقي للبنية if\else المتداخلة.



الشكل (3-3) المخطط الصندوقي لبنية الإختيار if /else المتداخلة

مثال 5-1:

في هذا المثال سيتم إدخال علامة الطالب من خلال المتحول الصحيح grade فإذا كانت العلامة المدخلة أكبر أو مساوية لـ90 سيتم طباعة المحرف A وتجاهل باقي تعليمات البرنامج.
في حين أنه يطبع B إذا كانت العلامة بين 80 و90 .
وسيطبع C إذا كانت العلامة المدخلة بين 70 و79.

وسيطبع M إذا كانت العلامة المدخلة بين 60 و69.
و سيطبع f إذا كانت العلامة المدخلة أصغر من 60.

```
#include<iostream>
using namespace std;
int main()
{
int grade;
cout<<"enter grade:";
cin>>grade;
if(grade >=90)
   cout<<"A"<<endl;
else
   if(grade >=80)
      cout<<"B"<<endl;
    else
      if(grade >= 70)
         cout<<"C"<<endl;
        else
          if(grade >= 60)
             cout<<"M"<<endl;
           else

             cout<<"F"<<endl;return 0}
```

خرج البرنامج

```
enter grade: 75
C
Press any key to continue
```

6- بنية الإختيار switch

The switch selection structure

تسمى بالبنية متعددة الإختيار multiple – selection – structure لأنها تسمح باختيار فعل محدد من بين مجموعة ممكنة من الأفعال المختلفة.

تملك الصيغة العامة التالية :

```
switch (expression) {
case constant 1: statement 1; break;
case constant 2: statement 2; break;
case constant 3: statement 3; break;
.
.
default : statement n ; }
```

آلية تنفيذ التعليمة :

تبدأ بنية الاختيار switch بالكلمة المفتاحية switch.

Expression: هو عبارة عن تعبير يأخذ قيماً صحيحة أو محرفية كذلك الأمر بالنسبة للثوابت constant.

الأمر switch يحدد قيمة التعبير expression فإذا كانت تلك القيمة مطابقة لقيمة أحد الثوابت في الحالات case يتم تنفيذ التعليمة statement (مركبـة أو بسيطة) والمرتبطة بتلك الحالة ثم يتم الخروج من البنية switch.

إذا لم يحدث أي تطابق بين قيمة expression وقيمة أحد الثوابت في حالات case يتم تنفيذ التعليمة المرتبطة بـ default ومـن ثـم الخروج من switch، ولكن وفي هذه الحالة وإذا لم تكن default موجودة سيتم الخروج من switch دون تنفيذ أي من تعليماتها.

إن عمليـة الخروج مـن switch تؤمن بواسطة التعليمة break التي تسبب نقل التحكم خارج البنية وسندرسها فيما بعد.

مثال 6-1 :

```
#include<iostream.h>
using namespace std;
int main()
{
char ch; int x=10;int y=5;
cout<<"enter mathematical operations:”;
cin>>ch;
switch (ch){
case'+':
cout<<x<<'+'<<y<<'='<<x+y<<endl;
break;
case '-':
cout<<x<<'-'<<y<<'='<<x-y<<endl;

break;
case '*':
cout<<x<<'*'<<y<<'='<<x*y<<endl;
break;
case '/':
cout<<x<<'/'<<y<<'='<<x/y<<endl;
break;
case '%':
cout<<x<<'%'<<y<<'='<<x%y<<endl;
break;}
return 0 }
```

إذا تم إدخال المحرف + سيتم إيجاد مجموع العددين x,y، أما إذا تم إدخال المحرف – سيتم إيجـاد فرق العددين x,y وهكذا، بالتالي يقوم البرنامج مقام آلة حاسبة بسيطة.

خرج البرنامج

```
enter  mathematical operations :*
10 * 5 = 50
Press any key to continue
```

وإذا تم إدخال محرف مخالف للمحارف المذكورة فلن نحصل على خرج.

```
enter  mathematical operations :=
Press any key to continue
```

ملاحظة 1 :
إن ثوابت الحالة case محارف لذلك وجب وضعها ضمن فاصلتين علويتين.
ملاحظة 2 :
يمكن إعطاء رسالة خطأ عند إدخال محرف غير مناسب بإدخال الجزء default في البرنامج بالشكل :

```
default  : cout << " error !! enter a new char " << endl ;
```

ملاحظة 3 :
إذا كان للثوابت المختلفة لحالات case نفس التعليمات يمكن أن يستبدل الشكل :

```
case  constant 1 : statement1 ; break;
case  constant 2 : statement1 ; break;
case  constant 3 : statement2 ; break;
case  constant 4 : statement2 ; break;
    .
    .
    .
```

بالشكل :

```
case  constant 1 : constant 2 : statement 1 ; break;
case  constant 3 : constant 4 : statement 2 ; break;
    .
    .
```

مثال 6-2 :
هذا البرنامج يطبع red إذا كان المحرف المدخل r أو R، في حين يطبع white إذا كان المحرف المدخل w أو W، ويطبع blue إذا كان المحرف المدخل b أو B ويطبع رسالة error char إذا كان المحرف المدخل غير ذلك.

```
#include<iostream>
using namespace std;
int main()
{
char choice;
cout<<”enter a char:”;
cin>>choice;
switch (choice){
case 'r':case 'R':
cout<<"red"<<endl;break;
case 'w':case 'W':
cout<<"white"<<endl;break;
case 'b':case 'B':
cout<<"blue"<<endl;break;
default:
cout<<"error char"<<endl;break;}
return 0;
}
```

خرج البرنامج

```
enter achar : r
red
```

ملاحظة 4 :

إذا ك انت الثوابت constant أعداداً صحيحة (ليست محارفاً) عندئذ لا توضع هذه الثوابت ضمن فواصل علوية.

مثال 6-3:
يمكن كتابة البرنامج في المثال 3-6 باستخدام البنية switch كمايلي:

```
#include<iostream>
using namespace std;
int main()
{
int score;
cin>>score;
switch (score/10){
case 10:case 9:
cout<<'A'<<endl;break;
case 8:cout<<'B'<<endl;break;
case 7:cout<<'c'<<endl;break;
case 6:cout<<'M'<<endl;break;
case 5:case 4:case 3:case 2:
case 1:cout<<'F'<<endl;break;
default:cout<<"error score."<<endl;
}
return 0;}
```

يتم إدخال علامة الطالب في المتحول score، ثم تقسم على10، طالما أن معامل switch صحيح سيتم أخذ القسم الصحيح من ناتج القسمة.