

الدرس السابع :

سوف نتناول في هذه المحاضرة الموضوعات التالي :

- ١- مقدمة لعبارات التحكم.
- ٢- الخوارزميات algorithms.
- ٣- شبه الكود pseudocode.
- ٤- هياكل التحكم.

بسم الله وبه نستعين

المقدمة

عندما نقوم بتنفيذ مهمة أو حل مشكلة برمجية ، يجب أولاً أن يكون لدينا فهم جيد لها ، وذلك لكي نستخدم قطع البناء ذات التقنيات المناسبة لها ، بطريقة وتخطيط نستخلص من خلالهما كود واضح ذات أداء وكفاءة عاليين. في هذا الفصل والذي يليه سوف نتعرف على عبارات التحكم في تحضير النظريات ومبادئ البرمجة الهيكلية (المركبة). في هذا الفصل (المحاضرة القادمة) نقدم عبارات التحكم (while , if , if... else) ، ثلاث من قطع البناء التي تسمح للمبرمج ان يحدد المنطق المطلوب لدوال الأعضاء لتؤدي مهامها. نحن نكرس جزء من هذا الفصل (والفصلين ٥ ، ٧) لتطوير أبعد للصف GradeBook ، حيث نقوم بإضافة دالة عضو للصف حيث يستخدم بها إحدى عبارات التحكم أو كلاهما لحساب معدل مجموع درجات الطالب وأيضاً هناك وسائل إضافية لضم عبارات لحل مشاكل مشابهة.

الخوارزميات algorithms

أي مسألة أو مهمة قابلة للتنفيذ، تكون منفذة بإتباع سلسلة من الإجراءات في ترتيب معين. ويركز هذا الترتيب على ناحيتين:

- ١- إجراءات التنفيذ نفسها. ٢- الترتيب الي نفذت به.
- وهذا ما يسمى بالخوارزم algorithm. ولكي نوضح هذا المصطلح ، نأخذ المثال التالي، " خوارزم النهوض إلى العمل " لموظف صغير، عندما ينهض من السرير ليذهب إلى العمل :

- ١- يغادر السرير.
 - ٢- يخلع البجامة.
 - ٣- يستحم.
 - ٤- يرتدي ثوبه.
 - ٥- يتناول فطوره.
 - ٦- يقود سيارته للعمل.
- هذه المهمة منفذة على شكل خطوات مرتبه بطريقة معينة، عندما يختل هذا الترتيب فإننا سوف نفقد

التحكم والسيطرة، وبالتالي تزداد المهمة صعوبة. على سبيل المثال، لو اختلف الترتيب وفق التالي :

- ١- يغادر السرير.

- ٢- يخلع بجامته.
 - ٣- يرتدي ثوبه.
 - ٤- يستحم.
 - ٥- يتناول فطوره.
 - ٦- يقود سيارته للعمل.
- بهذه الطريقة سوف يذهب للعمل مبولا. تحديد الترتيب في عبارات (إجراءات) تنفذ في برنامج كمبيوتر تسمى " تحكم البرنامج ".
في هذا الفصل سوف نأخذ عبارات التحكم لنرى كيف يسير تحكم برنامج.

شبه الكود pseudocode

شبه الكود لغة غير رسمية مبتكرة تساعد المبرمجين بأن يطوروا الخوارزميات بدون أن يقلقوا حول التفاصيل الدقيقة لنص لغة الـ C++، بعد
تحضير شبه الكود يتم تحويله إلى أجزاء مركبة (مهيكلية) من برامج الـ C++، وهو ليس برمجة فعلية، شكله يشبه اللغة الإنجليزية اليومية
، بحيث يكون سهل القراءة والفهم .
يساعد شبه الكود المبرمجين في التفكير في البرنامج قبل محاولة كتابته بأي لغة برمجة ، وسوف نعرض في هذا الفصل عدة امثلة نستخدم فيها شبه الكود في تطوير برامجنا. يمكن كتابة شبه الكود في أي محرر للنصوص يتم تجهيزها لحين الطلب.
عند القاء نظره على شبه الكود نلاحظ أن
الإعلانات (تلك التي لا تحتوي عمليات اسناد قيم اوليه للمتغيرات ولا تحتوي على بواني) لا يتم كتابتها ، على سبيل المثالك :
`int i;`
نلاحظ في هذه الإجراءات لا يوجد تنفيذ (على سبيل المثال : ادخال ، إخراج ، حساب ، اسناد قيمة) ، هذا يحدث في معظم الحالات.
سوف نعطي فيما يلي مثالا بسيط لتوضيح ماسبق شرحه، نأخذ برنامج و نمثل خوارزميه له هي
عبارة عن شبه كود.

البرنامج :

١

كود

// Fig. 2.5: fig02_05.cpp

```
2 // Addition program that displays the sum of two numbers.
3 #include <iostream> // allows program to perform input and output
4
5 // function main begins program execution
6 int main()
7 {
8     // variable declarations
9     int number1; // first integer to add
10    int number2; // second integer to add
11    int sum; // sum of number1 and number2
12
13    std::cout << "Enter first integer: "; // prompt user for data
14    std::cin >> number1; // read first integer from user into number1
15
16    std::cout << "Enter second integer: "; // prompt user for data
17    std::cin >> number2; // read second integer from user into number2
18
19    sum = number1 + number2; // add the numbers; store result in sum
20
21    std::cout << "Sum is " << sum << std::endl; // display sum; end line
22
23    return 0; // indicate that program ended successfully
24
25 } // end function main
```

خوارزم البرنامج السابق :

كود

- 1 Prompt the user to enter the first integer
- 2 Input the first integer
- 3

- 4 Prompt the user to enter the second integer
- 5 Input the second integer
- 6
- 7 Add first integer and second integer, store result
- 8 Display result

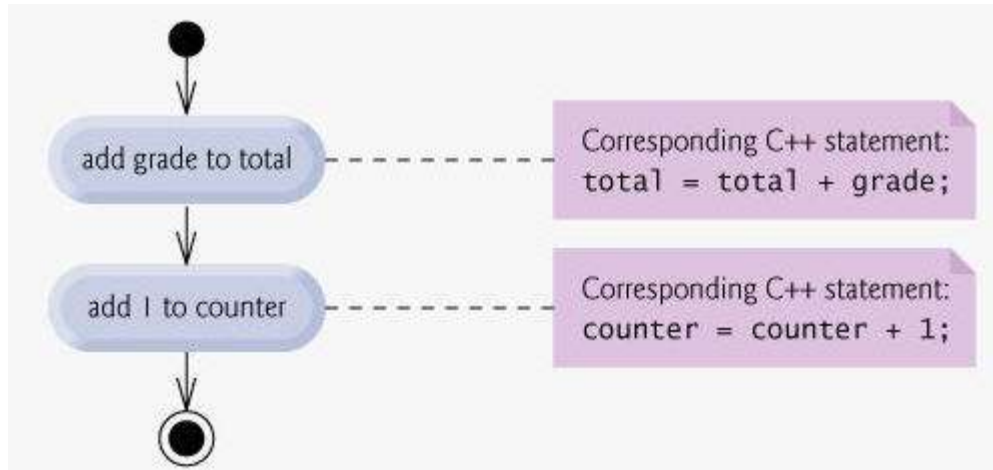
الخوارزم يمثل إدخال عددين صحيحين من المستخدم ، ثم يجمع هذين العددين ويعرض الناتج. السطور (١,٢) من الخوارزم تمثل السطور (١٣,١٤) من البرنامج. وهكذا . يستخدم شبه الكود لتمثيل العبارات في الدالة الرئيسية فقط (main) ، وليس لكامل البرنامج.

هياكل التحكم

عادةً العبارات في البرنامج تنفذ الواحدة تلو الأخرى في الترتيب الذي هي كتبت فيه ، هذا الترتيب للتنفيذ يسمى التنفيذ المتسلسل sequential execution. سوف نشرح فيما يلي خروج تقييد التنفيذ بالتسلسل عن الترتيب المتسلسل (اي انه ليس من المشترك أن تنفذ العبارة القادمة) .. وهذا يسمى نقل التحكم transfer of control. خلال السبعينات إتضح أن الإستخدام الجزافي في نقل التحكم سببه عبارة goto، التي تسمح للمبرمج أن ينقل التحكم إلى أي منطقته يريد من البرنامج، بحيث أصبحت الكثير من البرنامج شبيهه بمعكرونة الإسبقتي. أبحاث Bohm و Jacopini اثبتت أن البرامج يجب أن تكتب بدون عبارات goto، واصبح هذا تحدي المبرمجين في ذلك العصر. ولم يتسنى ذلك حتى السبعينات ، حيث تم نشر فكرة البرمجة الهيكلية (المركبة) ولكن لم تأخذ الجدية في تطبيقها. فيما بعد تم إتخاذ جدية أكثر في تطبيقها وأعطت النتائج انخفاض في وقت التطوير وسرعة في تسليم الأنظمة والمشاريع البرمجية. وتبين للجميع أن المفتاح هو إعتقاد البرمجة الهيكلية بشكل تام وعلى مدى أوسع ، إلى حد الإستغناء تماما عن عبارة goto. خرجت أعمال Bohm و Jacopini بأنه يجب الإستغناء عن عبارة goto بثلاث هياكل تحكم ، تحقق سرعة وأداء عاليين ، وهما ، الهيكل المتسلسل ، وهيكل الاختيار ، وهيكل التكرار. وسوف نشير إليهم في لغتنا هذه بعبارات التحكم (control statments).

نأخذ هذا الشكل ، وذلك لتوضيح مادتنا القادمة :

شكل ١-٥ :



الهيكل المتسلسل في الـ C++:

هو عبارة عن سلسلة بناء ، يكون ترتيب التنفيذ بها متسلسل وفق ترتيب العبارات ، سطر تلو الآخر، مالم ينتقل التحكم في إحدى العبارات ليكسر تسلسل الترتيب. الشكل السابق هو عبارته عن نموذج UML لمخطط حركة . وهو يمثل عمليتين حسابيتين منفذ بشكل متسلسل. في هذا الشكل ، العبارتين تقومان بجمع الدرجة لمتغير المجموع وجمع القيمة (١) لمتغير العداد. سوف نستخدم هذه العبارات للحصول على معدل درجات عدة طلاب بحيث يتم قسمة مجموع الدرجات على عددها. سوف نرى هذا في برنامج لاحظ ، وسوف يكون مثال بسيط لتوضيح الفكرة التي سوف نمارسها بشكل مكثف ومتطور في التمارين. مخططات الحركة تكون جزء من نماذج الـ UML لمخطط الحركة لسير عمل البرنامج. مخطط الحركة يتكون من رموز لها دلالات بحيث تصف حالة الإجراء وتسمى action state symbols (المعينات والدوائر الصغيرة المربوطة بسهام نقل تمثل سير الحركة). هذا الرسم حاله حال شبه الكود ، فهو يسهل للمبرمجين تطوير وتمثيل خوارزمياتهم بحيث تعرض كيفية عمل هياكل التحكم، بالرغم من أن الكثير يفضلوا شفه لأكواد في عمل ذلك. يحتوي مخطط حركة الهيكل المتسلسل في الشكل السابق على إجراءات (عبارات) يتم تنفيذها في البرنامج. الأسهم في الشكل تسمى أسهم انتقال، هذه الأسهم تمثل انتقالات التنفيذ (أولا ، جمع الدرجة للمجموع. ثانيا ، زيادة العداد بواحد). الدائرة المصممة في قمة المخطط تمثل أعداد قيمة أولية. في آخر الرسم يوجد دائرة مصممة محاطة بدائرة جوفاء وهي تمثل نهاية التنفيذ. في رسم الـ UML في الشكل السابق ، تلاحظون المستطيلات المطوية الزاوية ، تسمى (ملاحظات)، وهي ما تمثل تعليقات توضيحية تصف رموز المخطط. هذه الملاحظات تستخدم الكود المرتبط في كل حدث اجرائي في مخطط الحركة

عبارات الاختيار في الـ C++ :

في لغة الـ C++ لدينا ثلاث عبارات اختيار وهي : if , if .. else , switch

if : إذا تحقق الشرط (يكون صحيح) ينفذ الإجراء ، وإذا لم يتحقق يتجاوز .. فقط. وتسمى عبارة اختيار مفردة (لأنها تنفذ إجراء واحد فقط إذا تحقق الشرط).
if .. else : في حالة تحقق الشرط ، تؤدي إجراء معين ، وإذا لم يتحقق تنفذ إجراء آخر. وتسمى عبارة اختيار مزدوجة (لأنها تنفذ إجرائين ، عندما تكون صحيحة وعندما تكون خاطئة).
switch : تحتوي على عدة شروط ، كل شرط له إجراء خاص يتم تنفيذه إذا تحقق. وتسمى عبارة اختيار متعددة لأنها إجراءات متعددة ، لك إجراء شرط خاص تبني عليه.

عبارات التكرار في الـ C++ :

في لغة الـ C++ لدينا ثلاث عبارات تكرار وتسمى أيضا حلقات تكرار loops ، وهي : for , while

do .. while.

تؤدي هذه العبارات إجراء معين بشكل متكرر كل ما تحقق الشرط ، عدد هذه التكرارات تكون من الصفر إلى أن يفشل الشرط ، ما عدى العبارة do .. while ، فيكون من الواحد إلى أن يفشل الشرط ، لأنه هذه العبارة يجب تنفذ الإجراء مره واحدة على الأقل.

كل الكلمات (for , do , while , else , if , switch) كلمات مفتاحية (keywords) في الـ C++ هذه الكلمات محجوزة في الـ C++ لتطبيق تقنيات معينة.
فيما يلي قائمة بالمفتاحيات في الـ C++:

شكل ٥-٢ :

خطأ!

| C++ Keywords | | | | |
|---|--------------|----------|------------------|-------------|
| <i>Keywords common to the C and C++ programming languages</i> | | | | |
| auto | break | case | char | const |
| continue | default | do | double | else |
| enum | extern | float | for | goto |
| if | int | long | register | return |
| short | signed | sizeof | static | struct |
| switch | typedef | union | unsigned | void |
| volatile | while | | | |
| <i>C++-only keywords</i> | | | | |
| and | and_eq | asm | bitand | bitor |
| bool | catch | class | compl | const_cast |
| delete | dynamic_cast | explicit | export | false |
| friend | inline | mutable | namespace | new |
| not | not_eq | operator | or | or_eq |
| private | protected | public | reinterpret_cast | static_cast |
| template | this | throw | true | try |
| typeid | typename | using | virtual | wchar_t |
| xor | xor_eq | | | |

خطأ برمجي شائع :

إستخدام المفتاحية كمعرف يولد خطأ نصي (syntax error).

خطأ برمجي شائع :

حروف المفتاحية مع أي حرف كبير يولد خطأ نصي. كل مفتاحيات الـ C++ تحتوي حروف صغيره فقط.

خلاصة عبارات التحكم :

من كلامنا السابق ، نستفيد أنه يوجد لدينا ثلاث أنواع من عبارات التحكم (تسلسل ، اختيار ، تكرار).

لو نظرنا لكل كود (أو دالة عضو) لرأينا أنها تتكون من إحدى هذه الأنواع ، أو خليط منها يمثل الخوارزم الذي يطبقه هذا البرنامج.
نستطيع أن نقوم بعمل نموذج لكل عبارة تحكم ، وذلك بعمل مخطط حركة لوصف سلوكها بشكل رسومي. دائما ما يحتوي مخطط الحركة activity diagram على حدث أولي وحدث أخير (نقطة دخول ونقطة خروج) بشكل متعاقب يتخللها الإجراءات وفق ترتيبها وتسلسلها. وتسمى هذه الطريقة (التسلسل) بالترزيم (stacking) ، أي عمل العبارات بشكل رزمة ، كما في نموذجنا السابق (٥-١) الذي يمثل كودنا السابق. وهناك أيضا طريقة أخرى لإتصال (ترابط) عبارات التحكم ، تسمى هذه الطريقة بالتداخل (nesting). وهي الطريقة التي تكون فيها عبارات التحكم الواحدة داخل الأخرى.

خلاصة : يتكون كود ال ++C من ثلاث أنواع من علبوات التحكم ، تكون مرتبة بطريقتين ، التداخل والترزيم (stacking & nesting).
ومن الممكن أن يحتوي البرنامج على عبارات التحكم السبعة جميعها (المتسلسلة ، if .. else , if , switch, for, while , do .. while ,).

انتهت المحاضرة .

<http://www.arabteam2000-forum.com>