الدرس الرابع :
سوف نتناول في هذه المحاضرة مايلي :

١- كتابة اول برنامج.
٢- كتابة برنامج أخر يقوم بجمع عددين صحيحين.
٣- مفاهيم في الذاكرة.
٤- الحساب.
٥- اتخاذ القرار : المساواة والمقارنات العلائقية.

بسم الله نبدأ

عند بناء أي نظام أو تطبيق برمجي (برنامج) فإننا سوف نقوم بتحليله ومن ثم تصميمه وم ثم برمجته.

سوف نبدأ بعد قليل في كتابة الكود وسوف نتناول خمسة برامج ، ثلاثة منها لطباعة رسالة واثنين لمعالجة رقمين.

وهناك أيضا بعض الشرح للعمليات الرياضية ومفاهيم الذاكرة.

[size/]

[b]أول برنامج في الـ C++ ، طباعة سطر من النص.

[/b]سوف نناقشة فيما يلي برنامج بسيط يطبع سطر من النص ، هذا البرنامج يناقش عدد من المكونات الأساسية في برنامج الـ C++

كود

```
1  // Fig. 2.1: fig02_01.cpp
2  // Text-printing program.
3  #include <iostream> // allows program to output data to the screen
4
5  // function main begins program execution
6  int main()
```

```
7  {
8     std::cout << "Welcome to C++!\n"; // display message
9
10     return 0; // indicate that program ended successfully
11
12  } // end function main
```

السطرين (١,٢) :

كود

```
[size="3"] // Fig. 2.1: fig02_01.cpp
// Text-printing program.
[/size]
```

نلاحظ في بداية هذين السطرين العلامتين (//) وهي تعني أن ما يأتي بعدها إلى نهاية اسطر يعتبر تعليق وسوف يتجاهلة

المترجم تماما. والتعليقات مهمة في توضيح عمل البرنامج وجعله مفهوم عند عمليات الصيانة والتطوير.

السطر (٣) :

يعتبر السطر كاملا سطر موجه للمعالج الأولي وذلك لأنه ابتدأ بالعلامة #.

الـــ include <iostream> [font/]هي الأمر الموجه للمعالج وهو يقول له : " ضمن محتويات الملف <iostream> في البرنامج "

ويسمى الملف < iostream > بالـ[font="Calibri"]header file أي ملف الرأس ، وهذا الملف ضروري عندما نريد أن نقرأ البيانات وعندما

نريد أن نطبع على الشاشة.

خطأ برمجي شائع

نسيان الـ # أو السطر بكاملة ، وهذا يلود خطأ للمترجم. وذلك لأن المترجم لم يتعرف على الـ cout أو الـ cin (احد كائنات الإدخال

والإخراج).

السطر (٦) :

كود

[size="3"] int main()
[/size]

الدالة الرئيسية ، جزء أساسي من كل برنامج. والأقواس هذه تعني أن الـ main عبارة عن قطعة بناء تسمى الدالة (function).

برنامج الـ C++ عادةً ما يتكون من دالة واحدة على الأقل ، ودالة رئيسية على الأكثر.

عندما يبدأ التنفيذ لأي برنامج فإنه يبحث عن هذه الدالة لكي يبدأ التنفيذ منها ولا يشترط أن تكون هذه الدالة في بداية البرنامج

أو في مكان محدد. والـ int تشير إلى أن الدالة تستقبل عدد صحيح (فقط ، افهمها هكذا إلى حين شرحها بالتفصيل).

السطر (٧) :

كود

[size="3"]{[/size]

({) ويسمى قوس اليسار ، أو قوس الفتح ، وهو يشير إلى بداية جسم الدالة،
وينتهي عند السطر (١٢) ، } ، قوس اليمين

، او قوس الإغلاق ، وهو يشير إلى نهاية جسم الدالة.

السطر (٨) :

كود

```
std::cout << "Welcome to C++!\n"; // display message
```

من بداية السطر إلى الفاصلة المنقوطة يسمى " عبارة " ، وتشير الفاصلة المنقوطة إلى نهاية العبارة والسطر بأكملهيخبر الكمبيوتر بأن يؤدي عملية إخراج الرسالة مابين علامتي التنصيص على الشاشة. مابين علامتي التنصيص يسمى

في كثير من الاحيان سلسلة حرفية literal string او سلسلة string أو رسالة message وفيما يلي شرح السطر بالتفصيل :

Std :: cout :الـ std namespaces هي مساحة اسماء تحتوي على الكثير من الأسماء مثل (cout، cin, endl ... الخ)،

والجزء يعني أننا نستخدم إسم (وهو في هذه الحالة cout) الذي ينتمي إلى مساحات الأسماء std. ولا ننسى أن هذا الجزء

عبارة عن كائن.

<< : تسمى عامل إدراج السيل، ويسمى ما يليه على اليمين بالمعامل ، وهو ما سوف يقوم بإدراجة في سيل الإخراج. لاحظ

اتجاه هذه العلامة يكون بإتجاه ذهاب البيانات (هل هو اخراج أم ادخال).

" welcome to C++ \n " : وهي الرسالة التي سوف تظهر على الشاشة. (\n) لن تطبع على الشاشة شكليا، بل سوف

يتم القفز إلى السطر الجديد عن طريقها. (\ ) تسمى رمز هروب والحرف n يوحد معه ليشكل سلسلة هروب لسطر جديد.

وفيما يلي قائمة من سلاسل الهروب :

Escape sequence	Description
\n	Newline. Position the screen cursor to the beginning of the next line.
\t	Horizontal tab. Move the screen cursor to the next tab stop.
\r	Carriage return. Position the screen cursor to the beginning of the current line; do not advance to the next line.
\a	Alert. Sound the system bell.
\\	Backslash. Used to print a backslash character.
\'	Single quote. Use to print a single quote character.
\"	Double quote. Used to print a double quote character.

خطأ برمجي شائع :

نسيان الفاصلة المنقوطة في آخر العبارة سوف يولد خطأ عند تصحيح البرنامج (خطأ ترجمة).

السطر (١٠) :

كود

```
return 0; // indicate that program ended successfully
```

إنهاء دالة الـ main عند هذا السطر وإرجاع القيمة صفر، والصفر هنا يعني أن البرنامج تم إنهاؤه بنجاح. وسوف نشرح الدوال

في الفصول القادمة.

السطر (١٢) :

كود

```
[size="3"]  }
[/size]
```

إغلاق جسم الدالة الرئيسية.

لممارسة برمجية جيدة :

طباعة سطر أخر الدالة لتأكيد انتهاء ادالة والإنتقال لدالة جديدة .

ترك فراغ عند بداية السطر بعد فتح قوس الإبتداء للدالة ويفضل ترك ثلاث فراغات.

تعديل بسيط على البرنامج السابق .

كود

```
[size="3"]
[/size]

[size="3"]1  // Fig. 2.3: fig02_03.cpp
2  // Printing a line of text with multiple statements.
3  #include <iostream> // allows program to output data to the screen
4
5  // function main begins program execution
6  int main()
7  {
8     std::cout << "Welcome ";
9     std::cout << "to C++!\n";
10
11     return 0; // indicate that program ended successfully
12
13  } // end function main

[/size]
```

عندما نبدل السطر (٨) في البرنامج الأول بالسطرين التاليين في البرنامج الثاني فإنه (البرنامج الثاني) سوف يطبع الجملة في بنفس الطريقة السابقة.

مع ملاحظة وضع الفراغ بعد كلمة welcome .

تعديل آخر ، عندما نستبدل السطر (٨) بالسطر :

كود

```
[size="3"]std::cout << "Welcome\nto\n\n C++!\n";[/size]
```

فإن البرنامج سوف يطبع الجملة في ثلاث سطور والسبب يرجع لإدراج سلسلة الهروب لسطر جديد ثلاث مرات.

جربوا ذلك .

[b]برنامج جمع عددين

[b/]

برنامجنا التالي يقوم بجمع عددين يتم ادخالهما عن طريق المستخدم بواسطة لوحة المفاتيح. ويتم ذلك عن طريق

كائن الإدخال std :: cin وعامل الإستخلاص >> . ومن ثم جمع هذين العددين وطباعة النتيجة على الشاشة

عن طريق كائن الإخراج std :: cout .

كود

```
[size="3"] 1  // 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::end1; // display sum; end line
22
23    return 0; // indicate that program ended successfully
24
25  } // end function main
```

[/size]

السطور (من ١ إلى ٧) :

تم شرحها في برنامجنا السابق.

السطور (٩، ١٠، ١١) :

٩

كود

```
    int number1; // first integer to add
10    int number2; // second integer to add
11    int sum; // sum of number1 and number2
```

عمل تصريح أو إعلان عن ثلاث متغيرات sum, number1, number2 .

المتغير ما هو إلا مكان في ذاكرة الكمبيوتر يتم تخزين قيمة معينة لإعادة استخدامها فيما بعد.

التصريح يقول في هذه السطور : " قم بإنشاء ثلاثة متغيرات من نوع أعداد صحيحة int اسماؤها كالتالي

sum, number1، number2 ". فهذا يعني أن هذه المتغيرات تحتوي على أعداد صحيحة مثل (٠، -١١ ،

٧ ، ٦٩٣٨٠٧٣).

عندما نصرح عن متغير ما يجب ان نسميه ونحدد نوعه. في صياغة أخرى ، قبل أن نستخدم أي متغير في البرنامج يجب

أن يتم الإعلان عنه مسبقاً وأن يكون ذا اسم ونوع محددين. ولا يشترط ذلك أن يكون في أول البرنامج ولا في أي مكان

محدد.

ويمكنك أيضا أن تعرف كل المتغيرات في سطر واحد. فيمكن كتابتها بهذا الشكل :

كود

```
int sum ,number1, number2;
```

بعض المبرمجين يقومون بوضع كل تصريح في سطر ، وذلك لتسهيل التعليق عليها.

لممارسة البرمجة بشكل جيد :

ضع فراغ بعد كل فاصلة ، وذلك لجعل البرنامج سهل القراءة . بعض المبرمجين يقومون بوضع كل تصريح في

سطر ، وذلك لتسهيل التعليق عليها .

أيضا هناك أنواع اخرى من البيانات ، ومنها النوع double الذي يمثل الأعداد الحقيقية ، والنوع char الذي يمثل الرموز،

وأنواع أخرى.

هذه الأنواع الثلاثة (int, double ، char) تسمى بالأنواع الرئيسية ، أو الأنواع البدائية ، الأنواع المضمنة أو الضمنية.

عندما نسمي متغير فيجب أن نضع في إعتبارنا عدة امور ، أهمها :

١- ان يكون خليط من حروف وارقام ، بالإضافة إلى الشرطة السفلية (لا يتجاوز ذلك).

٢- أن لا يكون من الأسماء المحجوزة (الكلمات المفتاحية) مثل (class, int ، char .. الخ).

٣- أن لغة الـ C++ لغة شديدة الحساسية ، أي أنه لو اختلف حرف واحدفي المتغير فسوف يعتبر متغير آخر

، أي أن a1 ليست كالـ A1 .

فكرة برمجية مفيدة :

ليس هناك طول محدد للمعرف في الـ C++ ، ولكن ربما التطبيق أو البرنامج الذي تعمل عليه يفرض عليك طول معين .

استخدم معرِّف بـ ٣١ رمز أو اقل لتحقيق النتقلية .

لممارسة برمجية جيدة :

أختر اسم ذا معنى وتجنب استخدام الإختصارات ، وذلك لجعل قراءة البرنامج مفهومة اكثر، وتجنب دائما الاسماء التي تبدا

بخط سفلي طويل وذلك لعدم التشويش بينها وبين مسميات الـ C++ عند عملية الترجمة .

لمنع حدوث الأخطاء :

يجب علينا تجنب الأسماء التي من الحتمل أن تصبح كلمات محجوزة في المستقبل، مثل كلمة object ومن في نفس حالتها ، فإن

هذه الكلمات يمكن استخدامها الآن ولكن هناك احتمال كبير بأن تكون كلمات محجوزة في المستقبل.

السطر (١٤) :

كود

```
14    std::cin >> number1; // read first integer from user into number1
```

استخدمنا كائن الإدخال cin وهو اسم من اسماء مساحة الأسماء std واستخدمنا ايضا عامل الإستخلاص >> لأخذ القيمة من

لوحة المفاتيح. يقوم الكائن std :: cin بأخذ القيمة من المستخدم وتخزينها في المتغير الذي يلي عامل الإستخلاص >> .

لمنع حدوث الاخطاء :

يقوم البرنامج بالتأكد من صحة القيم المدخلة وذلك لمنع المعلومات الخاطئة من التأثير على حسابات البرنامج .

كيف تحدث عملية الإسناد في السطر (١٤) ؟

عند تنفيذ البرنامج يقف سير التنفيذ عن هذا السطر ومن ثم ينتظر المستخدم بأن يقوم بعملية الإدخال ،يستجيب المتستخدم

بطباعة عدد صحيح (الكمبيوتر يتعبره رمز) ، ثم يضغط مفتاح الإدخال enter لإرسال

الحرف للكمبيوتر. يقوم الكمبيوتر

بتحويل تمثيل الرمز إلى عدد صحيح ومن ثم يقوم بإسناد نسخة منه للمتغير number1 . وبعد ذلك يشير الـ number1 إلى

هذه القيمة المدخلة في بقية البرنامج.

السطر (١٩) :

كود

```
[size="3"]19     sum = number1 + number2; // add the numbers; store result in
sum
[/size]
```

عملية حاصل اسناد حاصل جمع العددين number1 و number2 للمتغير sum .

تسمى العمليتيين (+ ، =) عمليات ثنائية ، أ ي أن كل عملية تحتوي على معاملين ، في عملية الـ + المعاملان هما (number1،

numbe2) ، وفي عملية الـ = فإن المعاملان هما (حاصل جمع العددين ، المتغير sum). وذكرنا فيما سبق أن السطر بأكمله

إلى الفاصلة المنقوطة يطلع عليه مسمى عبارة.

لممارسة برمجية جيدة :

ضع فراغ إلى كل جانب من جوانب العملية الثنائية . على سبيل المثال ، اكتب sum = number1 + number2 بدلا من Sum=number1+number2 ، وذلك من أجل ترتيب البرنامج ووضوحه.

السطر (٢١) :

كود

```
21     std::cout << "Sum is " << sum << std::end1; // display sum; end line
```

هي تقريبا عمليتين في سطر واحد ، طباعة قيمة عدد صحيح وطباعة متسلسلة حرفية (القيمة هي sum والمتسلسلة هي sum is).

هناك اسم جديد من أسماء مساحة الأسماء std وهو الـ endl وهو يقوم بطباعة سطر جديد.

يمكن أن يكتب السطرين (١٩، ٢٠) في سطر واحد بالشكل التالي :

كود

[size="3"] std::cout << "Sum is " << sum << std::endl; // display sum; end line

[/size]

وبهذه الطريقة نستغني عن التصريح عن المتغير sum .

[b]مفاهيم في الذاكرة

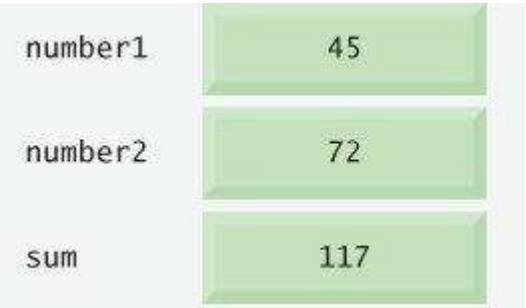
[/b]المتغيرات sum، number1, number2 هي عبارة عن مواقع في الذاكرة. كل متغير له إسم وحجم وقيمة ونوع. في

برنامج جمع عددين ، في السطر (١٤) في العبارة : std :: cin >> number1 ؛

عندما يتم ادخال الرموز إلى الكمبيوتر، يحول تمثيلها إلى أعداد صحيحة ويتم وضع كل مدخل في موقعة في الذاكرة.

المدخل الأول في البرنامج يكون موقعة number1 .

الرسم التالي للتوضيح :



نفترض أن هذا الموقع يحتوي على قيمة سابقة ، فإن وضع القيمة الجديدة فإن القيمة القديمة سوف تدمر ولا يمكن استردادها.

هذه المواقع لا يشترط ان تكون متجاورة في الذاكرة فيمكن أن تكون متفرقة.

في عملية الجمع والإسناد : number1 + number2 = sum لا تتغير القيم number1 ،

number2 . بل الذي يتغير

هو محتوى الـ sum .

[b]طريقة الحساب

[/b]معظم البرامج تقوم بعمليات حسابية. الشكل التالي يلخص العمليات الحسابية في الـ c++ :

C++ operation	C++ arithmetic operator	Algebraic expression	C++ expression
Addition	+	$f + 7$	f + 7
Subtraction	-	$p\ c$	p c
Multiplication	*	bm or $b \cdot m$	b * m
Division	/	x / y or $\frac{x}{y}$ or $x \div y$	x / y
Modulus	%	$r \bmod s$	r % s

لاحظ استخدام رموز خاصة لم تستخم في الجبر. علامة النجمة (*) تشير للضرب ، وعلامة النسبة

المئوية (%) تشير إلى باقي القسمة وتسمى الـ modulus التي سوف يتم شرحها بإختصار.

في عملية القسمة هناك نقطة جديرة بالإشارة إليها وهي ، عند عملية قسمة العدد الصحيح (البسط والمقام اعداد صحيحة)

يكون ناتج القسمة عدد صحيح. على سبيل المثال التعبير ٤/٧ يعني ناتج ١ والتعبير ٥/١٧ يعطي الناتج ٣ . وذلك لأن

الأجزاء والكسور في ناتج القسمة سوف يتم طرحهما من الناتج.

يجب أن يكون المعاملين للعملية % أعداد صحيحة. التعبير x/y ينتج عنه باقي القسمة

من x/y . أيضا ٧%٤ ينتج عنه

العدد ٣ ، والـ ٧%٥ ينتج عنه الناتج ٢.

خطأ برمجي شائع :

محاولة استخدام الـ % مع معاملات ليست اعداد صحيحة سوف يولد خطأ في الترجمة.

التعبيرات الحسابية في خط مستقيم :

برغم أن ه ذا يجعل شكل البرنامج أبعد عن الطبيعة إلا أن المترجم لا يقبل العمليات التي لا تكون بخط مستقيم، على سبيل

المثال العملي $\frac{a}{b}$ حتى تنفذ على البرنامج يجب كتابتها بالشكل a/b .

الأقواس لتجميع التعبيرات المتفرعة :

الأقواس في لغة الـ C++ تأخذ الشكل نفسه في التعبير الجبري ، مثلا ، لضرب المقدار c+b بعدد a ، نكتبه بالشكل التالي :

. (a * (a + b

ترتيب الأوليات في التعبيرات الحسابية :

الـ C++ تطبق العمليات في التعبيرات الرياضية في سلسلة وترتيب دقيق محدد بقواعد أولوية مسبقة. وهذا التطبيق يكون

بالترتيب التالي :

١- أولا تنفذ العمليات داخل الأقواس. وتستخدم الأقواس احيانا لجبر سير الترتيب في الطريق الذي نريد. عندما تكون الأقواس متداخلة فإن الأقواس الأعمق هي التي تنفذ أولا.

٢- ثانيا، عمليات الـ (%، /، *) تأتي بعد الأقواس ولها نفس المستوى ، وعند تتواليها في عملية حسابية تطبق من اليسار لليمين.

٣- ثالثا ، عمليات الـ (+، =) تأتي بعد العمليات السابقة في التنفيذ. ايضا إذا اتت بشكل متعاقب فإنها تطبق من اليسار لليمين.

عندما يكون هناك مجموعة عمليات في مستوى واحد أتت بشكل متعاقب في تعبير رياضي ونقول أن تطبيقها يكون من اليسار

لليمين أو العكس ، فإننا نشير هنا إلى الترابطية associativity .

وفيما يلي قائمة بالعمليات الحسابية وأولويات تنفيذها :

Operator(s)	Operation(s)	Order of evaluation (precedence)
( )	Parentheses	Evaluated first. If the parentheses are nested, the expression in the innermost pair is evaluated first. If there are several pairs of parentheses "on the same level" (i.e., not nested), they are evaluated left to right.
* / %	Multiplication Division Modulus	Evaluated second. If there are several, they are evaluated left to right.
+ -	Addition Subtraction	Evaluated last. If there are several, they are evaluated left to right.

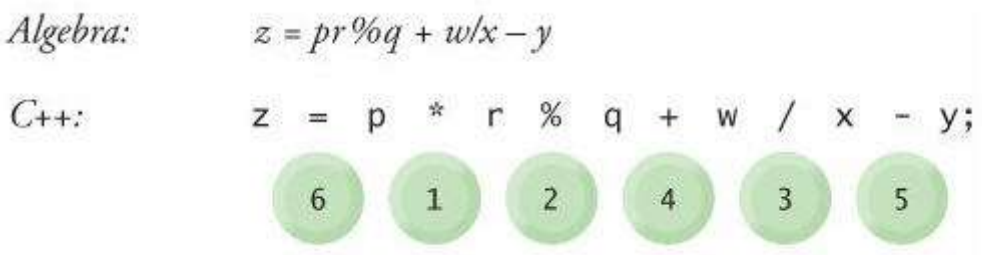
تعبير جبري ومقابله في الـ++C :

الجبري $m = \frac{a+b+c+d+e}{5}$

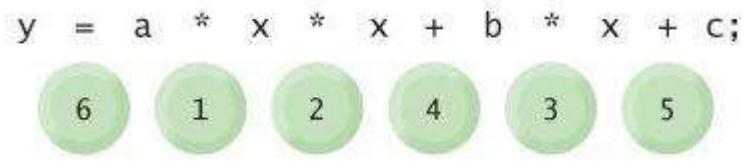
الـ ++C m = (a + b + c + d + e) / 5 الأقواس ضرورية هنا.

لنوضح ترتيب التنفيذ في معادلة معينة، لنأخذ العملية التالية (y – z = pr % q + w / x) فإن ترتيب التنفيذ

سوف يكون بالشكل التالي :

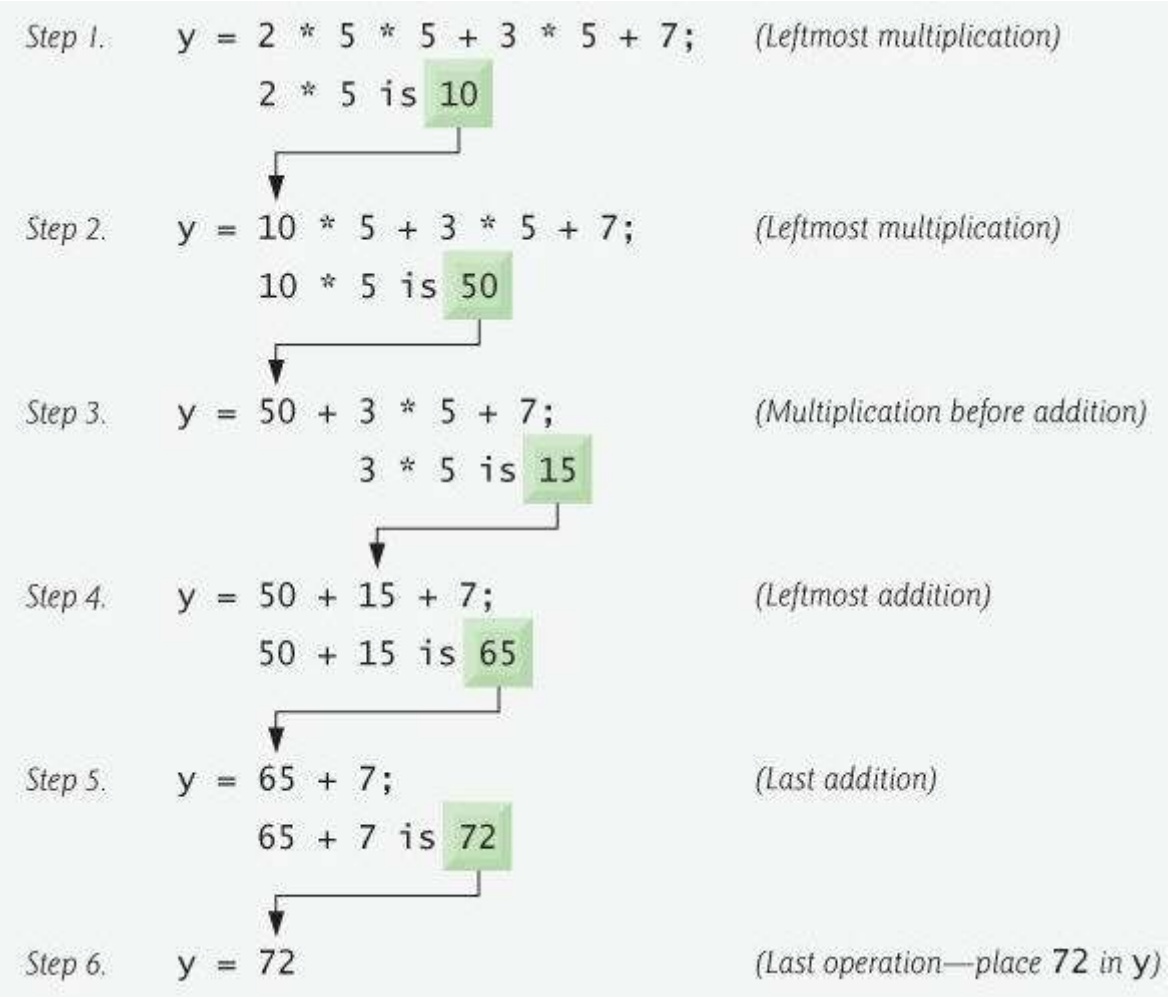


أيضا سوف نرى الأولويات في المعادلة من الدرجة الثانية ($y = ax^2 + bx + c$) تكون بالشكل التالي :



بالرسم التوضيحي سوف نوضح طريقة الحصول على النتيجة من تطبيق الأولية الحصيحة على مرعادلة :

خطأ!



[b]اتخاذ القرار : المساواة والمقارنات العلائقية

[/b]

سوف نقدم تحت هذا العنوان أداة مهمة في الـ C++ وهي العبارة الشرطية (if).

وعن طريقها يمكن للبرنامج أن

يتخذ قرار وينفذ عملية معينة يعتمد على نتاجها (صائب أم خاطئ). سوف نعطي مثالا على ذلك.

نستخدم أثناء العبارة الشرطية (if) بعض عمليات المقارنة ، اختصرناها في الشكل التالي :

Standard algebraic equality or relational operator	C++ equality or relational operator	Sample C++ condition	Meaning of C++ condition
Relational operators			
>	>	x > y	x is greater than y
<	<	x < y	x is less than y
≥	>=	x >= y	x is greater than or equal to y
≤	<=	x <= y	x is less than or equal to y
Equality operators			
=	==	x == y	x is equal to y
≠	!=	x != y	x is not equal to y

كلها لها نفس الأولوية وترابطيتها من اليسار لليمين.

خطأ برمجي شائع :

لا تضع أي فراغ خلال عمليات المقارنة (== ، !=، =< .. الخ) أو أن تقوم بعكس ترتيبهما (كأن تكتب != بدلا من =!) فإن

ذلك يؤدي إلى خطأ نصي.

خطأ برمجي شائع :

عدم التفريق بين الـ = والـ == ، فالاولى عملية اسناد ، والثانية عملية مقارنة ، والنتيجة خطأ منطقي.

فيما يلي سوف نشرح برنامج يحتوي على ست عمليات مقارنة .

كود

[size="3"]
[size="3"][/size]1 // Fig. 2.13: fig02_13.cpp
2 // Comparing integers using if statements, relational operators
3 // and equality operators.
4 #include <iostream> // allows program to perform input and output
5
6 using std::cout; // program uses cout
7 using std::cin; // program uses cin
8 using std::endl; // program uses endl
9
10 // function main begins program execution
11 int main()
12 {
13 int number1; // first integer to compare
14 int number2; // second integer to compare
15
16 cout << "Enter two integers to compare: "; // prompt user for data
17 cin >> number1 >> number2; // read two integers from user
18
19 if (number1 == number2)
20 cout << number1 << " == " << number2 << endl;
21
22 if (number1 != number2)
23 cout << number1 << " != " << number2 << endl;
24
25 if (number1 < number2)
26 cout << number1 << " < " << number2 << endl;
27
28 if (number1 > number2)
29 cout << number1 << " > " << number2 << endl;
30
31 if (number1 <= number2)
32 cout << number1 << " <= " << number2 << endl;
33
34 if (number1 >= number2)
35 cout << number1 << " >= " << number2 << endl;
36
37 return 0; // indicate that program ended successfully
38
39 } // end function main

[/size]

السطور (من ٦ إلى ٨) :

كود

[size="3"]
[size="3"][/size]

using std::cout; // program uses coutusing std::cin; // program uses cinusing std::endl; // program uses endl[/size]

استخدمنا التصريح using الذي يزيل الحاجة إلى إعادة استخدام البادئة std :: في البرنامج. أي انه بعد نستخد cin

، cout مباشرة بدلا std :: cout ، std :: cin في ، نكتبها مره في بداية البرنامج ونوفر جهد استخدامها بشكل متكرر

في باقي البرنامج.

لممارسة برمجية جيدة :

وضع using مباشرة بعد #include .

السطور (١٣، ١٤) :

كود

```
13    int number1; // first integer to compare
14    int number2; // second integer to compare
```

الإعلان عن المتغيرات في البرنامج.

السطور (١٩، ٢٠) :

كود

```
19    if ( number1 == number2 )
20       cout << number1 << " == " << number2 << endl;
[size="3"][/size]
```

مقارنة القيم number1، number2 لفحص شرط المساواة. إذا القيمتين متساويتين ، العبارة في السطر ٢٠ سوف تنفذ.

واذا لم يتحقق الشرط لا تنفذ العملية وينتقل التنفيذ إلى السطور التالية في البرنامج وتطبق باقي المقارنات.

لاحظ أن كل عبارة شرطية في برنامجنا هذا لديها عبارة واحد في التنفيذ اذا احتوى الشرط أكثر من عبارة (أكثر من فاصلة

منقوطة) فإن جسم العبارة الشرطية يجب أن يبدأ بالقوس { وينتي بالـ } . يسمى ما بين القوسين بالقطعة (block).

لممارسة برمجية جيدة :

لتسهيل القراءة لا تستخدم اكثر من عبارة شرطية في كل سطر .

خطأ برمجي شائع :

وضع فاصلة منقوطة مباشرة بعد اقواس العبارة الشرطية على اليمين كما يلي : if (~) ؛ فإن ذلك سوف يجعل الشرط

فارغ وجملة الشرط سوف تنفذ في كل الحالات .

انتهت المحاضرة

http://www.arabteam2000-forum.com