Chapter 1

Introduction to computer, programs and **c++**

Section 1.1-1.3,1.6-1.9

What is the programming ?

- كثير بنسمع عن مصطلح البرمجة او programming ، **بس شو هي البرمجة ؟**

البشر لما تتواصل مع بعضها البعض ، بتحتاج طريقة لتواصل ، وهي اللغة

نفس الاشي لما البشر بدها تتواصل مع الحاسوب بنتحتاج البرمجة

- يعني **البرمجة**: هي طريقة تواصل البشر مع الحاسوب ، عن طريق كتابة بعض الأوامر او codes وزي ما في لغات لتواصل البشر مع بعض ، في العديد من لغات البرمجة منها : (C++ , C# , Java , Python) والعديد من اللغات الاخرى

- **شو الفرق بين لغات البرمجة** ؟ لغات البرمجة بتتشابه في الأفكار والمبدأ لكن تختلف من ناحية الاستخدامات ، وعند اتقانك للغة وحدة بتقدر تتعلم غيرها من اللغات وتتمكن منها

- وفي هذا الكورس رح نتعلم كيف البرمجة عن طريق لغة ال C++ وان شاء الله رح تكون هاي الدوسية شاملة وكافية للكتاب والسلايدات واسئلة السنوات

- **لشو رح تأهلنا المواضيع الي رح نتعلمها خلال الكورس** ؟ رح نطلع متقنين لغة ال C++ بجميع مفاهيمها واساسياتها ورح نكون قادرين على حل المشاكل البرمجية او بما يعرف بال Problem Solving

What is a computer ?

- بالبداية وقبل ما نتعلم البرمجة **ما هو الحاسوب ؟**

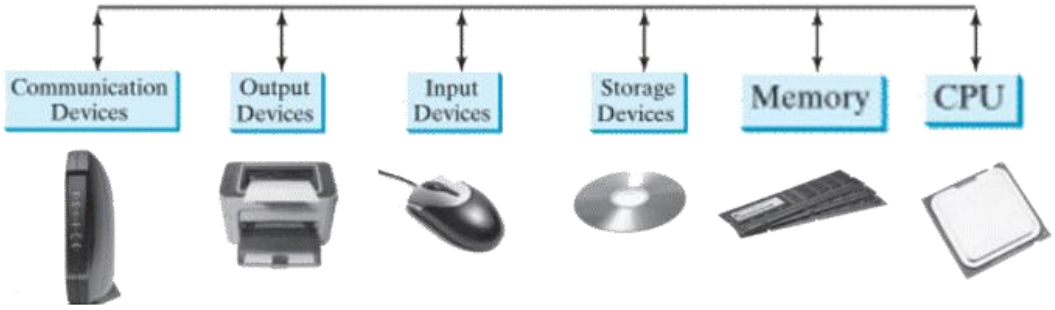
A **computer** is an electronic device that stores and processes data, and includes hardware and software.

Hardware --> physical aspect of the computer that can be touched

Software --> the invisible instructions that control the hardware and make it perform tasks.

- ما هي أجزاء الحاسوب ؟

A computer consists of a CPU, memory, hard disk, monitor, and communication devices.



وسنتعرف على بعض الأجزاء : (CPU , Memory , Storage , Devices)

- The central processing unit (CPU):

وحدة المعالجة المركزية ، او كما يعرف بـ عقل الحاسوب ، وهو اهم جزء من أجزاء الحاسوب

- وظيفته هو ترتيب وتنفيذ ال codes الي بتمثل الأوامر الي بدنا الحاسوب ينفذها مثال:

c = a + b;

هون ال **CPU** اجا امر بجمع **a** مع **b** ويسيف الناتج في الناتج بـ c

- طبعا اهم اشي بال **CPU** انه يكون سريع ، حتى ينفذ ال code بسرعة اكبر وتقاس سرعة ال **CPU** بالـ megahertz (MHz)

1 megahertz ⟵	بنفذ مليون أمر في الثانية

وممكن تقاس سرعة الـ CPU بالـ Gegaheartz الي بتساوي 10^9 بالثانية الوحدة !

وطبعا كل ما كان ال CPU اسرع بكون افضل و تكلفته اعلى .

- memory

ينفذهمCPU حتى الـ code- وهي ذاكرة الحاسوب، وبتخزن فيها البرنامج والـ

: byte هي الـ memory- والوحدة المستخدمة في التخزين في داخل الـ

1 byte	تساوي →	8 bits

0 او 1 . هي اصغر وحدة تخزين وبتتكون من قيمتين فقطbitوالـ

. **hard disk** في data الكهرباء بتحتفظ بجميع الـ memoryواذا انقطعت عن الـ

- Storage Devices

من الضياع dataهي الذاكرة الدائمة في الحاسوب، وفيها بتم الاحتفاظ بجميع الـ
بداخلها dataحتى مع انقطاع الكهرباء تبقى محافظة على جميع الـ

هي :Storage Devices- واهم 4 أنواع لـ

- Disk drives (hard disks)
- Solid-state devices (SSD, Flash),
- CD drives (CD-R and CD-RW)
- Tape drives

- How data is **store ?**

- لو مثلا نتخيل انه عنا محل وهذا المحل بيجي طلبات ، لكل طلب رح يكون في اله **رقم الطلب ومحتوى هذا الطلب** وطبعا كل الطلبات رح تتخزن **بالترتيب** الي وصلته .

نفس الاشي ال Memory رح تحتفظ بال Data بترتيب الي وصلها ، ورح يكون لكل ***Data*** رقم يدلني عليها الي هو الـ ***address***

address : هو الرقم بمثل ال location لمكان ال **Data** بداخل ال meomry

وبتم تقسيم ال memory الى مواقع كل موقع بحتوي على address و data

address	data	
.	.	
.	.	
2000	01000011	Data for character C
2001	01110010	Data for character r
2002	01100101	Data for character e
2003	01110111	Data for character w
2004	00000011	Data for number 3
.	.	

وزي ما اتفقنا الـ data بتتخزن على شكل **1 byte** الي هي **8 bits** .

وال data ممكن تكون أحرف او أرقام وفي بعض الاحيان بتكون الdata اكبر من انها تتخزن ب **1 byte** فعشان هيك ممكن نستخدم **2** او **3** byte حتى نخزن ال Data.

ما رح نهتم بكيف الـ data بتتحول لـ 0 و 1 المهم نفهم الفكرة العامة

- programs

مثل ما اتفقنا سابقًا لغة البرمجة هي طريقة التواصل الانسان مع الحاسوب،
وال program هو البرنامج الي بنكتبه وفي الأوامر الي لازم ينفذها الحاسوب.
وحتى نقدر نتخاطب مع الحاسوب لازم نختار لغة البرمجة المناسبة .

انواع لغات البرمجة :

Machine Language \ Assembly Language \ High-Level Language

- Machine Language :

ال machine language هي اللغة الي بفهمها الحاسوب وبتتكون من 0 و 1 فقط ، وكل سطر مكتوب بال 0 و1 اله معنى بفهمه الحاسوب

فمثال على عملية جمع رقمين:

1101 101010 011010

(ليست للحفظ)

- لكن عملية الكتابة بهاي اللغة شاقة وصعب تتبعها او تعديلها ، **فتطورت** اللغة الى النوع الثاني :

- Assembly Language :

لكتابة الكودات وتتم عن machine هي طريقة اسهل من الassembly language ال
، مثال:instruction طريق كتابة كلمات بتأدي نفس الوظيفة وتسمى بال

add result , 1 , 2

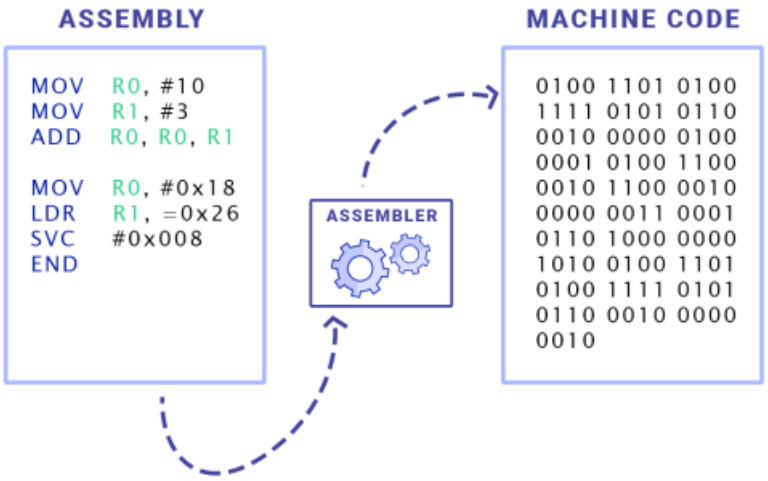
وهاد السطر بتحول لـ result 2 واحتفظ فيهم بـ اجمع 1 و M.L :

add number 1 number 2

101 000001 0000101

One line in **Assembly** == One line in **Machine code**

- **Assembler**تسمى بMachine الى Assemblyوالعملية الي بتم فيها تحويل من ال
- instruction وهو برنامج بحول كل الى 0 و 1 .



- High-Level Language :

هي اللغة الأقرب للغة الانسان وتشبه اللغة الإنجليزية ، ومن خلالها رح نتعلم البرمجة ،
مثال لحساب مساحة الدائرة بلغة ال c++ :

```
Area = 5 * 5 * 3.1416;
```

(لغة منطقية وقريبة من لغة الانسان)

- من الأمثلة على لغات البرمجة :

C++ , C , C# , Java , python

تم انشاء ال High-Level language حتى نتعامل معها بدلاً من ال assembly مما جعل عملية البرمجة ابسط وأسهل، وتختلف كل لغة برمجة عن الأخرى باستخداماتها لكنهم جميعهم متشابهين بالمبدأ العام والقواعد الأساسية

لكن كما تعلمنا سابقاً، ان الحاسوب ما بفهم الا لغة ال machine code الي هي 0 و 1
فبتم تحويل ال High level language الى ال machine code حتى يتم تنفيذها
وبتم احتفاظ بالكود بداخل ملف يسمى بال **source code**

ورح نتعرف على عملية تحويل ال source code الى machine code

Compiling source code :

- زي ما تعرفنا انه عنا 3 أنواع من لغات البرمجة ، والي رح نستخدمها في هذا الكورس هو النوع الثالث ال **High Level Language**

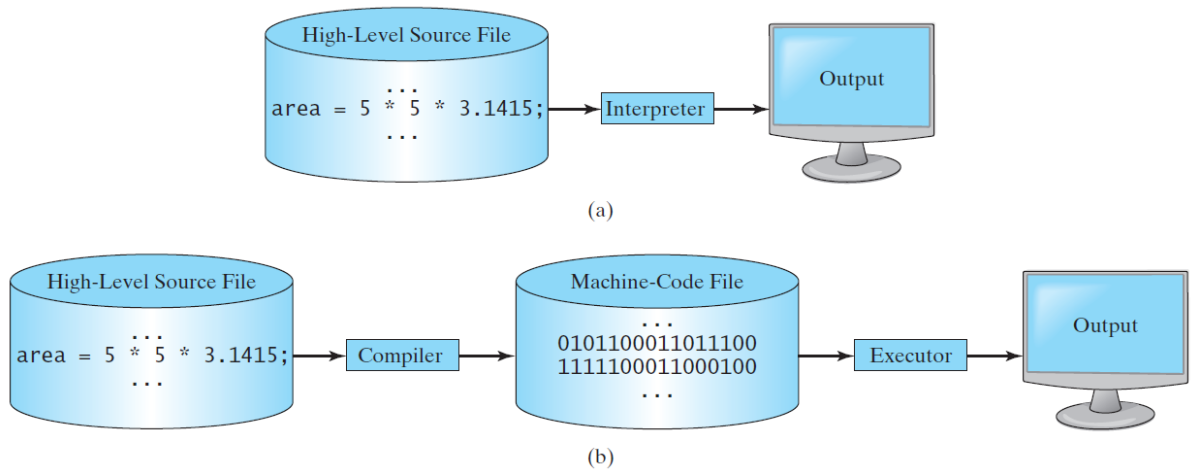
لكن اتفقنا انه الحاسوب ما بفهم هاي اللغة ، عشان هيك لازم نحول الكود من ال H.H.L الى ال M.L **كيف** ؟

بتم تحويل الـ source code عن طريق ال compiler الى object program بعد هيك بتحول الى Machine code وبتم تنفيذه .

Source code —(بتم تحويله الى)→ Object program —(بتم تحويله الى)→ Machine code

ويوجد طريقتين لتحويل الـ source code وهما :

- **interpreter :** بترجم وبنفذ البرنامج سطر سطر
- **compiler :** بترجم كل الكود بعدين بنفذ الكود كامل مرة وحدة



First simple c++ Program :

```
#include <iostream>                    ...  1
using namespace std;                   ...  2
int main()                             ...  3
{
    // Display Welcome to C++ to the console      ... 4
    cout << "Welcome to C++!" << endl;   ... 5
    return 0;                                   ... 6
}
```

- حتى نفهم اكثر كيف كتبنا هاد الكود وشو وظيفته خلينا نتبع اجزاءه وحدة وحدة :

لو تخيلنا انه عنا **لوحة** بدنا نرسم عليها :

- اول اشي رح نحتاج **ادوات** الرسم وبنجيبها من **محل** ببيع هاي الادوات

- وطبعا رح يكون عليهم **الشعار** الخاص بالشركة عشان نعرف من وين تم تصنيع كل اداة

- ولما ارسم رح استخدم لوحة الرسم الي بتمثل **حدود** الرسمة وبتمثل حيز العمل

بنفس هاي الفكرة رح تكون عندي بكتابة ال code :

- رح يكون في عنا **مكتبة** (#include <iostream>) الي هي المحل الي بناخذ منه **الادوات** الي بنحتاجها خلال كتابة الكود

- وعنا **الشعار** الي بدلني من وين جبت الاداة من اي مكتبة (using namespace std;)

- وعنا اللوحة الي رح نكتب فيها الكود تبعنا (int main( )) وحدود الcode رح تكون بين ال { } وتسمى بال **scope**. وبنتهي برنامجنا لما نوصل الى ال (return 0)

1) #include <iostream> :

بهذه الجزء بتم استدعاء المكتبات، واسم المكتبة بكون بين < > وهاي المكتبة بتمكنا من استخدام بعض الأوامر المفيدة (ورح نتعرف على العديد من المكتبات)

وواحدة من هاي المكتبات هي ال **iostream** :
الي بتحتوي على functions الادخال والاخراج **(cin , cout)**

2) using namespace std :

عند استخدام cin او cout كنا نلحقهم بـ std:: حتى نميز انهم من مكتبة iostrem
لكن الان using namespace std بتغني عن كتابتها

3) int main ( ) :

من هون ببلش كتابة الكود وتنفيذه، وال main يعتبر function وحدوده بتكون دائما بين ال { } وتسمى بال scope (اي اشي براتها ما رح يتم تنفيذه)

ملاحظة : - ال white space بتم اهمالها، واي سطر لازم ينتهي ب (;)

- الأجزاء 1 و 2 و3 دائما بكونوا موجودين في أي برنامج بدي اكتبه
مش مشكلة لو ما تم فهمهم بالبداية

4) //comments :

ال **Comments** هي عبارة عن جمل بتم كتابتها لكن الكمومبايلر بتجاهلها وما بنفذها ،
ويوجد منها نوعين :

- single line comment:

```
//this is single line comment
```

- multiple line comment:

```
/* this is
Multiple line
Comment */
```

5) Cout Statement:

- يوجد لدينا نوعين من العمليات مع الحاسوب :

- output : اخراج - input : ادخال

وبتم استخدام جملة ال cout لعمليات الإخراج على الشاشة ويوجد نوعين من الإخراج :

- طباعة أي نص بكون داخل ال (" ") - طباعة ناتج العمليات الحسابية

Example:

```
cout << " (10.5 + 2 * 3) / (45 - 3.5) " << endl;
output: (10.5 + 2 * 3) / (45 - 3.5)
```

تم طباعة النص الي بداخل " " كما هو

```
cout << (10.5 + 2 * 3) / (45 - 3.5) << endl;
output: 0.39759
```

تم حساب وطباعة ناتج العملية لأنها ليست بين " "

- من الممكن طباعة جملة واحدة ، او عدة جمل في سطر واحد :

cout<< "statement 2" << "statement 2" << .....<< "statement N" <<endl;

ال **endl** تعني اضافة سطر جديد عند انتهاء عملية الطباعة

6) return 0:

تعني انهاء الكود وعرض النتائج، وبتكون باخر سطر في ال `int main()`
وما بتم تنفيذ أي اشي بكون تحتها. (من الممكن الاستغناء عنها وعدم كتابتها)

Special Characters in C++

Character	Name	Description
#	Pound sign	تستخدم بداية تعريف المكتبة #include
<>	Opening and closing angle brackets	تستخدم لوضع داخلها اسم المكتبة
()	Opening and closing parentheses	تستخدم عند انشاء ال function رح نتعرف عليهم اكثر لقدام
{}	Opening and closing braces	هي عبارة عن block حتى يحصر اي جملة
//	Double slashes	تستخدم لوضع نوتس وما بتم قرأته
<<	Stream insertion operator	تستخدم مع جمل الطباعة cout<<
" "	Opening and closing quotation marks	بتم طباعة اي اشي بداخلها
;	semicolon	بنختم فيها كل سطر او جملة من كودنا

رح يتم حفظهم مع الممارسة

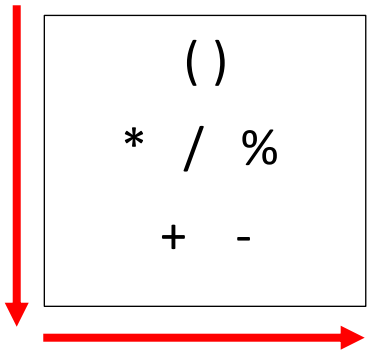
Numeric Operators

من الممكن استخدام لغة ال C++ لإجراء العمليات الحسابية

operation	Name	example
+	Addition	34 + 1 = 35
-	Subtraction	34.0 – 0.1 = 33.9
*	Multiplication	300 * 30 = 9000
/	Division	1.0 / 2.0 = 0.5
%	Modulus	20 % 3 = 2

الأولويات الحسابية :

الاقواس –> باقي القسمة الضرب والقسمة من اليسار–> الجمع والطرح من اليسار



ملاحظة :

- في العمليات الحسابية ، يكون الجواب دائما رقم صحيح
- الا اذا كان احد اطراف العملية عدد كسري يكون الجواب عدد كسري
- لا يمكن ان يكون احد اطراف ال % عدد كسري

Examples:

صحيح صحيح صحيح

1) cout << 5 + 8 ; → 13

cout << 4 - 3.5 ; → 0.5

كسري صحيح كسري

cout << 8.0 + 5 ; → 13.0

cout << 1.5 - 3.5 ; → -2.0

2) cout << 5 / 2 ; → 2

cout << 2 * 2.5 ; → 5.0

cout << 5.0 / 2 ; → 2.5

cout << 2 * 2 ; → 4

3) cout << 10 % 3 ; → 1

cout << 10 % -3 ; → 1

cout << -10 % 3 ; → -1

cout << -10 % -3 ; → -1

الـ % تتبع إشارة الأكبر

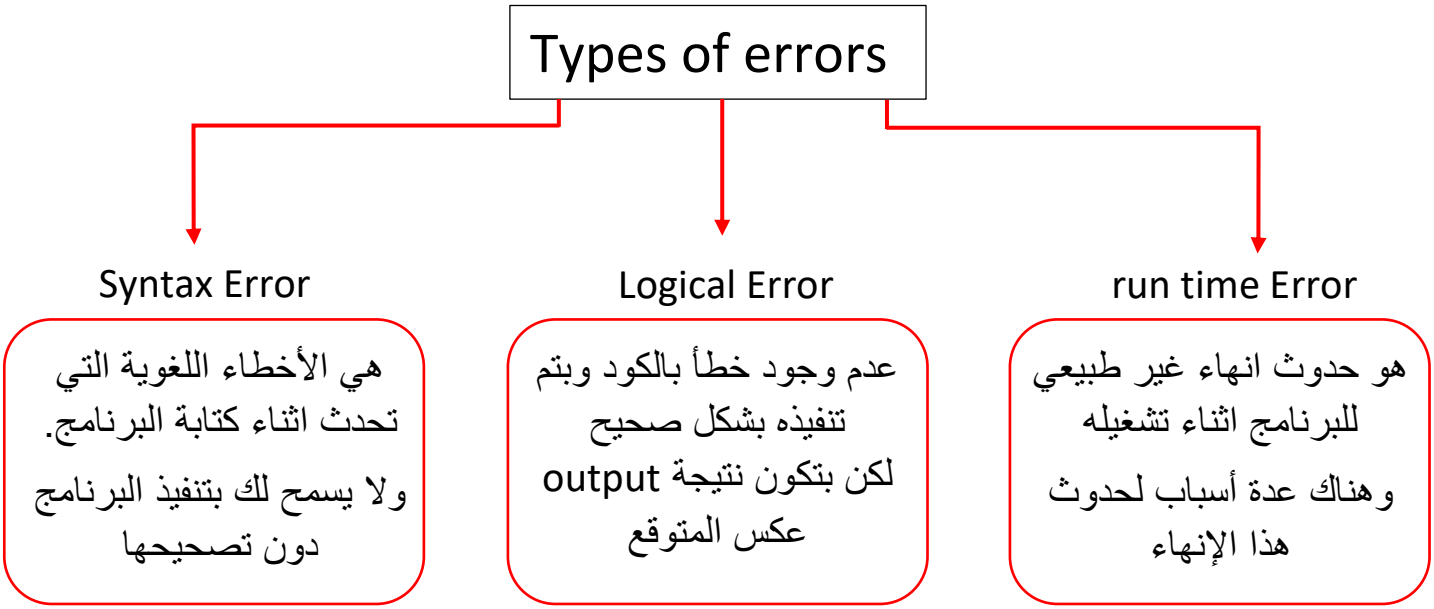
4) cout << 4 + 2 / 2 ; 5

cout << (4 + 2) / 2 ; 3

الأولويات الحسابية

Programming Errors

اثناء كتابتنا للكود ، قد نواجه بعض المشاكل او الاخطاء ، ومن هذه الاخطاء :



1) Syntax Errors:

```
#include [iostream]
using namespace std
int Main()
{
    cout << Hi << endl;
}
```

استخدام [ ] بدلا من < >

نسينا نحط ; اخر الجملة

حرف ال M لازم يكون m

ما حطينا ال Hi بين " "

- بنلاحظ ان لغة ال c++ تعتبر case santancev أي انها حساسة للأحرف

يعني استبدال رمز برمز اخر او حرف كبير بحرف صغير بسبب عندي مشكلة

main ≠ Main

<< ≠ >>

2) Logic Errors:

- مثال : اذا طلبنا طباعة ناتج جمع 5 و 3 مقسومة على 2:

```
cout << 5 + 3 / 2;    ->  4
```

بنلاحظ انه الإجابة الي طلعت هي 4 لكنها عكس المتوقع انها تطلع 6
وذلك لأنه نسينا نحط الأقواس للجمع، والحل الصحيح هو :

```
cout << (5 + 3) / 2;  -> 6
```

- مثال لو طلبنا طباعة Hello على سطر و c++ على سطر اخر :

```
cout << "Hello";
cout << "c++";
```

بنلاحظ انهم رح ينطبعوا على نفس السطر لأنه ما حطينا endl نهاية الجملة الأولى

```
cout << "Hello" << endl;
cout << "c++";
```

3) Runtime Errors:

يوجد عدة أنواع من الRuntime error سنتعرف عليها لاحقا، لكن منها:
قسمة العدد على صفر :

```
cout << 1 / 0 << endl;
```

Problems with answers :

. Show the output of the following code:

1)

```
#include <iostream>
using namespace std;

int main()
{
    //start
    cout << "Computer Skills For Engineers" << endl;

    return 0;
}
```

Output: Computer Skills For Engineers

2)

```
#include <iostream>
using namespace std;

int main()
{
    cout << "Hi, ";
    cout << "My name is ";
    cout << "C++!";

    return 0;
}
```

Output: Hi, My name is C++!

3)

```
#include <iostream>
using namespace std;

int main()
{
    cout << "(3 + 5) * 2 - 4 % 2 = ";
    cout << (3 + 5) * 2 - 4 % 2;
}
```

Output: (3 + 5) * 2 – 4 % 2 = 16

4)

```
#include <iostream>
using namespace std;

int main()
{
    cout << 3 + 2 / 2 << endl;;
    cout << (3 + 2) / 2 << endl;;
    cout << (3 + 2) / 2.0 << endl;;
}
```

Output: 4

2

2.5

.find the error in the following code:

```
#Include <iostream>
using name space std;
int main[]
{
   / this is comment
   cout >> "hello" >> endl;
   cout << "c++" __

   Return 0;
}
```

Correct:

```
#include <iostream>
using namespace std;
int main()
{
   // this is comment
   cout << "hello " << endl;
   cout << "c++";

   return 0;
}
```

. write the code that :

1) Show your name, university number, and major in separate lines

```
#include <iostream>
using namespace std;

int main()
{
     cout << "Ayman Qarout" << endl;
     cout << "0202790" << endl;
     cout << "Computer Engineering" << endl;
}
```

2) Write a program that displays the result of $\frac{1.2 \times 0.1+3.3 \times 0.3}{0.09+0.001}$:

```
#include <iostream>
using namespace std;

int main()
{
     cout << (1.2 * 0.1 + 3.3 * 0.3) / (0.09 + 0.001);
}
```

3) Write a program in C++ to print the following equations :

```
          1 + 1 = 2                    2 * 3 = 6
```

```
#include<iostream>
using namespace std;
int main()
{
     cout << "1 + 1 = " << 1 + 1 << endl;
     cout<< "2 * 3 = " << 2 * 3;
}
```