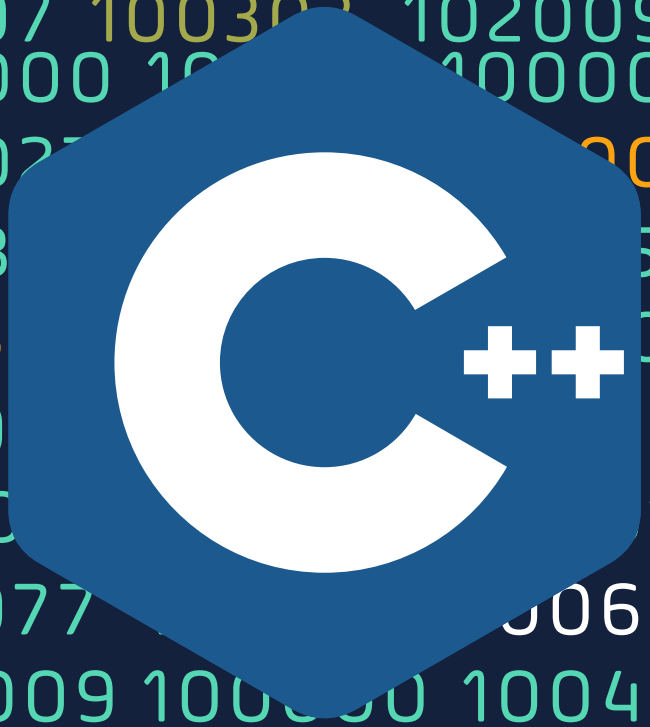


programming 1

By:

Suliman alkasasbeh

Introduction to programming languages

لغات البرمجة هي مجموعة من القواعد والتعليمات اللي بنستخدمها عشان نكتب برامج نتحكم فيها بالحاسوب. هي بمثابة لغة تواصل بين الإنسان والآلة، بحيث يكتب المبرمج الكود اللي بيوضح للحاسوب شو لازم يعمل.

أي لغة برمجة بتتكون من:

- القواعد (Syntax): مجموعة من القواعد يلي بتحدد الطريقة اللي من خلالها بتصيغ رموز هاللغة.

- دلالات الألفاظ (Semantics): هي القدرة انك تفهم معنى كل جملة مكتوبة بلغة معينة.

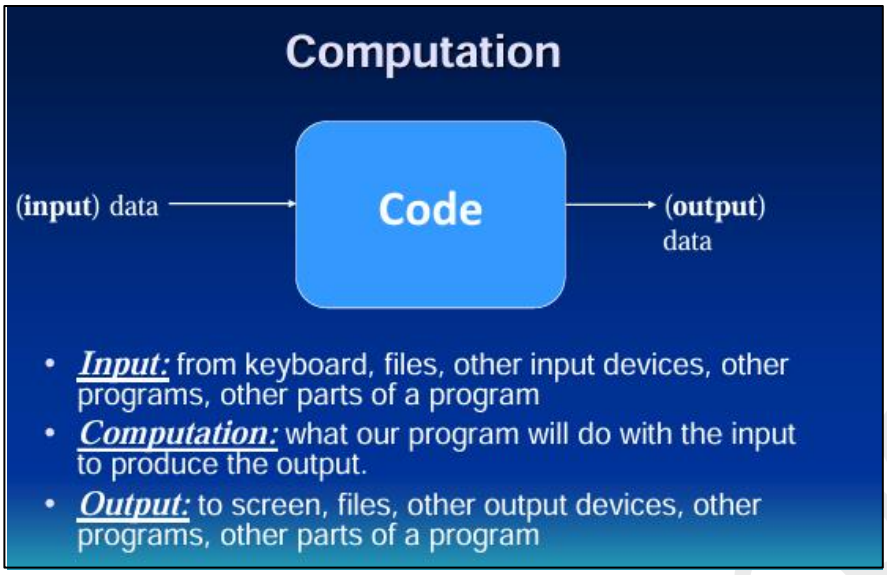
- مش كل اشي بنكتبه بالكود بكون منطقي حتى لو كان صحيح قواعديًا على سبيل المثال باللغة العربية ما بنفع احكي *"شربت السفينة التفاحة"* الجملة صحيحة من ناحية القواعد، لكن ليس لها معنى منطقي.

لماذا نحتاج إلى لغات البرمجة؟

الكمبيوتر لا يفهم إلا لغة بسيطة جدًا تتكون من 0 و 1 فقط (Binary Code).
لكن من المستحيل أن يكتب الإنسان برامج معقدة باستخدام 0 و 1 فقط، لذا تم تطوير لغات برمجة تسهّل التعامل مع الكمبيوتر.

- هناك لغات برمجة بسيطة تستخدم مجموعة محدودة من الأوامر مثل (Assembly).

- و هناك لغات برمجة متقدمة تسهّل الكتابة باستخدام رموز وكلمات قريبة من لغة البشر، مثل: C, C++, Java, Python, C#، وغيرها.



الصورة بتوضح فكرة العمليات الحسابية (Computation) في البرمجة، وكيف يتعامل البرنامج مع البيانات.

1. Input (المدخلات)
   - البيانات اللي بياخذها البرنامج من المستخدم (مثلاً من الكيبورد)، أو من ملفات، أو من أجهزة إدخال ثانية، أو حتى من برامج ثانية.
2. Computation (المعالجة)
   - العمليات اللي بيقوم فيها البرنامج على المدخلات، زي الحسابات، التعديلات، أو أي منطق برمجي عشان ينتج المخرجات.
3. Output (المخرجات)
   - النتيجة اللي بيطلعها البرنامج بعد المعالجة، وبتكون مثلاً على الشاشة، في ملف، أو تُرسل لجهاز أو برنامج آخر.

البرنامج بياخذ مدخلات ← يعالجها داخل الكود ← ويعطي مخرجات.

الكلمات المحجوزة (Reserved Words)

الكلمات المحجوزة، أو keywords، هي كلمات خاصة بلغة C++، وممنوع نستخدمها كأسماء متغيرات أو functions، لأنها إلها معنى محدد بالكومبايلر (رح نشرح شو ال functions والمتغيرات بشكل مفصل).

بعض الأمثلة على الكلمات المحجوزة:

- int ← بتستخدم لتعريف متغير عدد صحيح.
- float ← لتعريف عدد عشري.
- double ← زي float بس بدقة أعلى.
- char ← لتعريف حرف واحد.
- const ← لتعريف قيمة ثابتة ما بتتغير.
- return ← بترجع قيمة من الدالة.

Syntax

FutureReservedWord :: one of

abstract	enum	int	short
boolean	export	interface	static
byte	extends	long	super
char	final	native	synchronized
class	float	package	throws
const	goto	private	transient
debugger	implements	protected	volatile
double	import	public	

يعني ما بقدر أسمي متغير int، لأنه كلمة محجوزة بلغة C++

الفراغات والمسافات (Whitespaces)

كل برنامج مكتوب بـ C++ فيه whitespaces، وهي الفراغات اللي بتكون بين الكلمات والأوامر.

←أنواع whitespaces :

- المسافات الفارغة ← (blanks) لما تضغط "مسطرة".
- التاب ← (tabs) لما تضغط "Tab" لتنظيم الكود.
- السطر الجديد ← (newline) لما تضغط "Enter".

←ليش منستخدم whitespaces؟

- بفصل الرموز والكلمات المحجوزة عن بعضها، مثلاً: ←

```
intx = 5; // خطأ ❌
int x = 5; // صح ✅
```

- بخلّي الكود مرتب ومقروء أكثر، مثلاً: ←

```
int a=5;double b=2.5;char c='A'; // كود مش واضح ❌
int a = 5;
double b = 2.5;
char c = 'A';  // كود مرتب وواضح ✅
```

* يعني، إذا كتبت كود بدون فراغات، رح يكون مكركب وصعب تفهمه*

مثال على أهمية المسافات شوف الفرق بين هذول السطرين: ←

```
cout << "23 45";  // رح يطبع: 45 23
cout << "2345";   // رح يطبع: 2345
```

الفراغات جوّا النص ("هون يعني") بتأثر على النتيجة

Variables (المتغيرات)

أولاً : شو يعني متغير؟

- المتغير (Variable) هو مكان بتخزن فيه قيمة بتتغير أثناء تشغيل البرنامج، يعني ممكن نخزن فيه أرقام، حروف، أو نصوص.

- كل متغير لازم يكون اله اسم، ولازم نتبع شوية قواعد لما نسميه.

ثانياً: قواعد تسمية المتغيرات

- إيش الأشياء المسموح استخدامها بأسماء المتغيرات؟
  - حروف اللغة الإنجليزية (A-Z, a-z).
  - الأرقام(0-9).
  - الرمز (_) (Underscore).

- إيش الأشياء اللي مش مسموح؟
  - ما بصير يبدأ المتغير برقم ←

```
1  int 7number; // خطأ ❌
2  int number7; // صح ✅
```

  - ما بصير يكون في مسافة بالاسم ←

```
1  int first name; // خطأ ❌
2  int first_name; // صح ✅
3
```

اذا كان اسم المتغير من أكثر من مقطع استعمل ال (_) بدل المسافه مثل ما هو موضح بالصورة

  - ما بصير تسمي متغير بنفس اسم متغير ثاني بنفس المكان

```
1  int age; // int عرفنا متغير نوعه
2  int age; // (متغير بنفس الاسم) ❌ خطأ
```

- ما بصير تستخدم الكلمات المحجوزة كأسماء متغيرات ←

```
int int; // خطأ ❌
int return; // خطأ ❌
float char ; // خطأ ❌
char cout ; // خطأ ❌
```

*معلومة مهمة *

الـ C++ حساسة لحالة الأحرف (Case-Sensitive)، كيف يعني؟

← شو يعني"Case-Sensitive"؟

يعني الحروف الكبيرة والصغيرة تعامل كأحرف مختلفة .يعني إذا عندك متغير اسمه age ومتغير ثاني اسمه AGE ، الكمبيوتر رح يعتبرهم متغيرين مختلفين ، حتى لو الاسم متشابه تمامًا بس الفرق بالحروف الكبيرة والصغيرة.

مثال عملي:

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

int main() {
    int age = 25;   //  25 و قيمتة int من نوع age تعريف متغير اسمة
    int AGE = 30;   // 30 و قيمتة int من نوع AGE تعريف متغير اسمة

    cout << "قيمة المتغير age: " << age << endl;
    cout << "قيمة المتغير AGE: " << AGE << endl;

    return 0;
}
```

النتيجة عند تشغيل الكود:

```
age: 25 قيمة المتغير
AGE: 30 قيمة المتغير
```

لاحظ إنه "age" و "AGE" تعاملوا كمتغيرين مختلفين، حتى لو نفس الاسم بس الفرق بالحروف الكبيرة والصغيرة

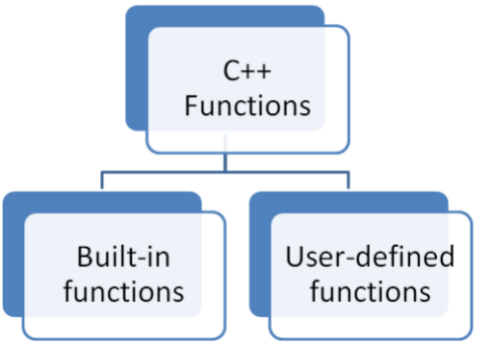
ثالثاً: أمثلة على أسماء متغيرات مسموحة (Legal) وممنوعة(Illegal)

أمثلة صحيحة	أمثلة خاطئة	سبب الخطأ
Int age;	int 1stNumber;	لا يجوز وضع رقم في بداية اسم المتغير
double height;	int return;	لا يجوز استعمال كلمات محجوزه كأسم متغير وفي المثال هذا تم استعمال كلمة return
char first_letter;	int first name;	لا يجوز وضع مسافات في اسم المتغير
float _salary;	double double;	لا يجوز استعمال كلمات محجوزه كأسم متغير وفي المثال هذا تم استعمال كلمة double

الدوال (Functions)

◈شو يعني دالة (Function) ؟

- الدالة عبارة عن مجموعة أوامر برمجية بتنكتب مرة وحدة وبنستدعيها بأي مكان بالكود بدل ما نكرر نفس الأوامر أكثر من مرة .هاد بساعد بتقليل حجم الكود، تسهيل قراءته، وإعادة استخدامه.



في C++عندنا نوعين أساسيين من الدوال:

- الدوال الجاهزة ← (Predefined Functions) موجودة أساسًا داخل مكتبات C++، بنستخدمها مباشرة بدون ما نكتبها.
- الدوال اللي بكتبها المبرمج ← (User-Defined Functions) هاي الدوال انت كمبرمج بتكتبها بنفسك حسب الحاجة و عندنا نوعين منهم , نوع برجع قيمة return function و نوع ما رجع قيمه void function.

خلينا نشرح نوع نوع:

➔الدوال الجاهزة (Predefined Functions)

◆هي الدوال اللي C++موفرتها تلقائيًا داخل مكتباتها، فإنت ما بتحتاج تكتبها، بس بتستدعيها لما تحتاجها.
◆حتى تقدر تستخدمها، لازم تضيف المكتبة المناسبة باستخدام #include مثل مكتبة ال cmath اللي رح نشرحها الان.

- شرح دوال الرياضيات الجاهزة في مكتبة ال(cmath)

C++ فيها مجموعة دوال رياضية جاهزة داخل مكتبة <cmath>، وهي بتسهل علينا عمليات رياضية مثل الجذر التربيعي، القوة، التقريب، إلخ.
◆لازم نضيف المكتبة #include <cmath>حتى نقدر نستخدم هاي الدوال.

➔ sqrt (x)

هذا الفنكشن بياخد رقم (double x) وبحسب الجذر التربيعي له و برجعلي اياه.

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

int main() {

    double x = 2.25 ;
    cout << sqrt(x);

}
```

```
1.5
```

- طبعا مش شرط يكون الرقم double ممكن int / float لكن يفضل يكون double تجنبا للأخطاء و لو ما دخلت double ال c++ بتعمل casting للرقم و بتحوله double -

الجذر االتربيعي ل 2.25 هو 1.5

➔ pow(x, y)

هذا الفنكشن بياخد رقمين (x, y) وبحسب x^y يعني (x مرفوع للقوة y)

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

int main() {

    double base = 2 , power = 5 ;
    cout << pow(base , power);

}
```

```
2^5 = 32
```

-القوه هي ضرب الرقم بنفسه على عدد الأس يعني بالمثال عندنا

2^5 يعني (2*2*2*2*2 =32) -

➔ floor(x)

التقريب لأسفل (القيمة الصحيحة الأصغر) بتاخد رقم (double x) وبتقربه لأقرب عدد صحيح أصغر أو يساويه.

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

int main() {

    double num1 = 48.79 , num2 = 74.359 , num3 = -3.45;
    cout <<" 48.79 -> " <<floor(num1) << endl;
    cout <<" 74.359 -> " <<floor(num2) << endl;
    cout <<" -3.45 -> " <<floor(num3) << endl;

}
```

```
48.79 -> 48
74.359 -> 74
-3.45 -> -4
```

لاحظ عندنا تم التقريب لاقرب عدد صحيح أصغر

طيب ليش ال num3 تم تقريبا لل -4؟

قيمتها -3.45 و بما انها عدد سالب تم تقريبا لل -4

لانه ال -3.45>-4.

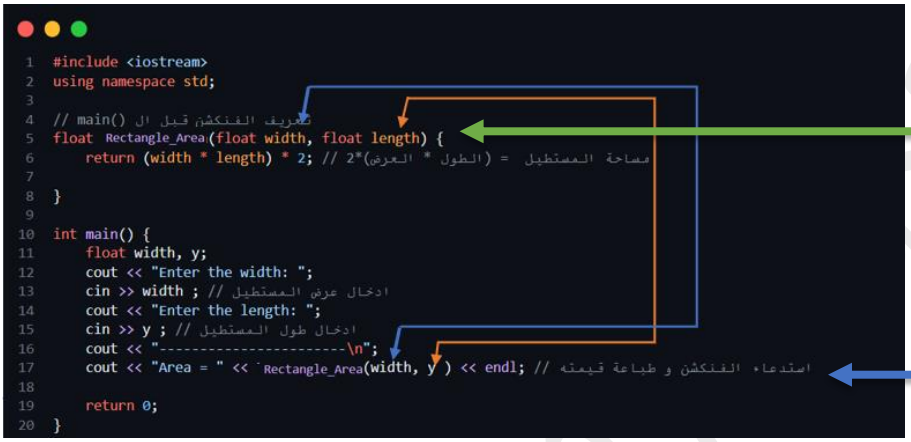
في عندنا نوعين من ال parameters (القيم اللي بستقبلها الفنكشن) عند تعريفه

➔(Formal Parameters)

هي عبارة عن متغيرات بتنكتب داخل الدالة، ومهمتها إنها تستقبل القيم اللي رح تنبعث إلها لما نستدعي الدالة.

➔(actual Parameters)

هي القيم اللي بنبعثها عند استدعاء الدالة، وبتروح تنحط مكان ال formal parameters مثل ما هو موجود بالمثال اللي بالصفحة السابقة.



بالمثال هون عندنا ال Formal parameters (width و length)

و ال actual Parameters (width و y)

Formal Parameter → name → greet(string name).

Actual Parameter → person → in greet(person).

Arrays

المصفوفات (Arrays) في C++

شو يعني مصفوفة؟

المصفوفة هي مجموعة من القيم من نفس النوع، بتنحفظ ورا بعض بالذاكرة، وبتقدر توصل لأي قيمة فيها باستخدام ال(Index) و هو بمثابة فهرس للعناصر داخل ال array.

ليش نستخدم المصفوفات؟

لما يكون عندك مجموعة قيم متشابهة في نوع ال Data Type، بدل ما تعرّف متغير لكل وحدة، بتقدر تخزّنهم كلهم بمصفوفة وتتعامل معهم بطريقة أسهل.

مثال:

بدل ما تعرّف 5 متغيرات نوعهم int

زي هيك:

```
1   int num1 = 10;
2   int num2 = 20;
3   int num3 = 30;
4   int num4 = 40;
5   int num5 = 50;
```

بتقدر تحطهم كلهم بمصفوفة وحدة كالتالي:

10	20	30	40	50

هون، القيم بتكون مخزنة بمناطق متجاورة بالذاكرة، وكل وحدة إلها رقم فهرس (Index).

المصفوفة الي عملناها فيها 5 قيم كل قيمة الها رقم اسمه index ببدأ من الصفر وبنتهي ((بعدد عناصر المصفوفة) – 1) كالتالي:

0	1	2	3	4
10	20	30	40	50