**Class**

ده بيكون جواه : ( Data type , Function , Constructor , deconstructor)

وليه 3 أنواع (Public , Private , Protected)

Public : is access modifier

ده هيخلي اللى في ال Class اقدر اوصلهم عادي

**Constructor**

ده Special method بتشتغل تلقائيا لما بعمل Create للـ الObject

وده بنستخدمه عشان نعمل initialization للــ object لما اعمله create

* Constructor is a special type of member function that is used to initialize the data members for an object of a class automatically, when an object of the same class is created.
* A constructor is a method that is automatically called when an object is created
* Constructors typically initialize object attributes and perform other object initialization tasks
* It is used to initialize the data members of new objects generally
* The constructor in C++ has the same name as the class or structure
* It constructs the values i.e. provides data for the object which is why it is known as constructors.

**Constructors – Initialization list**

يفضل لما تيجي تعمل Initialize للــ Constructor تعمله باستخدام حاجة اسمها List

وده احسن بدل ما نفتح الBody بتاع الــ Constructor ونكتبه جواه ال Initializations

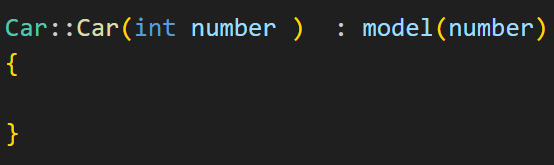
ودي بتفرق في سرعة التنفيذ بتاعت سرعة الـ Initialization

A picture containing text, font, screenshot, number

Description automatically generatedA picture containing font, text, screenshot, graphics

Description automatically generated

وممكن اخلي الUser يخلي ال Initializations زي ما هو عايز بالطريقة دي



The constructor section (.ctors) is a section in the executable file that contains pointers to the constructors of static objects or global variables with constructor functions. These constructor functions are automatically called before the program's main function is executed. The purpose of these constructors is to initialize the static objects or global variables.

**Deconstructor**

ده Special method بتشتغل تلقائيا لما ال scope بتاع ال Object بيخلص

مثال:

لو عايز أقول welcome هتكون في ال Instructor ولو عايز أقول goodbye هتكون في الــ Deconstructor

* Destructor is an instance member function which is invoked automatically whenever an object is going to be destroyed. Meaning, a destructor is the last function that is going to be called before an object is destroyed.
* It is not possible to define more than one destructor.
* The destructor is only one way to destroy the object create by constructor. Hence destructor can-not be overloaded.
* Destructor neither requires any argument nor returns any value.
* Destructor release memory space occupied by the objects created by constructor.
* In destructor, objects are destroyed in the reverse of an object creation

**Method and Constructor Overloading**

ده لما بيكون عندي اكتر من method بنفس الاسم ... طب ازاي ؟!

ده بيحصل طالما ال method اللى عندي بيكون ال signature بتاعهم مختلف

**Method** is a member Function.

A picture containing text, screenshot, font, number

Description automatically generatedSignature --> ( Number of arguments , Types of arguments , order of arguments)

الكلام ده برضو ينفع نطبقه على ال Constructor نفسه

 Car();  // Default constructor declaration

 Car(int number);  // Parameterized constructor declaration

يبقي كده احنا بنستخدم حوار ال overloading عشان يكون عندنا default values لو احنا مدخلناش حاجة

**Copy Constructor**

ده هنستخدمه عشان نعمل initialize للــ object جديد بنفس مواصفات ال object القديم ، كأني بعمله copy بالظبط

ودي built in ومش مضطر اني اكتب حاجة زيادة

     Car mycar;         // Default Constructor

     Car car2(55);      // Parameterized constructor

     Car car3(car2);    // Copy Constructor

     Car car4 = car2;   // Copy Constructor

هنا ال compiler هيفهم انك عايز تنسخ القيم اللى في car2 وتحطها في car4 or car3

احنا نقدر نبعت ال object للFunction كأنه Argument .. زي ما كنا بنعمل مع ال struct في ال C

Passing objects To Methods

Car Fun(Car c1);

بس لو عايز تغير في ال object اللى هتبعته يبقي تكتب العنوان بتاعه في ال Function

Car othermodel(Car &c1);

**Static Class Members**

Class members

* هي ال Attribute or the method اللى بنعملها definition داخل ال Class

**طب يعني اى Static دي ؟!**

* ده معناه ان ال Variable ده عايش معايا طول البرنامج على عكس باقي الحاجات في ال Class
* وده بيكون خاص بالــ Class نفسها مش ال instance or object الخاص بيها
* ده مش بيكون خاص بــ Object واحد ولكن بيكون visible to all objects

**بنستخدمه امتي :**

* في حالة ال Shared data لما يكون عندي داتا وعايز كل ال Objects of a class تقدر تشوفها
* اني احتاج اعمل counter مثلاً

**Initialization :**

طالما عملته static يبقي مينفعش تحط فيه اى قيمة في نفس السطر زي كده static int counter = 0; في ال .h

لازم تروح ال File.cpp بتاع الClass اللى فيه ال Definition وتكتبه بنفسك من تاني int Car::count = 0

( :: ) scope operator

Static attribute : we can’t initialize it inside the class

**بنستدعيها ازاي:**

من ال Class مباشرة ، دون الحاجة اني اعمل Object جديد

**ليه بحتاج اعمل static method ؟**

* عشان استخدمها كــ Utility method

على سبيل المثال Calculator class

class Calculator

{

private:

    /\* data \*/

public:

    int   add(int num1,int num2);

    int   add(int num1,int num2,int num3);

    float add(float num1,float num2);

    Calculator(void);

    ~Calculator();

};

انا مش كل ما احب اعمل add هضطر اني اعمل New object واستدعي add ، انا مش عايز كده ... ودي هيكبر الكود size بتاعي وهيكون كود كتيير بدون داعي ... طب هعمل اى؟

Calculator cal1 ;

cal1.add(5,3);

هعمل static method

class Calculator

{

private:

    /\* data \*/

public:

    static int   add(int num1,int num2);

    static int   add(int num1,int num2,int num3);

    static float add(float num1,float num2);

    Calculator(void);

    ~Calculator();

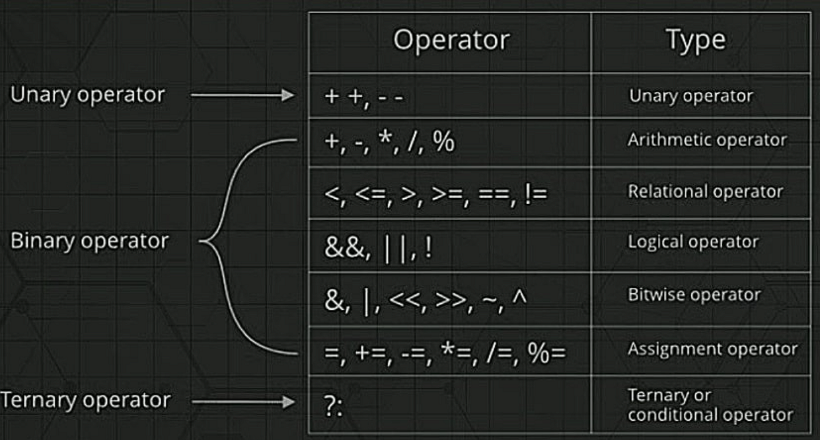
};

وبكده هستدعيها مباشرة من اسم ال Class

Calculator::add(2,5);

**Operator Overloading**

أنواع ال Operators زي ما واضع عندي في الصورة



* Operator overloading is an idea of giving special meaning to an existing operator in C++ without changing its original meaning.
* In C++, we can make operators work for user-defined classes.
* This means C++ has the ability to provide the operators with a special meaning for a data type, this ability is known as operator overloading.
* **الهدف من الـــ Operator overloading ده :**
* اني اخلي الــ Class بتاعي والــ User define data بتاعتي ليها نفس الإمكانيات والعمليات   
  بتاعت الــ default data types

عندنا في ال C++ الــ Operator ممكن يعمل كذا حاجة على حسب بيتعامل مع اى ، يعني مثلاً:

A picture containing text, font, screenshot, clock

Description automatically generated

* هنا الـــ (+) بيرجعلي ال Sum بتاع الرقمين دول

A picture containing text, screenshot, font, number

Description automatically generated

* هنا مجابش مجموعهم لكن   
  جاب الــ Concatenation بتاعت ال Strings دي

* **وبكده هكون عارف ان :**
* الــ C++ عارفه لما تلاقي (+) بين 2 int هتعمل اى وهترجع الSum بتاعهم
* A picture containing text, font, screenshot, symbol

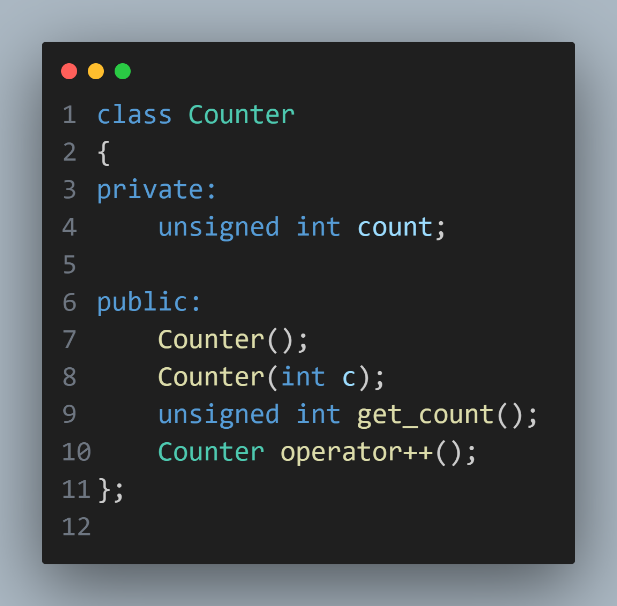
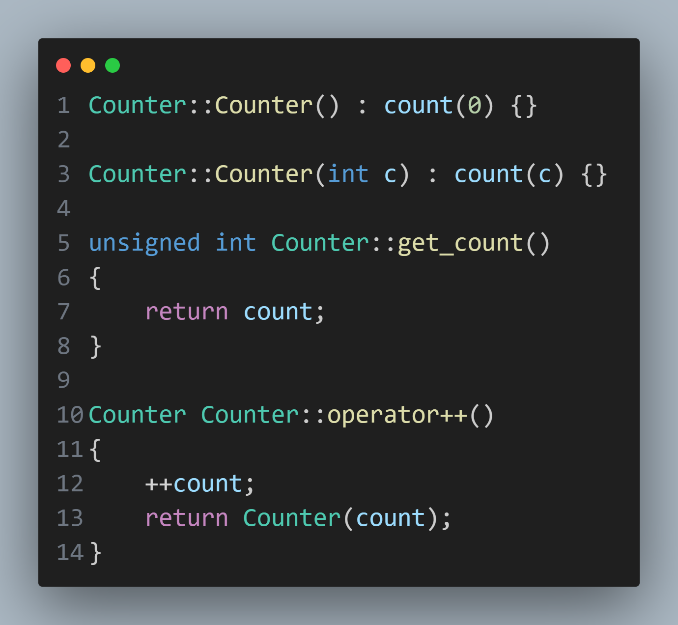
  Description automatically generatedو كمان عارفه لما تلاقي (+) بين 2 strings هتعمل اى وهترجع الــ Concatenation اللى هو “hello world”
* لكن هي متعرفش لما تلاقي (+) بين اتنين object هتعمل اى
* فانا عايز لما تلاقي (+) بين اتنين object اقولها تعمل اى بالظبط وهنعمل كده جوا ال Class وهنستخدم حاجة اسماها Operator overloading

**بنعمله ازاي ؟**

* بنستخدم keyword اسمها operator
* بنعمل Function or method جديدة في ال class اسمها Operator Function

File.cpp

File.hpp



**الـUnary Operator ليها عند الاستخدام شكلين (Prefix , Postfix)**

A picture containing text, screenshot, businesscard, font

Description automatically generated

كلمة int هنا ملهاش اى استخدم هي بس بتستخدم للتفرقة بين النوعين

**بعد كده الـ Binary Operator :**

دي بتحتاج 2 operand عشان تشتغل عليهم

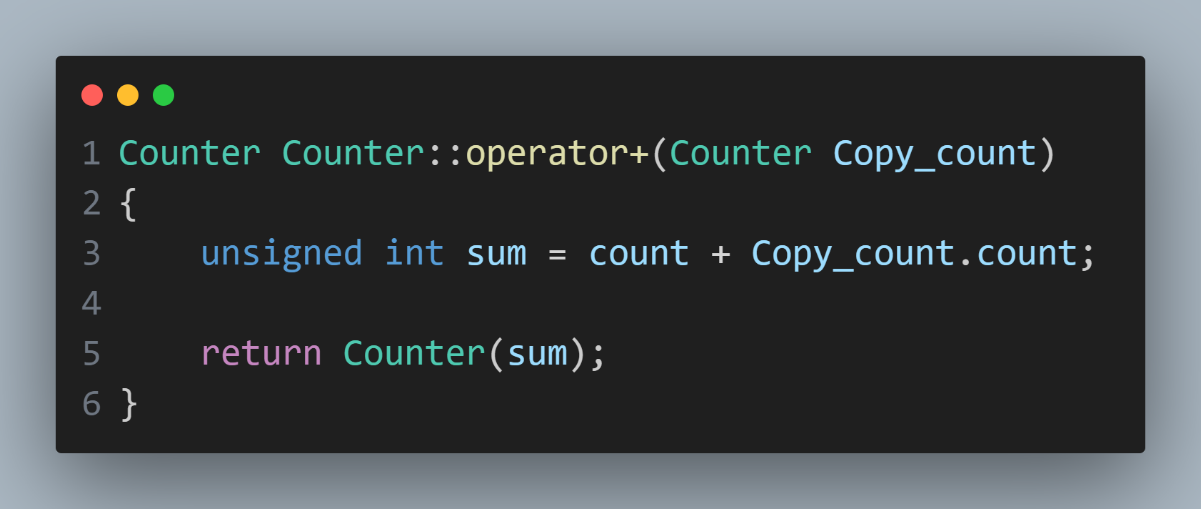
وبنكون عارفين ان الـOperator بيتبع الـ Object الأول

A picture containing text, screenshot, font, number

Description automatically generated

يعني مثلاً لو عايز اجمع اتنين Object مع بعض زي كده c3 = c1 + c2

A picture containing text, screenshot, font, businesscard

Description automatically generated

File.hpp

File.cpp

انا هنا رجعت Object جديد وبعتله قيمة الـ Sum كــ Initialization ليه