

القسم: الحاسبات
تقنية المعلومات
المستوى: ثاني
المقرر: OOP - عملي



الجمهورية اليمنية
وزارة التعليم العالي والبحث العلمي
جامعة الجزيرة
كلية العلوم والهندسة

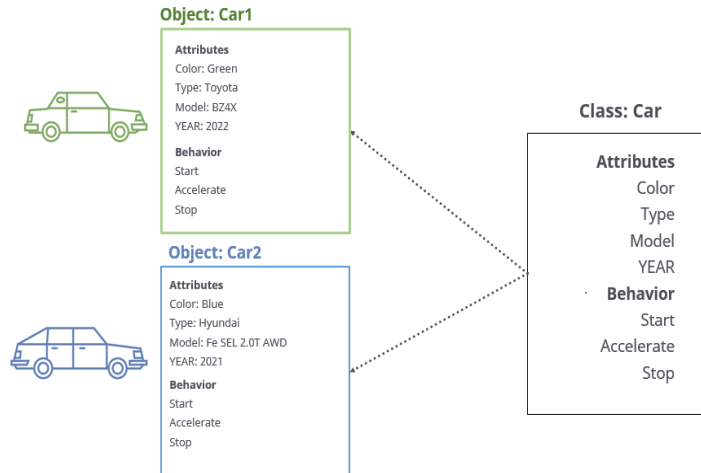
Introduction

Lect 1

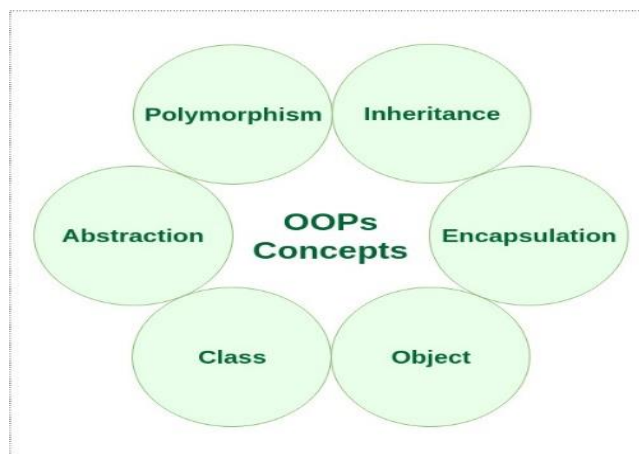
Eng. Abeer Mohammed

Object Oriented Programming

Object-Oriented Programming (OOP) is a programming paradigm that is based on the concept of classes and objects, where it is used to build programs by dividing them into manageable and simple parts. This paradigm focuses on organizing the source code around data or objects rather than functions and logic.



Object-Oriented Programming is characterized by several fundamental concepts



Inheritance: allows the creation of new classes based on existing ones, which facilitates code reuse.

Encapsulation: where data and their related functions are bundled into a single object, which enhances data security.

Polymorphism: objects can behave in different ways depending on the context of their use, which increases the flexibility of programming.

Abstraction: used to facilitate writing code for programmers. It allows you to execute what you want without needing to know all the details of how it was implemented.

الفرق بين البرمجة الكائنية والبرمجة الإجرائية

البرمجة الكائنية التوجه (OOP)	البرمجة الإجرائية
تركز البرمجة كائنية التوجه على البيانات التي يجب استخدامها لحل المشكلة المطلوبة	تركز لغة البرمجة الإجرائية على تنفيذ مجموعة من الخطوات المتسلسلة لحل المشكلة المطلوبة
(Classes & Objects) هي العناصر الأساسية للبرنامج	الدوال هي العناصر الأساسية للبرنامج
تضمن حماية البيانات بشكل كبير عن طريق إخفاء البيانات الحساسة عن المستخدمين	لا تضمن حماية البيانات وأمانها
مناسبة للتطبيقات الكبيرة والمعقدة	مناسبة للتطبيقات البسيطة عامة الأغراض وأنظمة التشغيل والبرامج المضمنة
تمكنك من نمذجة مشكلات العالم الحقيقي بفعالية ومرونة	لا تمكنك من نمذجة مشكلات العالم الحقيقي بفعالية

★ The Classes

Class is a group of similar objects. It is a template from which objects are created. It can have fields, methods, constructors etc. the classes are used to create custom types. The class defines the kinds of information and methods included in a custom type.

- A Class is a user-defined data type that has data members and member functions.
- Data members are the data variables and member functions are the functions used to manipulate these variables together these data members and member functions define the properties and behavior of the objects in a Class

```

Class Student

Data:
1- Student_name ,
2- University_Id
3- Birth_Date
4- Address
5- GPA
6- Study_Level

Operations ( )
1- Modify GPA()
2- Change Study level ()
3- Get Student Name ()
4- Get Student Address ()
    
```

```

Student 1

Data:
1- Student_name
2- University_Id
3- Birth_Date
4- Address
5- GPA
6- Study_Level

Operations ( )
1- Modify GPA()
2- Change Study level ()
3- Get Student Name ()
4- Get Student Address ()
    
```

```

Student 3

Data:
1- Student_name
2- University_Id
3- Birth_Date
4- Address
5- GPA
6- Study_Level

Operations ( )
1- Modify GPA()
2- Change Study level ()
3- Get Student Name ()
4- Get Student Address ()
    
```

```

Student 2

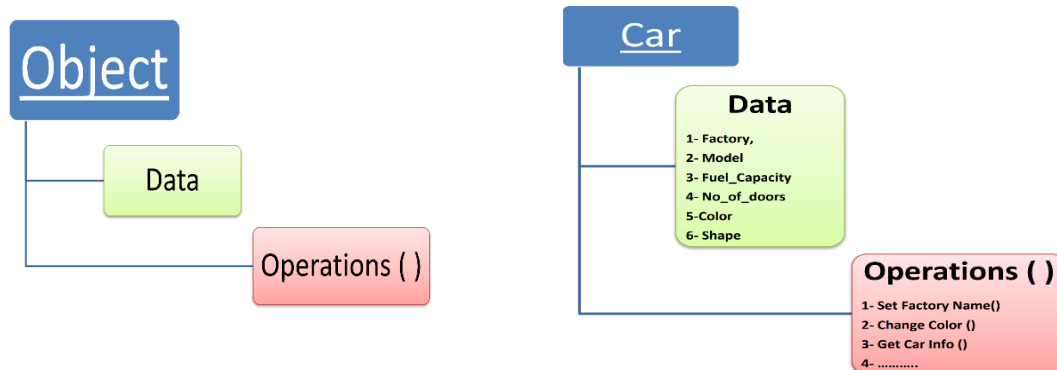
Data:
1- Student_name
2- University_Id
3- Birth_Date
4- Address
5- GPA
6- Study_Level

Operations ( )
1- Modify GPA()
2- Change Study level ()
3- Get Student Name ()
4- Get Student Address ()
    
```

★ Object

Object is an entity that has state and behavior. Here, state means data and behavior means functionality.

An Object is an identifiable entity with some characteristics and behavior. An Object is an instance of a Class. When a class is defined, no memory is allocated but when it is instantiated (i.e. an object is created) memory is allocated.



▪ Create a Class

To create a class, use the **class** keyword:

Example

Create a class called "MyClass":

```
class MyClass {    // The class
public:           // Access specifier
    int Number;    // Attribute (int variable)
    string Name;   // Attribute (string variable)
}
```

Example explained

- The **class** keyword is used to create a class called **MyClass**.
- The **public** keyword is an **access specifier**, which specifies that members (attributes and methods) of the class are accessible from outside the class.
- Inside the class, there is an integer variable **Number** and a string variable **Name**. When variables are declared within a class, they are called **attributes**.

```

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

class MyClass {
public:
    int Number;
    string Name;
};

```

▪ Create an Object

```

int main() {
    // Create an object of MyClass
    MyClass myObj;

    // Access attributes and set values
    myObj.Number = 15;
    myObj.Name = "Mohammed";

    // Print attribute values
    cout << myObj.Number << endl;
    cout << myObj.Name << endl;

    system("pause");
    return 0;
}

```

• Access Modifiers

An access modifier indicates how a field or method can be accessed.

Public

When the public access modifier is applied to a class member, the member can be accessed by code inside the class or outside.

Private

When the private access modifier is applied to a class member, the member cannot be accessed by code outside the class. The member can be accessed only by methods that are member of the same class.

In other word, members of a class can be marked with access modifiers, including public and private. A public member can be accessed by other classes. A private member can only be accessed by code in the same class.

Example-1: The program to store students' data

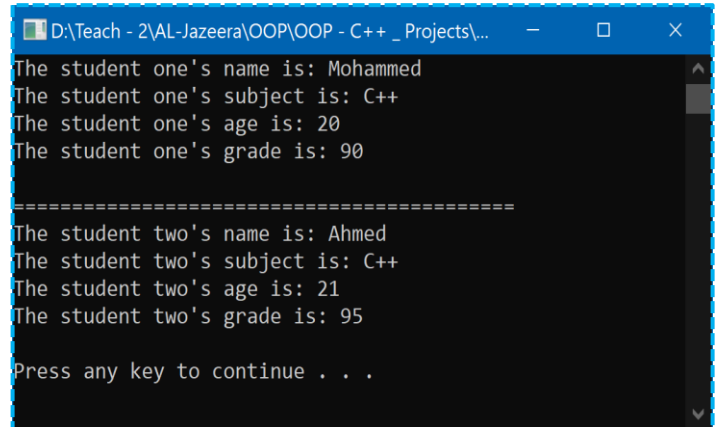
```
#include <iostream>
#include <string>
using namespace std;

// Define the Student class
class Students
{
public:
    // Public member variables
    string Name;    // Student's name
    string Subject; // Subject the student is enrolled in
    int Age;        // Age of the student
    int Grade;      // Grade of the student
};

int main()
{
    // Create two objects of the Student class
    Students number1; // First student object
    Students number2; // Second student object

    // Assign values to the member variables of the first student
    number1.Name = "Mohammed";
    number1.Subject = "C++";
    number1.Age = 20;
    number1.Grade = 90;

    // Display the information of the first student
    cout << "The student one's name is: " << number1.Name << endl;
    cout << "The student one's subject is: " << number1.Subject << endl;
    cout << "The student one's age is: " << number1.Age << endl;
    cout << "The student one's grade is: " << number1.Grade << endl;
    cout << "\n===== \n";
    // Assign values to the member variables of the second student
```



The screenshot shows a Windows command prompt window titled "D:\Teach - 2\AL-Jazeera\OOP\OOP - C++ _ Projects\...". The output of the program is displayed as follows:

```
The student one's name is: Mohammed
The student one's subject is: C++
The student one's age is: 20
The student one's grade is: 90

=====
The student two's name is: Ahmed
The student two's subject is: C++
The student two's age is: 21
The student two's grade is: 95

Press any key to continue . . .
```

```

number2.Name = "Ahmed";
number2.Subject = "C++";
number2.Age = 21;
number2.Grade = 95;

// Display the information of the second student
cout << "The student two's name is: " << number2.Name << endl;
cout << "The student two's subject is: " << number2.Subject << endl;
cout << "The student two's age is: " << number2.Age << endl;
cout << "The student two's grade is: " << number2.Grade << endl;

system("pause");
return 0;
}

```

✳ Member Function

A member function of a class is a function that has its definition or its prototype within the class definition like any other variable. It operates on any object of the class of which it is a member, and has access to all the members of a class for that object.

Example-2:

```

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

class Books
{
public:
    string name;
    int price;
    int pages;
    string Author;
    // Method to print book details
    void print()
    {
        cout << "The name of the book is: " << name << endl;
    }
}

```

```

        cout << "The price of the book is: " << price << endl;
        cout << "The number of the book pages are: " << pages << endl;
        cout << "Name of the book author: " << Author << endl;
    }
};

int main()
{
    Books book1;
    Books book3;

    book1.name = "C#";
    book1.price = 40;
    book1.pages = 750;
    book1.Author = "Mazen";
    book1.print(); // Print details of the first book
    cout << "\n===== \n";

    Books book2 = { "Python", 80, 800, "Mohammed" };
    book2.print(); // Print details of the second book
    cout << "\n===== \n";

    book3.print();

    cout << endl;
    system("pause");
    return 0;
}

```

```

D:\Teach\AL-Qalam\OOP_C#\C# - Projects\class-lect2-2\class-lect2-2\bin\Debu...
The name of the book is: C#
The price of the book is: 40
The number of the book pages are: 750
Name of the book author: Mazen

=====

The name of the book is: Python
The price of the book is: 80
The number of the book pages are: 800
Name of the book author: Mohammed

=====

The name of the book is:
The price of the book is: 0
The number of the book pages are: 0
Name of the book author:

```