Lecture-4

Class

Class is a group of objects that share common properties and relationships. In C++, a class is a new data type that contains member variables and member functions that operates on the variables. A class is defined with the keyword class. It allows the data to be hidden, if necessary, from external use. When we defining a class, we are creating a new abstract data type that can be treated like any other built in data type. Generally, a class specification has two parts:

a) Class declaration

b) Class function definition

The class declaration describes the type and scope of its members. The class function definition describes how the class functions are implemented.

Syntax:

```
class class-name
{
Private:
        Varable declarations;
        Function declaration;
Public:
        Variable declarations;
        Function declaration;
}
```

The members that have been declared as private can be accessed only from within the class. On the other hand, public members can be accessed from outside the class also. The data hiding is the key feature of oops. The use of keywords private is optional by default, the members of a class are private.

In C++, OOP involves defining and working with classes and objects. Classes are used to create user-defined data types, and objects are instances of those classes. Variables in C++ classes can be declared in different ways, and I'll provide with some examples of variable

declarations in a simple class.

```
#include<iostream>
using namespace std;
class Car {
public:
    // Member variables (attributes or properties)
    string make;
    string model;
    int year;
    double price;
    // Member functions (methods)
    void startEngine() {
        cout << "Engine started for " << make << " " << model <<"
"<<year<<" "<<price<<endl;
    }
};
main()
{
       Car myCar; // Create an object using the default constructor
       myCar.make = "Toyota";
       myCar.model = "Camry";
       myCar.year = 2022;
       myCar.price = 25000.00;
       myCar.startEngine(); // Call a member function
}
```

Example

```
#include<iostream>
using namespace std;
class Rectangle {
public:
    double calculateArea() {
        double area = length * width;  // Local variable
        return area;
    }
    double length;  // Member variable
    double width;   // Member variable
};
main()
{
       Rectangle rec;
       rec.length=5;
       rec.width=5;
       cout<<rec.calculateArea();
}
```

The variables declared inside the class are known as data members and the functions are known as members mid the functions. Only the member functions can have access to the private data members and private functions. However, the public members can be accessed from the outside the class. The binding of data and functions together into a single class type variable is referred to as encapsulation.

Syntax:

```
Class item
{
        int member;
        float cost;
Public:
        void getldata(int a, float b);
        void putdata (void);
}
```

The class item contains two data members and two function members, the data members are private by default while both the functions are public by declaration. The function getdata() can be used to assign values to the member variables member and cost, and putdata() for displaying their values . These functions provide the only access to the data members from outside the class.

Creating Objects

Once a class has been declared we can create variables of that type by using the class name. Example: item x; creates a variables x of type item. In C++, the class variables are known as objects. Therefore, x is called an object of type item. Item x, y, z also possible.

```
class item
{
----------
----------
----------
}x ,y ,z;
```

would create the objects x, y, z of type item.

```
#include <iostream>
#include <string>
using namespace std;
class Car {
public:
    // Data members (attributes)
    string make;
    string model;
    int year;
    // Member functions (methods)
    void startEngine() {
        cout << "The " << year << " " << make << " " << model << "'s
engine is running." << endl;
    }
    void stopEngine() {
        cout << "The " << year << " " << make << " " << model << "'s
engine is stopped." << endl;
    }
};
main() {
    // Create two objects of the Car class
    Car car1;
    Car car2;
    // Set attributes for car1
    car1.make = "Toyota";
    car1.model = "Camry";
    car1.year = 2022;
    // Set attributes for car2
    car2.make = "Honda";
    car2.model = "Civic";
    car2.year = 2021;
    // Call methods for both objects
    car1.startEngine();
    car2.startEngine();
    car1.stopEngine();
    car2.stopEngine();
}
```

Accessing Class Member

The private data of a class can be accessed only through the member functions of that class. The main () cannot contains statements that the access number and cost directly.

Syntax:

object name.function-name(actual arguments);

Example:- x. getdata(100,75.5);

It assigns value 100 to number, and 75.5 to cost of the object x by implementing the getdata() function . similarly, the statement

x. putdata ( ); //would display the values of data members.

x. number = 100 is illegal. Although x is an object of the type item to which number belongs, the number can be accessed only through a member function and not by the object directly.

Example:

```
Class xyz
{
    int x;
    int y;
public:
    int z;
};
Main()
{
    xyz  p;
    p.x =4; // error x is private
    p.z = 5; // Ok z is public
}
```

```
#include <iostream>
class Student {
public:
    // Data members (attributes)
    string name;
    int age;
    double gpa;


    // Member functions (methods)
    void displayInfo() {
        cout << "Name: " << name << endl;
        cout << "Age: " << age << " years" << endl;
        cout << "GPA: " << gpa << endl;
    }


    void study() {
        cout << name << " is studying." << endl;
    }
};
```

To access the member functions of a class, you also use the object's name followed by the **dot** operator and the name of the method. Example:

```
main() {
    // Create an object of the student class
    Student student1;
    // Access and set the object's data members
    student1.name = "Alice";
    student1.age = 20;
    student1.gpa = 3.8;
    // Access and call the object's methods
    student1.displayInfo();
    student1.study();
}
```

Example:

```
#include <iostream>
#include <string>
class Book {
public:
    // Data members (attributes)
    std::string title;
    std::string author;
    int publicationYear;
    // Member function to display book details
    void displayInfo() {
        cout << "Title: " << title << endl;
        cout << "Author: " << author << endl;
        cout << "Publication Year: " << publicationYear << endl;
    }
};
main() {
    // Create two Book objects
    Book book1;
    Book book2;
    // Initialize data members directly
    book1.title = "To Kill a Mockingbird";
    book1.author = "Harper Lee";
    book1.publicationYear = 1960;
    book2.title = "1984";
    book2.author = "George Orwell";
    book2.publicationYear = 1949;
    // Access and call methods for both books
    book1.displayInfo();
    book2.displayInfo();
}
```