Class and Objects

Class	A user-defined data type that defines attributes (data members) and behaviors (member functions). It acts as a blueprint for objects
Object	An instance of a class with specific values and behaviors

Pillars of OOP

Encapsulation	Encapsulation is wrapping up the data and methods together within a single entity. In C++, classes are used for encapsulation.
Abstra-ction	Showing only the necessary details and hiding the internal details is known as abstraction.
Polymorphism	Providing different functionalities to the functions or operators of the same name is known as Polymorphism.

Access Specifiers

• public: Accessible from outside the class.
• private: Cannot be accessed directly from outside.
• protected: Like private, but accessible in derived classes.

```
class Demo {
private:
int secretNumber;
public:
int publicNumber;
};
```

Class Example

```
class Car {
public:
        string brand;
       int year;
};
int main() {
       Car myCar;
        myC ar.b rand = " Toy ota ";
        myC ar.year = 2022;
}
```

Inheritance

Deriving the properties of a class (Parent class) to another class (Child class) is known as Inheritance. It is used for code reusabilty.

Building a Class

Attribute (Data Members)	variables which represent the attributes of the class
Functionality (Member Functions)	functions which represent the behaviors of the class

Constructor

a special member function that is automatically called when an object of a class is created

Object Instantiation

• Creating objects from a class.
• Accessing attributes/methods using dot syntax.

```
Circle c1(5.0);
cout << c1.radius;
```

Demonstration

```
Compute circumference and area of a circle.
class Circle {
private:
        double radius;
public:
        Cir cle (double r) { radius = r; }
        double getCir cum fer ence() { return 2
3.1416 radius; }
        double getArea() { return 3.1416 * radius
radius; }
};
int main() {
        Circle c1(5);
       cout << " Cir cum fer ence: " << c1.get -
Cir cum fer ence();
       cout << " Area: " << c1.get Area();
}
```

Class Rectangle

```
class Rectangle {
public:
       int length, width;
       int area() {
                return length * width;
       }
};
```