✅ Part 1: 50 Simple MCQs on OOP in C++

✅ Part 2: 10 Scenario-Based MCQs on OOP in C++

✅ Part 3: 35 Find output and find error type.

Part 1 –

1. What is constructor overloading?
a) Using constructors from another class
b) Creating multiple constructors with different parameters
c) Copying constructors
d) Deleting constructors
✅ **Answer: b)**

---

2. Which keyword is used to create a derived class in C++?
a) extends
b) derives
c) : (colon)
d) inherits
✅ **Answer: c)**

---

3. What is function overloading?
a) Using the same function in multiple files
b) Writing the same function name with different parameter types
c) Overwriting a function from base class
d) Creating two constructors
✅ **Answer: b)**

---

4. Which concept is used to hide internal details and show only essential features?
a) Inheritance
b) Polymorphism
c) Abstraction
d) Encapsulation
✅ **Answer: c)**

---

5. A constructor has:
a) Return type
b) Same name as class and no return type
c) No name
d) Always returns int
✅ **Answer: b)**

---

6. What does polymorphism allow in C++?
a) Memory sharing
b) Multiple objects from one class
c) One function behaving differently in different contexts
d) Hiding data
✅ **Answer: c)**

---

7. Which inheritance type has one base class and multiple derived classes?
a) Single
b) Multilevel
c) Multiple
d) Hierarchical
✅ **Answer: d)**

---

8. What is the use of a default constructor?
a) Initialize values passed from main
b) Create an object without arguments
c) Inherit class
d) Delete an object
✅ **Answer: b)**

---

9. Which access specifier allows members to be accessed by derived classes only?
a) public
b) private
c) protected
d) static
✅ **Answer: c)**

---

10. What is hybrid inheritance?
a) Using a constructor in inheritance
b) Combining more than one type of inheritance
c) Inheriting only private data
d) Mixing polymorphism with encapsulation
✅ **Answer: b)**

---

11. Which constructor is called automatically when an object is created?
a) Parameterized constructor
b) Copy constructor
c) Default constructor
d) Static constructor
✅ **Answer: c)**

---

12. What is the purpose of a copy constructor?
a) To copy an object into another
b) To call private members
c) To delete objects
d) To define default values
✅ **Answer: a)**

---

13. Which of the following is NOT a type of polymorphism in C++?
a) Compile-time
b) Runtime
c) Static
d) Object-time
✅ **Answer: d)**

---

14. Which function is automatically called when an object goes out of scope?
a) Constructor
b) Destructor
c) Operator
d) Init
✅ **Answer: b)**

---

15. What is the syntax to call the base class constructor from a derived class?
a) Base()
b) Derived::Base()
c) Initialization list syntax
d) Constructor chaining
✅ **Answer: c)**

---

16. What does method overriding require?
a) Function name only
b) Same name and different parameters
c) Same signature in base and derived classes
d) Static function
✅ **Answer: c)**

---

17. Which of the following demonstrates hierarchical inheritance?
a) A -> B -> C
b) A, B -> C
c) A -> B, A -> C
d) A -> B
✅ **Answer: c)**

---

18. Operator overloading in C++ is done using:
a) Overload keyword
b) New operator
c) friend function
d) `operator` keyword
✅ **Answer: d)**

---

19. Which of the following supports multiple inheritance in C++?
a) Java
b) Python
c) C++
d) None
✅ **Answer: c)**

---

20. A constructor that accepts arguments is called:
a) Default constructor
b) Copy constructor
c) Parametrized constructor
d) Static constructor
✅ **Answer: c)**

---

21. What is the main goal of encapsulation?
a) To inherit classes
b) To increase function speed
c) To protect data
d) To perform polymorphism
✅ **Answer: c)**

---

22. Which member function is automatically called when an object is destroyed?
a) Destructor
b) finalize()
c) Free()
d) Cleanup
✅ **Answer: a)**

---

23. Which of the following can be overloaded in C++?
a) class
b) operator
c) object
d) access specifier
✅ **Answer: b)**

---

24. Which of the following is a correct way to declare a constructor in a class named `Box`?
a) void Box()
b) Box()
c) int Box()
d) constructor Box()
✅ **Answer: b)**

---

25. Which feature allows a function or operator to behave differently for different data types?
a) Encapsulation
b) Inheritance
c) Polymorphism
d) Constructor
✅ **Answer: c)**

---

26. What is required for constructor overloading to happen?
a) Same number of parameters
b) Different number or type of parameters
c) Multiple classes
d) Public variables
✅ **Answer: b)**

---

27. What is the return type of a constructor?
a) void
b) int
c) No return type
d) static
✅ **Answer: c)**

---

28. Which inheritance type does this represent: `class B : public A {}`?
a) Multiple
b) Hierarchical
c) Single
d) Multilevel
✅ **Answer: c)**

---

29. What does `virtual` keyword help with?
a) Abstraction
b) Function overloading
c) Runtime polymorphism
d) Compile-time polymorphism
✅ **Answer: c)**

---

30. Which of the following best defines abstraction?
a) Hiding implementation details
b) Inheriting multiple classes
c) Overriding base class methods
d) Defining multiple constructors
✅ **Answer: a)**

---

31. Which type of inheritance involves a class derived from two or more base classes?
a) Multilevel
b) Multiple
c) Hierarchical
d) Single
✅ **Answer: b)**

---

32. Which of the following is automatically called when an object is created?
a) Destructor
b) Constructor
c) Main()
d) Init()
✅ **Answer: b)**

---

33. What is the purpose of `this` pointer in C++?
a) Refers to parent class
b) Refers to derived class
c) Refers to the current object
d) Refers to static member
✅ **Answer: c)**

---

34. Which concept allows a class to have more than one method with the same name but different parameters?
a) Method hiding
b) Method overriding
c) Method overloading
d) Inheritance
✅ **Answer: c)**

---

35. How many times is a constructor called in a program?
a) Once only
b) Twice
c) Every time an object is created
d) Never
✅ **Answer: c)**

---

36. What is the output type of overloaded operators?
a) Always int
b) Same as operands
c) Depends on operator
d) Can be customized
✅ **Answer: d)**

---

37. Which of the following can be inherited?
a) private data
b) static members
c) constructors
d) public and protected members
✅ **Answer: d)**

---

38. A class that cannot be instantiated is known as:
a) Abstract class
b) Inherited class
c) Derived class
d) Hidden class
✅ **Answer: a)**

---

39. Which of these allows the use of one interface for multiple implementations?
a) Encapsulation
b) Inheritance
c) Abstraction
d) Polymorphism
✅ **Answer: d)**

---

40. What does a destructor look like in C++?
a) ~ClassName()
b) destroy()
c) destructor ClassName()
d) !ClassName()
✅ **Answer: a)**

---

41. Which constructor is called when a new object is created as a copy of an existing object?
a) Default constructor
b) Parameterized constructor
c) Copy constructor
d) Inline constructor
✅ **Answer: c)**

---

42. Which feature of OOP ensures that only essential details are visible to the user?
a) Inheritance
b) Abstraction
c) Overloading
d) Polymorphism
✅ **Answer: b)**

---

43. How can you define an abstract class in C++?
a) By inheriting from another class
b) By making all data members private
c) By including at least one pure virtual function
d) By declaring it with the `abstract` keyword
✅ **Answer: c)**

---

44. What is the default access specifier for class members in C++?
a) private
b) protected
c) public
d) global
✅ **Answer: a)**

---

45. What happens if you don't define a constructor in your class?
a) Compilation error
b) Constructor is inherited
c) Compiler provides a default constructor
d) Object cannot be created
✅ **Answer: c)**

---

46. Which inheritance type is shown by: `class C : public A, public B {}`?
a) Single
b) Multilevel
c) Multiple
d) Hierarchical
✅ **Answer: c)**

---

47. What is the purpose of the `virtual` keyword?
a) To create new variables
b) To make a function overrideable
c) To protect data
d) To call constructors
✅ **Answer: b)**

---

48. Which of these features is used to reuse code in C++?
a) Encapsulation
b) Inheritance
c) Abstraction
d) Virtualization
✅ **Answer: b)**

---

49. Which of these cannot be overloaded?
a) +
b) =
c) ::
d) ==
✅ **Answer: c)**

---

50. What is the order of constructor calling in multilevel inheritance?
a) Derived to base
b) Random
c) Base to derived
d) Only base is called
✅ **Answer: c)**

Part 2 – Scenario-Based MCQs

---

1. You are designing a game. Each character has health and stamina. You want to create a class and ensure other parts of the program cannot access or modify these values directly. What concept should you use?
a) Inheritance
b) Encapsulation
c) Overloading
d) Polymorphism
✅ **Answer: b)**

---

2. You have a class `Shape` with a function `draw()`. Circle, Square, and Triangle are derived from `Shape`, and each has its own version of `draw()`. Which concept is this?
a) Overloading
b) Abstraction
c) Runtime Polymorphism
d) Constructor Overloading
✅ **Answer: c)**

---

3. You created a base class `Person` and derived class `Student`. When creating a `Student` object, the `Person` constructor is also called first. What is this an example of?
a) Function Overloading
b) Destructor Chaining
c) Constructor Inheritance
d) Constructor Chaining
✅ **Answer: d)**

---

4. In a banking system, you want to define a class `Account` that cannot be instantiated but can be used to derive `SavingsAccount` and `CurrentAccount`. What should you do?

a) Use private constructor
b) Use static class
c) Make `Account` an abstract class
d) Use virtual functions
✅ **Answer: c)**

---

5. You want to create a function `add()` in a calculator program that can accept both two and three numbers. What technique will you use?
a) Operator Overloading
b) Constructor Chaining
c) Function Overloading
d) Virtual Function
✅ **Answer: c)**

---

6. A class `Printer` has a method `print()` and another method `print(int copies)`. What is this an example of?
a) Function overriding
b) Function hiding
c) Function overloading
d) Inheritance
✅ **Answer: c)**

---

7. You want to show only necessary details of a class `Employee` to users and hide salary computation logic. What is this called?
a) Encapsulation
b) Abstraction
c) Inheritance
d) Polymorphism
✅ **Answer: b)**

---

8. You are writing a program for a college. `Teacher` and `Student` are derived from `Person`, and `TeachingAssistant` inherits both. What type of inheritance is this?
a) Multilevel
b) Multiple
c) Hierarchical
d) Hybrid
✅ **Answer: d)**

---

9. You want a single function `area()` in a class to calculate the area of a square, circle, and rectangle. What will you use?
a) Operator Overloading
b) Function Overloading
c) Inheritance
d) Abstract Classes
✅ **Answer: b)**

---

10. You are creating a class `Vehicle` with `Car` and `Truck` derived from it. What type of inheritance is this?
a) Multilevel
b) Hierarchical
c) Multiple
d) Hybrid
✅ **Answer: b)**

---

15 Part 3 : Find the Output / Find the Error

---

1. What is the output of the following code?

```
class Test {

public:

    Test() {

        cout << "Constructor called";

    }

};


int main() {

    Test t;

    return 0;
```

```
}
```

a) Constructor not called
b) Constructor called
c) Error
d) Runtime error
✅ **Answer: b)**

---

2. Find the output:

```
class A {
public:
    A() {
        cout << "A ";
    }
};

class B : public A {
public:
    B() {
        cout << "B";
    }
};

int main() {
    B obj;
    return 0;
```

```
}
```

a) A B
b) B A
c) Compilation error
d) B
✅ **Answer: a)**

---

3. Identify the error:

```
class Test {
    int x;
    void show() {
        cout << x;
    }
};

int main() {
    Test t;
    t.show();
    return 0;
}
```

a) No error
b) Runtime error
c) 'show' is private
d) 'x' is undefined
✅ **Answer: c)**

---

4. Output of the following code:

```
class A {
public:
    void display() {
        cout << "Base";
    }
};

class B : public A {
public:
    void display() {
        cout << "Derived";
    }
};

int main() {
    B obj;
    obj.display();
    return 0;
}
```

a) Base
b) Derived
c) Compilation error
d) Both
✅ **Answer: b)**

---

5. Find the output:

```
class Demo {

public:

    ~Demo() {

        cout << "Destructor called";

    }

};


int main() {

    Demo d;

    return 0;

}
```

a) Nothing
b) Destructor called
c) Error
d) Compilation failed
✅ **Answer: b)**

---

6. What is the output?

```
class Base {

public:

    Base() {

        cout << "Base ";

    }
```

```
};

class Derived : public Base {
public:
    Derived() {
        cout << "Derived";
    }
};

int main() {
    Derived d;
    return 0;
}
```

a) Derived Base
b) Base Derived
c) Error
d) Nothing
✅ **Answer: b)**

---

7. Identify the error:

```
class Test {
public:
    void print();
};
```

```
int main() {
    Test t;
    t.print();
    return 0;
}
```

a) No error
b) Undefined reference to `print()`
c) Runtime error
d) Syntax error
✅ **Answer: b)**

---

8. Output of this program:

```
class Sample {
public:
    Sample() {
        cout << "Sample Constructor";
    }
};

int main() {
    Sample *s = new Sample();
    return 0;
}
```

a) Nothing
b) Sample Constructor

c) Error
d) Constructor not called
✅ **Answer: b)**

---

9. What is the output?

```
class A {
public:
    void show() {
        cout << "A";
    }
};

class B : public A {
public:
    void show() {
        cout << "B";
    }
};

int main() {
    A *obj = new B;
    obj->show();
    return 0;
}
```

a) B
b) A
c) AB
d) Runtime error
✅ **Answer: b)**

---

10. Output of the following:

```
class MyClass {
    int a;
public:
    MyClass() {
        a = 5;
    }
    void show() {
        cout << a;
    }
};

int main() {
    MyClass obj;
    obj.show();
    return 0;
}
```

a) 5
b) Error
c) Garbage value

d) Undefined
✅ **Answer: a)**

---

11. Error in the code below?

```
class Test {
    int a = 10;
public:
    void show();
};

void Test::show() {
    cout << a;
}

int main() {
    Test t;
    t.show();
}
```

a) Error in show()
b) Output is 10
c) a is private
d) Invalid main
✅ **Answer: b)**

---

12. What will happen?

```
class A {
```

```
    int x = 5;
};

int main() {
    A obj;
    cout << obj.x;
    return 0;
}
```

a) 5
b) Compilation error (private access)
c) Garbage value
d) Runtime error
✅ **Answer: b)**

---

13. Output of the code:

```
class A {
public:
    void show() {
        cout << "A";
    }
};

class B : public A {};

int main() {
```

```
    B b;
    b.show();
    return 0;
}
```

a) A
b) B
c) Error
d) Nothing
✅ **Answer: a)**

---

14. What will this code output?

```
class A {
public:
    A() {
        cout << "Hello ";
    }
    ~A() {
        cout << "World";
    }
};

int main() {
    A obj;
    return 0;
}
```

a) Hello
b) World
c) Hello World
d) Nothing
✅ **Answer: c)**

---

15. Find the output:

```
class A {

public:

    A() {

        cout << "Base ";

    }

    ~A() {

        cout << "End ";

    }

};


int main() {

    A a1, a2;

    return 0;

}
```

a) Base End
b) Base Base End End
c) End End Base Base
d) Error
✅ **Answer: b)**

Find Output / Find Error MCQs

---

1. What is the output of the following code?

```
class Test {
public:
    Test() { cout << "Constructor "; }
};
int main() {
    Test t;
    return 0;
}
```

a) Constructor
b) Error
c) Nothing
✅ **Answer: a) Constructor**

---

2. Find the output:

```
class A {
public:
    A() { cout << "A "; }
};
class B : public A {
public:
    B() { cout << "B "; }
```

```
};

int main() {

    B obj;

}
```

a) B A
b) A B
c) Compilation error
✅ **Answer: b) A B**

---

3. Identify the error:

```
class Test {

    int x;

    void show() { cout << x; }

};

int main() {

    Test t;

    t.show();

}
```

a) 0
b) Compilation error
c) 'show' is private
✅ **Answer: c) 'show' is private**

---

4. Output of the following code:

```
class A {
```

```
public:
    void display() { cout << "Base"; }
};
class B : public A {
public:
    void display() { cout << "Derived"; }
};
int main() {
    B obj;
    obj.display();
}
```

a) Base
b) Derived
c) Error
✅ **Answer: b) Derived**

---

5. What is the output?

```
class Demo {
public:
    ~Demo() { cout << "Destructor"; }
};
int main() {
    Demo d;
    return 0;
}
```

a) Nothing
b) Destructor
c) Compilation error
✅ **Answer: b) Destructor**

---

6. Output of the program:

```
class Base {

public:

    Base() { cout << "Base "; }

};

class Derived : public Base {

public:

    Derived() { cout << "Derived"; }

};

int main() {

    Derived d;

}
```

a) Derived Base
b) Base Derived
c) Error
✅ **Answer: b) Base Derived**

---

7. Find the error:

```
class Test {

public:
```

```
    void print();
};
int main() {
    Test t;
    t.print();
}
```

a) No error
b) Undefined reference to `print()`
c) Compilation error
✅ **Answer: b) Undefined reference to `print()`**

---

8. What is the output?

```
class Sample {
public:
    Sample() { cout << "Constructor"; }
};
int main() {
    Sample *s = new Sample();
}
```

a) No output
b) Constructor
c) Error
✅ **Answer: b) Constructor**

---

9. What is the output?

```
class A {
public:
    void show() { cout << "A"; }
};
class B : public A {
public:
    void show() { cout << "B"; }
};
int main() {
    A* obj = new B;
    obj->show();
}
```

a) A
b) B
c) Error
✅ **Answer: a) A**

---

10. Find the output:

```
class MyClass {
    int a;
public:
    MyClass() { a = 10; }
    void display() { cout << a; }
};
int main() {
```

```
    MyClass obj;

    obj.display();

}
```

a) 0
 b) 10
 c) Error
 ✅ **Answer: b) 10**

---

11. What will be the output?

```
class A {

public:

    A() { cout << "A "; }

    ~A() { cout << "Z "; }

};

int main() {

    A a;

}
```

a) A
 b) Z
 c) A Z
 ✅ **Answer: c) A Z**

---

12. Find the error:

```
class A {

    int x = 5;
```

```
};

int main() {

    A obj;

    cout << obj.x;

}
```

a) 5
 b) Compilation error (x is private)
 c) Runtime error
 ✅ **Answer: b) Compilation error**

---

13. Output?

```
class A {

public:

    void show() { cout << "A"; }

};

class B : public A {};

int main() {

    B b;

    b.show();

}
```

a) A
 b) B
 c) Error
 ✅ **Answer: a) A**

---

14. What does this code output?

```
class A {
public:
    A() { cout << "Start "; }
    ~A() { cout << "End"; }
};
int main() {
    A obj;
}
```

a) Start
b) End
c) Start End
✅ **Answer: c) Start End**

---

15. Output?

```
class A {
public:
    void print() { cout << "A"; }
};
class B : public A {
public:
    void print() { cout << "B"; }
};
int main() {
    A *ptr = new B;
```

```
    ptr->print();
}
```

a) A
b) B
c) Error
✅ **Answer: a) A** (No virtual function)

---

16. What is the output?

```
int main() {
    int x = 5;
    int& y = x;
    y = 10;
    cout << x;
}
```

a) 5
b) 10
c) Error
✅ **Answer: b) 10**

---

17. Find the output:

```
class A {
public:
    A(int a) { cout << a; }
};
int main() {
```

```
    A obj(10);
}
```

a) Error
b) 10
c) Nothing
✅ **Answer: b) 10**

---

18. Output of the program:

```
class A {
public:
    A() { cout << "A "; }
};
class B : virtual public A {
public:
    B() { cout << "B "; }
};
int main() {
    B b;
}
```

a) A B
b) B A
c) Error
✅ **Answer: a) A B**

---

19. Output?

```
class Test {

public:

    Test(int a) { cout << a; }

};

int main() {

    Test t(15);

}
```

a) Error
 b) 15
 c) 0
 ✅ **Answer: b) 15**

---

20. Output of constructor overloading:

```
class A {

public:

    A() { cout << "Default "; }

    A(int x) { cout << "Param "; }

};

int main() {

    A a1, a2(10);

}
```

a) Default Param
 b) Param Default
 c) Error
 ✅ **Answer: a) Default Param**

---

21. Find the error:

```
class Test {
    static int x;
};
int Test::x = 0;
int main() {
    cout << Test::x;
}
```

a) 0
b) Compilation error
c) Runtime error
✅ **Answer: a) 0**

---

22. Output?

```
class Base {
public:
    Base() { cout << "Base "; }
};
class Derived : public Base {};
int main() {
    Derived d;
}
```

a) Nothing
b) Base
c) Error
✅ **Answer: b) Base**

---

23. What will be the output?

```
class A {

public:

    void display() { cout << "A"; }

};

class B : public A {

public:

    void display() { cout << "B"; }

};

int main() {

    A *a = new B();

    a->display();

}
```

a) A
b) B
c) Error
✅ **Answer: a) A**

---

24. Identify the output:

```
class MyClass {

public:
```

```
    MyClass() { cout << "Object Created"; }
};
int main() {
    MyClass obj;
}
```

a) Nothing
b) Object Created
c) Error
✅ **Answer: b) Object Created**

---

25. Output:

```
class A {
public:
    A() { cout << "Hello "; }
    ~A() { cout << "World"; }
};
int main() {
    A a;
}
```

a) Hello
b) World
c) Hello World
✅ **Answer: c) Hello World**

PBL Questions:

---

1. Class and Object Basics (Easy)

Problem:
Create a class `Student` with attributes `name` and `roll_no`. Take input from the user and display the student's details.

Input:

John

101

Output:

Name: John

Roll No: 101

---

2. Constructor Initialization (Easy)

Problem:
Write a class `Book` with a parameterized constructor to initialize `title` and `price`. Print the book details.

---

3. Single Inheritance (Easy)

Problem:
Create two classes: `Person` (with `name`) and `Employee` (inherits from Person, with `salary`). Display both attributes.

---

4. Multilevel Inheritance (Medium)

Problem:
Create a base class `Vehicle`, a derived class `Car`, and a further derived class `ElectricCar`. Add a method to print the complete info.

---

5. Method Overloading (Easy)

Problem:
Create a class `Calculator` with overloaded methods `add()` for adding two integers, three integers, and two floats.

---

6. Method Overriding & Polymorphism (Medium)

Problem:
Create a base class `Shape` with method `draw()`. Override this method in derived classes `Circle` and `Rectangle`. Use a pointer to call `draw()` based on object type.

---

7. Operator Overloading (Medium)

Problem:
Overload the `+` operator to add two complex numbers using a `Complex` class.

---

8. Abstract Class & Interface-like Behavior (Medium)

Problem:
Create an abstract class `Animal` with a pure virtual function `makeSound()`. Derive `Dog` and `Cat` from it and override the function.

---

9. Constructor Overloading (Easy)

Problem:
Write a class `Box` with overloaded constructors: one with no parameters, one with width and height, and one with all dimensions.

---

10. Exception Handling (Medium)

Problem:
Take two integers from the user and perform division. Handle the case where the denominator is zero using try-catch.

11. Encapsulation Using Getters and Setters (Easy)

Problem:
Create a class `BankAccount` with private data members `accountNumber` and `balance`. Provide getter and setter methods to access and update them.

---

12. Protected Access Modifier (Easy)

Problem:
Create a base class `Parent` with a protected member `familyName`. Inherit a class `Child` and display the family name using it.

---

13. Hierarchical Inheritance (Medium)

Problem:
Create a base class `Employee`, and two derived classes `Manager` and `Clerk`. Display all relevant information using hierarchical inheritance.

---

14. Hybrid Inheritance (Medium)

Problem:
Create a class structure to demonstrate hybrid inheritance using `Student`, `Sports`, and `Marks` classes, all contributing to a `Result` class.

---

15. Composition ("Has-a" relationship) (Easy)

Problem:
Create a class `Engine` and a class `Car` that has an `Engine` object. Show composition by calling a function of Engine through Car.

---

16. Static Members (Easy)

Problem:
Write a class `Counter` that counts the number of objects created using a static member.

---

17. Friend Function (Medium)

Problem:
Create a class `Box` and use a friend function `compareVolume()` to compare volumes of two `Box` objects.

---

18. Inline Functions (Easy)

Problem:
Define an inline member function in class `Math` that returns the square of a number.

---

19. Default Constructor and Destructor (Easy)

Problem:
Create a class `Demo` that displays messages from its constructor and destructor.

---

20. Virtual Function and Runtime Polymorphism (Medium)

Problem:
Create a base class `Account` with a virtual function `calculateInterest()`, and override it in `SavingsAccount` and `CurrentAccount`.

---

21. Array of Objects (Easy)

Problem:
Create a class `Item` with properties `id` and `price`. Create an array of 5 objects and display their details.

---

22. Multiple Inheritance (Medium)

Problem:
Create classes `Teacher` and `Researcher`. Derive a class `Professor` that inherits from both. Show multiple inheritance in action.

---

23. Operator Overloading: Unary Operator (Medium)

Problem:
Overload the unary `-` operator to negate the value of a class object representing a number.

---

24. Copy Constructor (Medium)

Problem:
Create a class `Book` and implement a copy constructor that copies the details of another book object.

---

25. Constructor with Default Arguments (Easy)

Problem:
Create a class `Rectangle` with a constructor that has default values for width and height.

---

26. Virtual Destructor (Medium)

Problem:
Create a base class `Shape` and derived class `Triangle`. Use a virtual destructor to ensure proper cleanup.

---

27. Overloading Comparison Operator (Medium)

Problem:
Overload the `==` operator in a `Time` class to compare two time objects.

---

28. File Handling with Class (Medium)

Problem:
Create a class `Student` and write object data to a file and read it back.

---

29. Object as Function Argument (Easy)

Problem:
Create a class `Circle` and write a function that takes a `Circle` object as an argument to compute area.

---

30. Array of Pointers to Objects (Medium)

Problem:
Create a class `Animal` and an array of pointers to dynamically allocate and manage 3 animals

ALL THE BEST