OOP Concepts: Exception Handling, Inheritance, Polymorphism

Problem Set (Intermediate to Hard)

Exception Handling (15 questions)

1. Explain the differences in exception handling mechanisms between C++ and Java.

2. What is the significance of `throw` keyword in exception handling? Show an example in C++.

3. Why must Java only throw objects derived from `Throwable`? What happens if you try to throw a primitive type?

4. What happens when an exception is thrown but not caught in C++?

5. How can you create a user-defined exception class in Java? Give an example.

6. Write a C++ program that throws and catches exceptions of types `int`, `string`, and a custom enum.

7. How does the `finally` block in Java help in resource management? Provide a code example.

8. What is exception rethrowing? Show how to rethrow exceptions in both C++ and Java.

9. Find the error and fix it in this C++ exception code snippet:

```cpp
try {
    throw 5;
} catch (char* e) {
    cout << "Caught string exception" << endl;
}
```

10. How do you catch all exceptions regardless of type in C++?

11. Write Java code using `throws` keyword in method signature and handle exception in caller.

12. Explain stack unwinding in C++ exception handling.

13. Can a `catch` block catch exceptions of base type and derived exceptions? Explain with example.

14. What is the difference between checked and unchecked exceptions in Java?

15. How does the C++ `noexcept` specifier affect exception handling.

Inheritance (20 questions)

16. Explain the difference between `private`, `protected`, and `public` inheritance in C++.

17. How does access specifier in inheritance affect member accessibility in Java?

18. What is the order of constructor and destructor calls in multiple inheritance?

19. Write C++ code to demonstrate the diamond problem and solve it using virtual inheritance.

20. Why does Java not allow multiple inheritance of classes? What feature compensates for this?

21. How do you use `super` in Java constructors to call base class constructor?

22. What happens if a derived class does not call the base class constructor explicitly in C++?

23. Write a C++ program showing constructor chaining with arguments passed to base class.

24. Find and fix errors in this C++ code involving multiple inheritance and ambiguous member access.

25. Explain upcasting and downcasting with examples in C++ and Java.

26. How does Java achieve runtime polymorphism without pointers?

27. What is a virtual destructor? Why is it important in base classes?

28. Demonstrate method overriding in Java and how access specifiers behave in this context.

29. Write a C++ example where base class pointer refers to derived class object and calls virtual functions.

30. Explain the use of interfaces in Java and their role in replacing multiple inheritance.

31. How can a nested interface be used in Java? Give an example.

32. Discuss how protected members behave differently in Java and C++ inheritance.

33. What are abstract classes? How are they used in both C++ and Java?

34. Write a C++ class with a pure virtual function and derive two classes implementing it.

35. What happens if a derived class does not implement all abstract methods in Java?

36. Describe how final variables are initialized in Java constructors.

Polymorphism and Operator Overloading (15 questions)

37. Explain early binding vs late binding. Which C++ keyword enables late binding?

38. What happens if a virtual function is called via base class pointer pointing to derived class?

39. Write C++ code to overload the `+` and `*` operators for a `Complex` class.

40. How do you distinguish prefix and postfix increment operators when overloading?

41. Write a class that overloads the subscript operator `[]` with bounds checking.

42. What is the purpose of friend functions in operator overloading? Provide an example.

43. Explain why destructors cannot be overloaded but can be virtual.

44. Find and fix errors in this operator overloading code:

```cpp
Coord Coord::operator++() {
    x++;
    y++;
}
```

45. Explain `dynamic_cast` in C++ and how it differs from `static_cast`.

46. Why does Java not support operator overloading? How does Java compensate for it?

47. What is auto-boxing and unboxing in Java? Give examples.

48. Write Java code to override a method and call the base class method using `super`.

49. Describe function overloading ambiguities in C++ and how Java avoids them.

50. Write a C++ program demonstrating a memory leak caused by missing virtual destructor and fix it.

51. How can default arguments cause ambiguity in function overloading? Provide an example.

Debugging & Error-Finding (10 questions)

52. Find the logical error in this C++ exception handling code:

```cpp
try {
    throw "Error";
} catch (int e) {
    cout << "Caught int" << endl;
}
```

53. Why is this Java code failing to compile? Fix it.

```java
class Test {
    void foo() throws IOException {
        throw new IOException();
    }
    void bar() {
        foo();
    }
}
```

54. In C++, why does this multiple inheritance code produce ambiguity?

```cpp
class A { public: void show() {} };
class B : public A {};
class C : public A {};
class D : public B, public C {};
```

55. Find the error and fix in this C++ operator overloading function for assignment operator:

```cpp
MyClass& operator=(MyClass ob) {
    this->x = ob.x;
    return *this;
}
```

56. Why does this Java overriding example not override the base method?

```java
class Base {
    public void display() {}
}
class Derived extends Base {
    public void display(int x) {}
}
```

57. What is the output of this C++ virtual function example? Explain.

```cpp
class Base {
public:
    virtual void show() { cout << "Base"; }
};

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

int main() {
    Base *b = new Derived();
    b->show();
    delete b;
}
```

58. Identify and fix problems in this Java interface implementation:

```java
interface A {
    void show();
}

class B implements A {
}
```

59. Explain why this code causes compilation error and how to fix it:

```cpp
class MyClass {
public:
    MyClass(int x = 0);
    MyClass();
};
```

60. Write a C++ program that throws an exception, but the exception is not caught anywhere. What is the program behavior?