Section 1: Theoretical Questions (Answer Briefly)

1. Explain the difference between an interface and an abstract class in Java. When would you use one over the other?

Ans.

Feature	Interface	Abstract Class
Definition	A collection of abstract methods and constants.	A class that can have abstract and concrete methods.
Methods	Only abstract methods (Java 7); default and static methods allowed (Java 8+).	Can have both abstract and concrete methods.
Fields	Only public, static, and final fields.	Can have instance variables (with any access modifier).
Inheritance	A class can implement multiple interfaces.	A class can extend only one abstract class.
Use Case	When multiple inheritance is needed or for defining a contract.	When partial implementation is needed with shared behavior.

2. What is the purpose of the final, finally, and finalize keywords in Java? Provide examples.

Ans. final, finally, finalize:

- final:
  - For variables: Makes them constants (their value cannot be changed).
  - For methods: Prevents them from being overridden.
  - For classes: Prevents them from being subclassed.
  - Example: final int x = 10;
- finally:
  - A block of code that is always executed after a try block, regardless of whether an exception occurred.
  - Used for cleanup operations (closing resources).
  - Example:

```
try {
    // ... code that might throw an exception
```

```
} finally {
    // ... cleanup code
}
```

1. finalize:
    - A method called by the garbage collector before an object is reclaimed.
    - Used for cleanup of native resources (less commonly used now).
    - It is generally recomended to not use finalize due to unpredictable behavior.
    - Example:

```
@Override
protected void finalize() throws Throwable {
    // ... cleanup native resources
    super.finalize();
}
```

3. How does Java handle memory management and garbage collection? What is the role of the JVM in this process?

Ans. Memory Management and Garbage Collection:

- Java uses automatic memory management through garbage collection.
- The JVM (Java Virtual Machine) is responsible for allocating and deallocating memory.
- The garbage collector identifies and reclaims objects that are no longer reachable (not referenced by any active part of the program).
- This process helps prevent memory leaks and simplifies memory management for developers.