Section 1: Introduction to Java

1. JVM Architecture

1. **Which of the following statements about JVM's memory management is correct?**

    a) The JVM stack stores objects, and the heap stores method frames.

    b) The heap stores objects, and the JVM stack stores method frames and local variables.

    c) Both the JVM stack and heap store objects.

    d) The heap stores method frames, and the JVM stack stores class information.

2. **In JVM, which phase of class loading ensures that there are no illegal or invalid bytecode instructions?**

    a) Linking

    b) Initialization

    c) Verification

    d) Resolution

2. Installation of Java

3. **After installing Java, you set the JAVA_HOME variable. Which of the following commands would not be able to find Java if JAVA_HOME is set incorrectly?**

    a) javac

    b) java

    c) java -version

d) All of the above

3. Configuring SDE Eclipse

4. **Which of the following issues might arise if you have an incorrect JRE System Library set up in your Eclipse IDE?**

    a) Compile-time errors due to missing library references

    b) Runtime errors due to incompatible class files

    c) Slow performance during debugging

    d) Both a and b

4. Understanding JRE and JVM

5. **Which of the following statements best explains the relationship between JVM, JRE, and JDK?**

    a) JDK includes JRE, which includes JVM.

    b) JRE includes JDK, which includes JVM.

    c) JVM includes JRE, which includes JDK.

    d) JDK includes JVM, which includes JRE.

Section 2: Object-Oriented Programming

1. Working on Constructors

6. **Which of the following statements about constructors in Java is incorrect?**

    a) Constructors can be overloaded.

    b) Constructors can be inherited.

    c) Constructors cannot return a value.

    d) Constructors can have access modifiers.

2. Achieving Encapsulation

7. **Given a class with private fields and public getter/setter methods, which of the following would break encapsulation?**

   a) Direct access to private fields within the class itself

   b) Modifying private fields directly from a subclass

   c) Using setter methods to modify private fields

   d) None of the above

3. Code Reusability via Inheritance

8. **Which scenario best demonstrates the concept of method overriding in Java?**

   a) A superclass method being hidden by a subclass method with the same name and different signature.

   b) A superclass method being redefined in a subclass with the same name and same signature.

   c) A superclass method being called by a subclass method.

   d) A superclass method being redefined in a subclass with a different name.

4. Achieving Polymorphism

9. **Which of the following is an example of run-time polymorphism?**

   a) Method overloading

   b) Method overriding

   c) Operator overloading

   d) Both a and b

5. Working on methods of java.lang.Object class

10. **What is the significance of the hashCode() method in the java.lang.Object class?**

    a) It determines the bucket location in hash-based collections like HashMap.

    b) It determines the equality of two objects.

    c) It creates a deep copy of an object.

    d) It is used for synchronization.

6. Object Casting

11. **Which of the following statements about downcasting is true?**

    a) Downcasting is always safe and does not require explicit casting.

    b) Downcasting can fail at runtime if the actual object type does not match the cast type.

    c) Downcasting can be performed using implicit casting.

    d) Downcasting is only required for converting primitive types.

7. Passing Objects as Arguments

12. **Which of the following demonstrates that Java uses "pass-by-value" even for object references?**

    a) Changing the state of an object inside a method reflects outside the method.

    b) Assigning a new object to a parameter reference inside a method does not affect the original reference.

    c) Both a and b

d) Neither a nor b

8. Abstraction via Abstract Classes and Interfaces

13. **Which of the following statements is true about abstract methods in interfaces as compared to abstract classes?**

    a) Abstract methods in interfaces cannot have default implementations.

    b) Abstract methods in interfaces must be public and abstract.

    c) Abstract methods in interfaces can be protected.

    d) Abstract methods in interfaces can have any access modifier.

9. Diamond Problem using Interfaces

14. **Given two interfaces, A and B, both having a default method with the same signature, and a class C implements both interfaces, how can class C resolve the diamond problem?**

    a) Override the conflicting method and use A.super.method() or B.super.method() to specify which interface's method to call.

    b) Declare the class as abstract.

    c) Instantiate an object of the interface type.

    d) It is not possible to resolve the conflict.

10. Creating Static Classes and Static Methods

15. **Which of the following scenarios correctly demonstrates the use of a static method?**

    a) Calling a static method using an instance of the class.

    b) Accessing non-static members from within a static method.

c) Accessing static members using the class name without creating an instance.

d) Overriding a static method in a subclass.

Section 3: Wrapper Classes

1. Java Keywords

16. **Which of the following is not a valid use of the 'final' keyword?**

    a) To declare constants

    b) To prevent inheritance

    c) To prevent method overriding

    d) To make a class immutable

2. Primitive Data Types

17. **What would be the output of the following code snippet?**

```
int x = 10;
float y = 10.0f;
if (x == y) {
    System.out.println("Equal");
} else {
    System.out.println("Not Equal");
}
```

    a) Equal

    b) Not Equal

    c) Compilation error

    d) Runtime error

3. Using Operators

18. **Given the following code, what will be the value of result?**

```
int result = 10;
result += (result++ * 2) + (--result);
```

a) 31

b) 32

c) 33

d) 34

4. Using if-else and switch statements

19. **Which of the following switch cases is correctly defined and covers all potential int values for a variable x?**

```
switch (x) {
   // Cases here
}
```

a) case 1:, case 2:, default:

b) case -1:, case 0:, case 1:, default:

c) case Integer.MIN_VALUE:, case 0:, case Integer.MAX_VALUE:

d) None of the above

Iterating with loops: while, do-while, for, enhanced for

20. **What will be the output of the following code snippet?**

```
int[] arr = {1, 2, 3, 4, 5};
for (int i :
```

arr) { if (i % 2 == 0) continue; System.out.print(i + " "); } ```

a) 1 3 5
b) 2 4
c) 1 2 3 4 5
d) 1 2 3 4

6. Wrapper Classes and Autoboxing Concepts

21. **What is the result of autoboxing in the following code?**

```
Integer i = 1000;
Integer j = 1000;
System.out.println(i == j);
```

a) true

b) false

c) Compilation error

d) Runtime error

7. Single-Dimensional Array

22. **Given the following code, what will be the output?**

```
int[] arr = new int[5];
arr[2] = 10;
for (int i : arr) {
   System.out.print(i + " ");
}
```

a) 0 0 10 0 0

b) 10 0 0 0 0

c) 10

d) 0 0 0 0 10

8. Multi-Dimensional Arrays

23. **Which of the following correctly declares and initializes a 2D array in Java?**

    a) int[][] arr = {{1, 2}, {3, 4}};

    b) int[][] arr = new int[2][2]{{1, 2}, {3, 4}};

    c) int arr[][] = {1, 2, 3, 4};

    d) int[][] arr = new int[2][2]; arr[0] = {1, 2}; arr[1] = {3, 4};

9. Array of Objects

24. **Given the following code, what will be the output?**

```
class Person {
    String name;
    Person(String name) {
        this.name = name;
    }
}

public class Test {
    public static void main(String[] args) {
        Person[] people = new Person[2];
        people[0] = new Person("Vijay");
        people[1] = new Person("Smitha");
        for (Person p : people) {
            System.out.println(p.name);
        }
    }
}
```

    a) Vijay Smitha

    b) Vijay Vijay

    c) Smitha Smitha

d) Compilation error

10. Arrays Utility Class

25. **Which method of the Arrays class is used to fill an array with a specified value?**
    - a) fill()
    - b) populate()
    - c) set()
    - d) assign()

Section 4: String Classes

1. String Class

26. **Which of the following operations creates a new String object?**

```
String str1 = "hello";
String str2 = new String("hello");
```

a) Both str1 and str2 refer to the same object in memory.

b) str1 refers to a literal pool object, str2 refers to a new object.

c) Both str1 and str2 refer to new objects.

d) str1 refers to a new object, str2 refers to a literal pool object.

2. StringBuffer class

27. **Which method of StringBuffer is used to insert a string at a specified index?**

a) append()

b) insert()

c) prepend()

d) add()

3. StringBuilder class

28. **Which of the following statements about StringBuilder is incorrect?**

    a) StringBuilder is synchronized.

    b) StringBuilder is mutable.

    c) StringBuilder has a default capacity of 16 characters.

    d) StringBuilder can be used in multi-threaded environments without additional synchronization.

4. Introduction to Regex (Regular Expression)

29. **In regex, what does the pattern [^0-9a-zA-Z] match?**

    a) Any alphanumeric character

    b) Any digit or letter

    c) Any non-alphanumeric character

    d) Any digit only

Section 5: Working with Exceptions

1. Defining the Purpose of Java Exceptions

30. **Which of the following statements is true about checked and unchecked exceptions in Java?**

    a) Checked exceptions are checked at compile time, while unchecked exceptions are checked at runtime.

    b) Checked exceptions are checked at runtime, while unchecked exceptions are checked at compile time.

    c) Both checked and unchecked exceptions are checked at compile time.

d) Both checked and unchecked exceptions are checked at runtime.