# 1Z0-808 Mock Exam

ExamId: 101
Items: 56
Dificulty: HARD

August 5, 2025

(1) (questionId: 102228) What is the result of attempting to compile and run the Test class?

```
interface I1 {
    default void go() { System.out.println("I1"); }
}
interface I2 {
    default void go() { System.out.println("I2"); }
}
class C1 implements I1, I2 {
    public void go() {
        I1.super.go();
    }
}
public class Test {
    public static void main(String[] args) {
        new C1().go();
    }
}
```

Choose the most correct answer.

- 0) A compile-time error at ‘class C1‘.
- 1) The code compiles and prints ”I1”.
- 2) The code compiles and prints ”I2”.
- 3) A compile-time error at ‘I1.super.go();‘ because ‘super‘ can only be used with classes.

(2) (questionId: 103024) What is the result of compiling this code?

```
import java.io.*;

public class CatchOrder {
    public void process() {
        try {
            if (System.currentTimeMillis() % 2 == 0) {
                throw new IOException();
            } else {
                throw new FileNotFoundException();
            }
        } catch (IOException e) { // line X
            System.out.println("IO");
        } catch (FileNotFoundException e) { // line Y
            System.out.println("File Not Found");
        }
    }
}
```

Choose the most correct answer.

- 0) Compilation succeeds.
- 1) Compilation fails at line X.
- 2) Compilation fails at line Y.
- 3) Compilation fails at both line X and line Y.

(3) (questionId: 100124) An abstract class is defined as follows:

```
abstract class AbstractRunner {
    public static void main(String[] args) {
        System.out.println("Running from abstract class");
    }
}
```

What is the outcome of compiling and executing ‘java AbstractRunner‘? Choose the most correct answer.

- 0) Compilation fails.
- 1) An ‘InstantiationException‘ is thrown at runtime.
- 2) An ‘AbstractMethodError‘ is thrown at runtime.
- 3) It compiles and runs successfully, printing the message.

(4) (questionId: 100921) What is the result of attempting to compile and run the following class?

```
public class SwitchCaseConstant {
    public static void main(String[] args) {
        final int a = 1;
        final int b;
        b = 2;
        int x = 0;
        switch (x) {
            case a: // case 1
                System.out.print("A");
            case b: // case 2
                System.out.print("B");
        }
    }
}
```

Choose the most correct answer.

- 0) It prints ‘A‘.
- 1) It prints ‘B‘.
- 2) It prints ‘AB‘.
- 3) A compilation error occurs.

(5) (questionId: 103356) What is the result of executing the following code? This question tests the case-sensitivity and symbol correctness of formatter patterns.

```
import java.time.LocalDate;
import java.time.format.DateTimeFormatter;

public class PatternCaseTest {
    public static void main(String[] args) {
        String dateStr = "2-8-2025";
        DateTimeFormatter f = DateTimeFormatter.ofPattern("d-m-yyyy");
        LocalDate date = LocalDate.parse(dateStr, f);
        System.out.println(date);
    }
}
```

Choose the most correct answer.

- 0) ‘2025-08-02‘
- 1) ‘2025-02-08‘
- 2) The code does not compile.
- 3) A ‘DateTimeParseException‘ is thrown.

(6) (questionId: 100229) Given the file ‘pkg/A.java‘:

```
package pkg;
public class A {
    public void print() { System.out.println("A"); }
}
```

And the file ‘B.java‘:

```
import pkg.A;
public class B {
    public static void main(String[] args) {
        A a = new A();
        a.print();
    }
}
```

From the project root, which command sequences will compile and run the code successfully? (Assume Linux/macOS). (Choose all that apply) Choose all the correct answer.

- 0) ‘javac pkg/A.java B.java; java B‘
- 1) ‘javac -d . pkg/A.java B.java; java B‘
- 2) ‘javac B.java; java B‘ (assuming ‘pkg/A.class‘ already exists)
- 3) ‘javac pkg/A.java B.java; java -cp . B‘
- 4) ‘javac B.java; java -cp . B‘ (assuming ‘pkg/A.class‘ does not exist)

(7) (questionId: 101225) Which of the following code snippets will result in a compilation error? (Choose all that apply) Choose all the correct answer.

- 0)

  ```
  public enum E1 { A, B; private E1() {} }
  ```

- 1)

  ```
  public enum E2 { C, D; protected E2() {} }
  ```

- 2)

  ```
  public enum E3 { E, F; E3() {} }
  ```

- 3)

  ```
  public enum E4 { G, H; public E4() {} }
  ```

(8) (questionId: 100523) Examine the following code. What will be the outcome?

```
final int i = 10;
byte b = i;
System.out.println(b);
```

Choose the most correct answer.

- 0) The code fails to compile because a cast '(byte)' is required.
- 1) The code compiles and prints '10'.
- 2) The code fails to compile because 'i' is final and cannot be assigned.
- 3) The code compiles but throws a runtime exception.

(9) (questionId: 100028) You are in the directory `/root`. You have the following files: `/root/com/example/App.java` `/root/lib/helper.jar` The class `App` depends on a class inside `helper.jar`. Which command(s) will successfully compile `App.java`? (Choose all that apply) Choose all the correct answer.

- 0) `javac -cp lib/helper.jar com/example/App.java`
- 1) `javac -classpath lib/helper.jar com/example/App.java`
- 2) `javac com/example/App.java -cp lib/helper.jar`
- 3) `javac -cp lib/helper.jar;com/example/App.java`
- 4) `javac -d . -cp lib/helper.jar com/example/App.java`

(10) (questionId: 103453) Consider an interface with a static method (a Java 8 feature). What is the result of this code?

```
// File: I.java
public interface I {
    static void run() { System.out.println("I"); }
}

// File: C.java
public class C {
    public static void run() { System.out.println("C"); }
}
```

```
// File: Main.java
import static I.*;
import static C.*;

public class Main {
    public static void main(String[] args) {
        run();
    }
}
```

Choose the most correct answer.

- 0) It prints ‘I‘.
- 1) It prints ‘C‘.
- 2) The code fails to compile due to ambiguity.
- 3) The code fails to compile because you cannot statically import methods from an interface.

(11) (questionId: 103650) What is the output of the following code? This question tests ‘final‘ parameters.

```
class Box { public int size; }

public class FinalParamTest {
    public static void modify(final Box b) {
        b.size = 100;
        // b = new Box(); // This line is commented out
    }

    public static void main(String[] args) {
        Box box = new Box();
        box.size = 10;
        modify(box);
        System.out.println(box.size);
    }
}
```

Choose the most correct answer.

- 0) ‘10‘
- 1) ‘100‘
- 2) The code fails to compile because a method cannot modify a ‘final‘ parameter.
- 3) The code fails to compile for another reason.

(12) (questionId: 100024) What is the result of compiling and running the following class?

```
public class Test {
    static {
        System.out.print("Static block. ");
    }

    public static void main(String[] args) {
        System.out.print("Main method.");
    }
}
```

Choose the most correct answer.

- 0) Main method.
- 1) Static block. Main method.
- 2) Main method. Static block.
- 3) Compilation fails.

(13) (questionId: 103124) What is the result of attempting to compile this code?

```
class Box implements AutoCloseable {
    private void close() throws Exception {}
}
public class TestPrivateClose {
    public static void main(String[] args) {
        try (Box b = new Box()) {
            // ...
        }
    }
}
```

Choose the most correct answer.

- 0) The code compiles but fails at runtime with an 'IllegalAccessException'.
- 1) A compilation error occurs because the 'close()' method is private.
- 2) The code compiles and runs without issue, as the JVM can access the private method.
- 3) A compilation error occurs because the 'main' method doesn't handle the 'Exception' from 'close()'.

(14) (questionId: 102025) Which line causes a compilation error?

```
class T1 {
    T1() { super(); }
    T1(int i) { this(); }
}
class T2 extends T1 {
    T2() { super(5); }
    T2(int i) { this(); }
    T2(String s) {}
```

```
}
```

Choose the most correct answer.

- 0) ‘T1()  super(); ‘
- 1) ‘T1(int i)  this(); ‘
- 2) ‘T2()  super(5); ‘
- 3) ‘T2(int i)  this(); ‘
- 4) ‘T2(String s) ‘

(15) (questionId: 101521) What is the output of the following code?

```
public class Chain {
    private int value;

    public Chain() {
        this(5);
        System.out.print("A");
    }

    public Chain(int value) {
        this(value, "X");
        System.out.print("B");
        this.value += value;
    }

    public Chain(int value, String s) {
        System.out.print(s);
        this.value = value;
    }

    public static void main(String[] args) {
        Chain c = new Chain();
        System.out.print(c.value);
    }
}
```

Choose the most correct answer.

- 0) XBA5
- 1) ABX10
- 2) XBA10
- 3) The code fails to compile.

(16) (questionId: 101721) What is the output of the following code? This question tests method hiding.

```
class Animal {
```

```
    static void eat() { System.out.println("Animal eats"); }
}
class Dog extends Animal {
    static void eat() { System.out.println("Dog eats"); }
}
public class Test {
    public static void main(String[] args) {
        Animal myAnimal = new Dog();
        myAnimal.eat();
    }
}
```

Choose the most correct answer.

- 0) Animal eats
- 1) Dog eats
- 2) The code fails to compile.
- 3) A runtime exception is thrown.

(17) (questionId: 101326) Which of the following code snippets will result in ‘s2‘ referring to the same object as ‘s1‘ in the string pool? (Choose all that apply)

```
String s1 = "Test";
```

Choose all the correct answer.

- 0) ‘String s2 = ”Test”;‘
- 1) ‘String s2 = new String(”Test”);‘
- 2) ‘String s2 = new String(”Test”).intern();‘
- 3) ‘String s2 = ”Te” + ”st”;‘

(18) (questionId: 100629) What is the output of the code?

```
public class Test {
    public static void main(String[] args) {
        Integer a = 10;
        Integer b = 10;
        Integer c = a + b;
        Integer d = 20;
        System.out.println(c == d);
    }
}
```

Choose the most correct answer.

- 0) `true`
- 1) `false`
- 2) Compilation fails.

- 3) An exception is thrown at runtime.

(19) (questionId: 101524) What is the output of the following code?

```
class Wallet {
    public int cash;
}

public class Thief {
    public static void main(String[] args) {
        Wallet w = new Wallet();
        w.cash = 100;
        steal(w);
        System.out.println(w.cash);
    }

    public static void steal(Wallet victimWallet) {
        victimWallet.cash -= 50;
        victimWallet = new Wallet(); // Thief gets a new wallet
        victimWallet.cash = 10;
    }
}
```

Choose the most correct answer.

- 0) 100
- 1) 50
- 2) 10
- 3) 0

(20) (questionId: 100428) What happens when this code is compiled and run?

```
System.out.println(10 / 0);
```

Choose the most correct answer.

- 0) It fails to compile.
- 1) It prints ‘Infinity‘.
- 2) It prints ‘NaN‘.
- 3) It compiles but throws an ‘ArithmeticException‘ at runtime.

(21) (questionId: 101823) Which of these statements are true regarding Java’s memory management and garbage collection? (Choose all that apply) Choose all the correct answer.

- 0) Objects are stored on the heap, while object references are typically stored on the stack.
- 1) The ‘finalize()‘ method is a reliable mechanism for cleaning up critical resources like database connections.

- 2) An 'island of isolation' refers to a group of objects that reference each other but have no external reachable references, making them eligible for GC.
- 3) Generational garbage collectors divide the heap into young and old generations to improve efficiency, assuming most objects die young.
- 4) Calling 'System.exit(0)' will trigger garbage collection and finalization for all live objects before the JVM shuts down.

(22) (questionId: 101021) What is the output of this code with labeled statements?

```
public class LabeledBreak {
    public static void main(String[] args) {
        outer:
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {
                if (i == 1) {
                    break outer;
                }
                System.out.print(i + "" + j + " ");
            }
        }
    }
}
```

Choose the most correct answer.

- 0) 00 01 02
- 1) 00 01 02 20 21 22
- 2) 00 01 02 10 11 12 20 21 22
- 3) The code does not compile.

(23) (questionId: 100224) You execute a program with 'java -jar myapp.jar'. The manifest file inside 'myapp.jar' contains the line 'Class-Path: lib/utils.jar'. The JVM will: Choose the most correct answer.

- 0) Ignore the 'Class-Path' attribute in the manifest.
- 1) Automatically add 'lib/utils.jar' to the classpath.
- 2) Throw an error because 'Class-Path' is not a valid manifest attribute.
- 3) Only use 'lib/utils.jar' if the '-cp' flag is also specified.

(24) (questionId: 102323) Examine the following code. What is its result?

```
public class Runner {
    public static void main(String[] args) {
        int y = 1;
        Runnable r = () -> {
            // Line 1
            System.out.println(y);
        };
```

```
        // Line 2
    }
}
```

What happens if the statement ‘y = 2;‘ is placed at Line 2? Choose the most correct answer.

- 0) The code compiles and runs fine.
- 1) The code fails to compile due to an error at Line 1 (‘System.out.println(y);‘).
- 2) The code fails to compile due to an error at ‘y = 2;‘ because ‘y‘ is now effectively final.
- 3) The code compiles but throws a runtime exception.

(25) (questionId: 101421) What is the output of the following program?

```
public class Test {
    public static void main(String[] args) {
        StringBuilder sb = new StringBuilder("Initial");
        reassign(sb);
        System.out.print(sb + ":");
        modify(sb);
        System.out.print(sb);
    }
    static void reassign(StringBuilder sb) {
        sb = new StringBuilder("New");
    }
    static void modify(StringBuilder sb) {
        sb.append("-Mod");
    }
}
```

Choose the most correct answer.

- 0) ‘Initial:Initial-Mod‘
- 1) ‘New:New-Mod‘
- 2) ‘Initial:Initial‘
- 3) ‘New:Initial-Mod‘

(26) (questionId: 102625) Due to type erasure, what does the following generic class effectively become after compilation?

```
public class Node<T extends Comparable<T>> {
    private T data;
    private Node<T> next;
    public Node(T data, Node<T> next) {
        this.data = data;
        this.next = next;
    }
    public T getData() { return data; }
```

```
}
```

Choose the most correct answer.

- 0)

```
public class Node {
    private Comparable data;
    private Node next;
    // ... constructor and methods with casts
}
```

- 1)

```
public class Node {
    private Object data;
    private Node next;
    // ... constructor and methods with casts
}
```

- 2)

```
public class Node<Comparable> {
    private Comparable data;
    private Node<Comparable> next;
    // ...
}
```

- 3) The generic information is retained fully in the bytecode.

(27) (questionId: 101320) What is the output of the following code?

```
final String f = "Ja";
String s1 = f + "va";
String s2 = "Java";
System.out.println(s1 == s2);
```

Choose the most correct answer.

- 0) ‘true‘
- 1) ‘false‘
- 2) The code does not compile because ‘f‘ is ‘final‘.
- 3) An exception is thrown at runtime.

(28) (questionId: 100426) What is printed to the console by the following code?

```
int value = 'a' + 'b';
System.out.println(value);
```

Choose the most correct answer.

- 0) ‘ab‘
- 1) ‘195‘

- 2) The code fails to compile.
- 3) ‘9798‘

(29) (questionId: 100321) What is the result of attempting to compile the following code?

```
public class NestedComment {
    /*
     * This is an outer comment.
     * /* This is a nested comment. */
     * The outer comment ends here.
     */
    public static void main(String[] args) {
        System.out.println("Hello");
    }
}
```

Choose the most correct answer.

- 0) Compilation is successful, and the program prints ”Hello”.
- 1) Compilation fails due to an unclosed comment.
- 2) Compilation is successful, but a warning is issued about nested comments.
- 3) Compilation fails due to illegal syntax inside a comment.

(30) (questionId: 102125) What is the output of the following code?

```
class Parent {
    void process(Object o) {
        System.out.println("Parent-Object");
    }
}
class Child extends Parent {
    @Override
    void process(Object o) {
        System.out.println("Child-Object");
    }
    void process(String s) {
        System.out.println("Child-String");
    }
}
public class Test {
    public static void main(String[] args) {
        Parent p = new Child();
        p.process("test");
    }
}
```

Choose the most correct answer.

- 0) Parent-Object

- 1) Child-Object
- 2) Child-String
- 3) The code fails to compile.

(31) (questionId: 102424) Which of the following statements are true? (Choose all that apply) Choose all the correct answer.

- 0) `int[] x, y[];` declares `x` as a 1D array and `y` as a 2D array.
- 1) An array's size can be changed after it has been created.
- 2) `new int[0]` creates an array of size 0.
- 3) An `ArrayStoreException` is a checked exception.

(32) (questionId: 103226) What is the result of the following code?

```
import java.util.function.Function;

public class TrickyThis {
    private String value = "Enclosing";

    public Function<String, String> create() {
        return x -> this.value + ":" + x;
    }

    public static void main(String[] args) {
        TrickyThis t = new TrickyThis();
        System.out.println(t.create().apply("Lambda"));
    }
}
```

Choose the most correct answer.

- 0) 'Enclosing:Lambda'
- 1) 'Lambda:Enclosing'
- 2) A compilation error occurs due to the use of 'this'.
- 3) A 'NullPointerException' is thrown at runtime.

(33) (questionId: 102525) Which code snippet demonstrates the correct way to create a generic 'ArrayList' that can hold any subclass of 'Number'? Choose the most correct answer.

- 0) `List<? super Number> list = new ArrayList<Integer>();`
- 1) `List<? extends Number> list = new ArrayList<Integer>();`
- 2) `List<T extends Number> list = new ArrayList<T>();`
- 3) `List<Number> list = new ArrayList<Integer>();`

(34) (questionId: 102822) What is the outcome of compiling and running the following code?

```
public class Test {
    static {
        if (true) {
            throw new NullPointerException("Error in static block");
        }
    }
    public static void main(String[] args) {
        System.out.println("Hello");
    }
}
```

Choose the most correct answer.

- 0) A 'NullPointerException' is caught by the JVM and 'Hello' is printed.
- 1) The program prints 'Hello' and exits normally.
- 2) An 'ExceptionInInitializerError' is thrown, and the program terminates.
- 3) A 'NullPointerException' is thrown, and the program terminates.

(35) (questionId: 103557) Which of the following method calls are ambiguous and will cause a compilation error? (Choose all that apply)

```
class Ambiguity {
    static void m(int a, long b) {} // M1
    static void m(long a, int b) {} // M2
    static void m(int... a) {}      // M3
    static void m(Number n) {}      // M4
    static void m(Object o) {}      // M5
}
```

Choose all the correct answer.

- 0) 'Ambiguity.m(5, 10);'
- 1) 'Ambiguity.m(5L, 10L);'
- 2) 'Ambiguity.m(5);'
- 3) 'Ambiguity.m(new Integer(5));'
- 4) 'Ambiguity.m(null);'

(36) (questionId: 101929) Examine the code:

```
public final class MyData {
    private final StringBuilder builder;

    public MyData(StringBuilder b) {
        this.builder = b;
    }

    public StringBuilder getBuilder() {
        return builder;
```

```
    }
}

// Main method in another class
StringBuilder sb = new StringBuilder("Initial");
MyData data = new MyData(sb);
sb.append(" Changed");
System.out.println(data.getBuilder());
```

What is the output? Choose the most correct answer.

- 0) Initial
- 1) Initial Changed
- 2) A new ‘StringBuilder‘ object’s string representation.
- 3) Compilation fails because ‘final‘ fields cannot be assigned in a constructor.
- 4) Compilation fails because ‘StringBuilder‘ is mutable.

(37) (questionId: 100726) What is printed to the console?

```
public class TrickyScope {
    static TrickyScope ts = new TrickyScope();
    static int val = 10;
    {
        // instance initializer
        val = 20;
    }

    public static void main(String[] args) {
        System.out.println(val);
    }
}
```

Choose the most correct answer.

- 0) 10
- 1) 20
- 2) 0
- 3) Compilation fails.

(38) (questionId: 101427) Given ‘StringBuilder sb = new StringBuilder(”abcde”);‘. Which statements about its capacity are true? (Choose all that apply) Choose all the correct answer.

- 0) The initial capacity is 21 (5 for ”abcde” + 16 default).
- 1) ‘sb.trimToSize();‘ will likely change its capacity to 5.
- 2) After ‘sb.append(”fghijklmnopqrstuvwxyz”);‘, the capacity will be larger than its length.

- 3) ‘sb.ensureCapacity(10);‘ will not change the capacity.

(39) (questionId: 100728) Which statements about the following code are correct? (Choose all that apply)

```
public class Outer {
    private String name = "Outer";

    class Inner {
        private String name = "Inner";

        void printNames() {
            String name = "Local";
            System.out.println(name);
            System.out.println(this.name);
            System.out.println(Outer.this.name);
        }
    }

    public static void main(String... args) {
        new Outer().new Inner().printNames();
    }
}
```

Choose all the correct answer.

- 0) The code will fail to compile.
- 1) The output will be: Local
- 2) The output will be: Local Inner Outer
- 3) `this.name` refers to the instance variable of the `Inner` class.
- 4) `Outer.this.name` is used to access the instance variable of the enclosing `Outer` class.

(40) (questionId: 100121) What is the result of attempting to compile and run the following code?

```
public class TrickyMain {
    public static void main(String args) {
        System.out.println("Hello");
    }
}
```

Choose the most correct answer.

- 0) It compiles and runs, printing ”Hello”.
- 1) It fails to compile because the ‘main‘ parameter is not an array.
- 2) It compiles, but at runtime the JVM reports that ‘main‘ is not found.
- 3) It compiles and runs, but ‘args‘ is null.

(41) (questionId: 101127) Consider the following code. Which line causes a compilation error?

```
label1: while (true) {      // Line 1
    int x = 0;               // Line 2
    label2: do {             // Line 3
        x++;                 // Line 4
        continue label1;     // Line 5
    } while(x < 5);          // Line 6
    break label2;            // Line 7
}
```

Choose the most correct answer.

- 0) Line 3
- 1) Line 5
- 2) Line 7
- 3) The code compiles without errors.

(42) (questionId: 100628) Examine this code. What will be printed to the console?

```
public class Test {
    public static void main(String[] args) {
        Integer i1 = 10;
        Long l1 = 10L;

        System.out.println(i1.equals(l1));
    }
}
```

Choose the most correct answer.

- 0) `true`
- 1) `false`
- 2) The code does not compile.
- 3) A runtime exception is thrown.

(43) (questionId: 101822) What is the final value of ‘count‘ printed to the console?

```
public class GCCount {
    static int count = 0;
    int id;

    public GCCount(int id) { this.id = id; }

    public static void main(String[] args) {
        new GCCount(1);
        GCCount g2 = new GCCount(2);
        GCCount g3 = new GCCount(3);
```

```
        g2 = g3;
        new GCCount(4);
        g3 = null;
        // Point X
        System.gc();
        System.out.println(count);
    }

    @Override
    protected void finalize() {
        count++;
    }
}
```

Choose the most correct answer.

- 0) 0
- 1) 2
- 2) 3
- 3) 4
- 4) The output is not guaranteed.

(44) (questionId: 101224) What is true about the serialization of enums? Choose the most correct answer.

- 0) Enums are not serializable by default and require implementing ‘java.io.Serializable‘ and defining a ‘serialVersionUID‘.
- 1) When an enum is deserialized, the constructor is called again to create a new instance.
- 2) Java’s serialization mechanism ensures that deserializing an enum constant will always return the pre-existing constant instance, thus preserving singleton identity.
- 3) Deserializing an enum may result in a different object instance if the enum declaration has changed, causing ‘==‘ to fail.

(45) (questionId: 101625) What is the result of attempting to compile and run the following code?

```
abstract class Builder {
    Builder() { System.out.print("B"); }
}

public class House extends Builder {
    House() {
        // super() is implicitly called here
        System.out.print("H");
    }
```

```
    public static void main(String[] args) {
        new House();
    }
}
```

Choose the most correct answer.

- 0) The code fails to compile because an abstract class cannot have a constructor.
- 1) The code compiles and prints "BH".
- 2) The code compiles and prints "HB".
- 3) The code fails to compile because 'new Builder()' is not allowed.

(46) (questionId: 100821) What is the final value of 'a'?

```
int a = 2;
a = a++ * a++;
```

Choose the most correct answer.

- 0) 4
- 1) 6
- 2) 8
- 3) 9

(47) (questionId: 101723) What is the output of the following code? This question tests static initialization order.

```
public class Init {
    static { a = b * 2; }
    static int a = 10;
    static int b = 5;
    static { a = b * 3; }

    public static void main(String[] args) {
        System.out.println(a);
    }
}
```

Choose the most correct answer.

- 0) 10
- 1) 15
- 2) 30
- 3) The code fails to compile.

(48) (questionId: 100929) What is true about the following code snippet? (Choose all that apply)

```
public class Tricky {
    public static void main(String[] args) {
        boolean a = true, b = false, c = false;
        if ( a || (b=true) && (c=true) )
            ;
        System.out.println(a + " " + b + " " + c);
    }
}
```

Choose all the correct answer.

- 0) The output is ‘true false false‘.
- 1) The output is ‘true true true‘.
- 2) The variable ‘b‘ is assigned ‘true‘ during the evaluation of the ‘if‘ condition.
- 3) The variable ‘c‘ is assigned ‘true‘ during the evaluation of the ‘if‘ condition.
- 4) The code does not compile.

(49) (questionId: 101629) Which statements correctly describe the complete order of initialization for an object ‘new Sub()‘ where ‘Sub extends Super‘? (Choose all that apply) Choose all the correct answer.

- 0) Static initializers of ‘Super‘ run before static initializers of ‘Sub‘.
- 1) All instance initializers (both ‘Super‘ and ‘Sub‘) run before any constructor code.
- 2) The constructor body of ‘Super‘ runs before the instance initializers of ‘Sub‘.
- 3) The constructor body of ‘Sub‘ is the very last thing to run for the ‘Sub‘ object’s initialization.
- 4) Static initializers of ‘Sub‘ run before instance initializers of ‘Super‘.
- 5) Instance initializers of ‘Sub‘ run before the constructor body of ‘Sub‘.

(50) (questionId: 102921) What is the value returned by the method ‘check()‘?

```
public class Test {
    public static int check() {
        try {
            return 1;
        } catch (Exception e) {
            return 2;
        } finally {
            return 3;
        }
    }
    public static void main(String[] args) {
        System.out.println(check());
    }
}
```

Choose the most correct answer.

- 0) ‘1‘
- 1) ‘2‘
- 2) ‘3‘
- 3) The code does not compile.

(51) (questionId: 102725) What is the output?

```
class Legacy {
    public int compareTo(Object o) { return 0; }
}
class Generic extends Legacy implements Comparable<Generic> {
}
// in a method
Comparable c = new Generic();
System.out.println(c.compareTo("test"));
```

Choose the most correct answer.

- 0) ‘0‘
- 1) A ‘ClassCastException‘ is thrown at runtime.
- 2) The code does not compile.
- 3) The output is unpredictable.

(52) (questionId: 100528) What is the result of the following code snippet?

```
float f = (float) Double.POSITIVE_INFINITY;
int i = (int) f;
System.out.println(i);
```

Choose the most correct answer.

- 0) ‘0‘
- 1) ‘-1‘
- 2) ‘2147483647‘
- 3) A runtime ‘ArithmeticException‘ is thrown.

(53) (questionId: 100324) A class contains a method with the following Javadoc comment. What is the result of attempting to compile the source file containing this code?

```
/**
 * Processes a request.
 * @parameter name The name of the user.
 * @return The result of the processing.
 */
public String process(String name) { return "Processed: " + name; }
```

Choose the most correct answer.

- 0) Compilation fails because ‘@parameter‘ is not a valid Javadoc tag.
- 1) Compilation succeeds.
- 2) Compilation succeeds, but the ‘javadoc‘ tool will fail to execute.
- 3) Compilation fails with a warning about the unrecognized tag.

(54) (questionId: 100826) What is the output of the following program?

```
public class Test {
    public static void main(String[] args) {
        int x = 5;
        boolean b1 = true;
        boolean b2 = false;
        if ((x == 4) && !b2 )
            System.out.print("1 ");
        System.out.print("2 ");
        if ((b2 = true) && b1 )
            System.out.print("3 ");
    }
}
```

Choose the most correct answer.

- 0) 2
- 1) 2 3
- 2) 1 2 3
- 3) 1 2

(55) (questionId: 101027) What is the final value of ‘count‘?

```
int count = 0;
for (int i = 0; i < 5; i++) {
    for (int j = 0; j < 5; j++) {
        if (j == 2)
            continue;
        count++;
    }
}
```

Choose the most correct answer.

- 0) 25
- 1) 20
- 2) 15
- 3) 10

(56) (questionId: 101121) What is the result of attempting to compile and run this code?

```
public class LabeledBlock {
    public static void main(String[] args) {
        int x = 5;
        myBlock: {
            if (x == 5) {
                break myBlock;
            }
            System.out.print("Inside");
        }
        System.out.print("Outside");
    }
}
```

Choose the most correct answer.

- 0) It prints ‘InsideOutside‘.
- 1) It prints ‘Outside‘.
- 2) It prints ‘Inside‘.
- 3) It fails to compile.