1

Domain : Reusing Implementations Through Inheritance

Given:

```
class Foo {

    protected String myField = "Foo";


    protected Object myMethod() {

        return myField;

    }

}


public class Bar extends Foo {

    public String myField = "Bar";


    public String myMethod() {

        return myField;

    }


    public static void main(String[] args) {

        Foo foo = new Bar();

        System.out.println(foo.myField);

        System.out.println(foo.myMethod());

    }

}
```

What is the program's output?

A. Foo
Foo

B. Foo
Bar

C. Bar
Foo

D. Bar
Bar

E. Compilation fails

2

**Domain :** Using Operators and Decision Constructs

Given a method:

```
void switchString(String arg) {

    switch (arg) {

        case "A" | "B":

            System.out.println("Hi");

        default:

            System.out.println("Hello");

    }

}
```

What is printed to the console if the given method is called with argument "a"?

- A. Hi
- B. Hello
- C. Nothing
- D. Compilation fails

3

**Domain :** Working With Java Primitive Data Types and String APIs

Given a class:

```
public class Test {

    public static void main(String[] args) {

        int i, j = 1, k; // Line 1

        k = i + j; // Line 2

        System.out.println(k);

    }

}
```

What is the output of the program?

- A. 1
- B. 2
- C. Compilation fails on line 1
- D. Compilation fails on line 2
- E. Compilation fails on line 1 and line 2

4

**Domain :** Handling Exceptions

```
Given:
public class Test {
    public static void main(String[] args) {
        try {
            throwException(); // Line 1
        } catch (IOException e) { // Line 2
            e.printStackTrace();
        }
    }
    static void throwException() throws RuntimeException { // Line 3
        throw new RuntimeException();
    }
}
```

Which lines of code cause a compile-time error?

- A. Line 1 only
- B. Line 2 only
- C. Line 3 only
- D. Line 1 and line 2
- E. Line 2 and line 3
- F. Line 3 and line 1
- G. Compilation succeeds

**Domain :** Programming Abstractly Through Interfaces

Given:

```
interface Foo {
    String myField; // Line 1
    String myMethod(); // Line 2
}
public abstract class Bar implements Foo { // Line 3
    abstract void myMethod(String arg);  // Line 4
}
```

Which line fails to compile?

- A. Line 1
- B. Line 2
- C. Line 3
- D. Line 4
- E. Compilation succeeds

6

**Domain :** Reusing Implementations Through Inheritance

Which of the following describes polymorphism in Java?

A. **Subclasses inherit all accessible fields and methods from the parent class**

B. **Subclasses can define their own methods to override behaviors of the parent method**

C. **Subclasses can define their own unique behaviors and yet share some of the same functionality of the parent class**

D. **None of the above**

7

**Domain :** Working With Java Primitive Data Types and String APIs

Given:

```
String text = "hello";

text.concat("bye");

text.concat("hello");

text.concat("bye");

System.out.println(text);
```

What is the output?

- A. hello
- B. hellobye
- C. hellobyehellobye
- D. bye

8

**Domain :** Using Operators and Decision Constructs

Given:

```
public class Test {

    void switchNumber(long number) {

        switch (number) {

            case 1.0:

                System.out.println("Floating point");

            case 1:

                System.out.println("Integer");

        }

    }


    public static void main(String[] args) {

        Test test = new Test();

        test.switchNumber(1L);

    }

}
```

What is the output of the program?

- A. Floating point
- B. Integer
- C. Nothing
- D. Compilation fails

9

Domain : Working with Java Arrays

Given:

```
int[] array1 = {1, 2, 3};

int[] array2 = {4, 5};

int[][] matrix = new int[3][2];


for (int i = 0; i < array1.length; i++) {

    matrix[i][0] = array1[i];

}

for (int i = 0; i < array2.length; i++) {

    matrix[i][1] = array2[i];

}


for (int[] row : matrix) {

    for (int element : row) {

        System.out.print(element + " ");

    }

    System.out.println();

}
```

What is the output when executing the given code fragment?

A.
```
1 2 3
4 5
```

B.
```
1 4
2 5
3
```

C.
```
1 4
2 5
3 0
```

D. An ArrayIndexOutOfBoundsException is thrown

E. A NullPointerException is thrown

10

**Domain :** Programming Abstractly Through Interfaces

Given:

```
interface Foo {

    int myField = 0;

    void myMethod();

}
```

What of the following statements is correct?

- A. All classes that implement the Foo interface must override the myMethod method
- B. An implementation class of the Foo interface cannot define a field with name myField
- C. All methods in an implementation class of Foo that has name myMethod must be public
- D. The myField field can be changed in an object whose class implements the Foo interface
- E. None of the above

11

**Domain :** Reusing Implementations Through Inheritance

Given:

```
class Foo {
    Foo(String arg) {
        System.out.println("Foo: " + arg);
    }
}

public class Bar extends Foo {
    Bar(String arg) {
        System.out.println("Bar: " + arg);
    }

    public static void main(String[] args) {
        new Bar("test");
    }
}
```

What is the program's output?

- A. Foo: test
- B. Bar: test
- C. Foo: test
  Bar: test
- D. Bar: test
  Foo: test
- E. Compilation fails

12

**Domain :** Creating and Using Methods

Given:

```
public class Test {

    static String text1 = printAndEcho("a");

    static {

        printAndEcho("b");

    }

    static String text2 = printAndEcho("c");


    static String printAndEcho(String text) {

        System.out.print(text);

        return text;

    }


    public static void main(String[] args) { }

}
```

What is the output ?

- A. Nothing
- B. abc
- C. ac
- D. b
- E. Compilation fails

13

Domain : Handling Exceptions

Given:

```
try {
    throw new IOException();
} catch (IOException e) {
    System.out.println("IOException");
} finally {
    System.out.println("finally");
} catch (Exception e) {
    System.out.println("Exception");
}
```

What is the given code's output?

A. IOException
Exception
finally

B. IOException
finally
Exception

C. Exception
IOException
finally

D. Exception
finally
IOException

E. Compilation fails

14

**Domain :** Working With Java Primitive Data Types and String APIs

Given:

```
StringBuilder builder = new StringBuilder("ABCDE")
        .delete(1, 2)
        .deleteCharAt(3);
System.out.println(builder);
```

What is the program's output?

- A. ABCDE
- B. ACD
- C. ADE
- D. DE
- E. Compilation fails

15

**Domain :** Using Operators and Decision Constructs

Given:

```
int x = 1, y = -2;
x -= 3;
y /= 4;
System.out.println(x + " " + y);
```

What is the output of the given code?

- A. -2 0
- B. -2 2
- C. 2 2
- D. 2 4

16



```
Given:

interface Foo {
    abstract void methodA();

    void methodB();

    static void methodC() {
        // a valid body
    }
}

abstract class Bar implements Foo {
    @Override
    public abstract void methodB();

    @Override
    public static void methodC() {
        // a valid body
    }
}
```

Which of the following changes when applied independently makes the *Bar* class compile?

- A. Remove all the @Override annotations
- B. Remove the abstract keyword on methodA in the Foo interface
- C. Remove the static keyword on methodC in the Bar class
- D. Declare a method in the Bar class to override methodA in the Foo interface
- E. Provide a body to methodB in the Bar class to make it a concrete method
- F. Both options B and C
- G. Both options D and E

17

Domain : Reusing Implementations Through Inheritance

Which of the following class declarations can be compiled?

A.
```
public class Foo {
    void myMethod();
}
```

B.
```
public class Foo {
    abstract void myMethod();
}
```

C.
```
public abstract class Foo {
    void myMethod() { }
}
```

D.
```
public abstract class Foo {
    static abstract void myMethod();
}
```

E. None of the above

18

**Domain :** Describing Objects and Classes

Given:

```
public class Test {
    public static void main(String[] args) {
        Test test1 = new Test(); // Line 0
        Test test2 = test1; // Line 1
        Test test3 = new Test(); // Line 2
        test2 = test3; // Line 3
        test3 = test1; // Line 4
        test1 = test2; // Line 5
        test3 = test2; // Line 6
    }
}
```

After which line the object created on line 0 is eligible for garbage collection?

- A. Line 1
- B. Line 2
- C. Line 3
- D. Line 4
- E. Line 5
- F. Line 6

19

**Domain :** Working With Java Primitive Data Types and String APIs

Given:

```
String string = "abcabcabc";
int index1 = string.lastIndexOf("cab");
int index2 = string.lastIndexOf("bca", index1);
System.out.println(index1 + " " + index2);
```

What's the output of the program?

- A. 7 -1
- B. 7 6
- C. 5 4
- D. 5 1
- E. 5 -1
- F. 4 3
- G. 4 1

20

**Domain :** Using Operators and Decision Constructs

Given:

```
public class Test {
    public static void main(String[] args) {
        Test test = new Test();
        String result = test.identifyNumber(1);
        System.out.println(result);
    }

    String identifyNumber(int number) {
        switch (number) {
            default:
                return "Zero";
            case 1:
                return "Positive";
            case -1:
                return "Negative";
        }
    }
}
```

What is the program's output?

- A. Positive
- B. Negative
- C. Zero
- D. Compilation fails

21

Domain : Working with Java Arrays

Given:

```
int[] array1 = {1, 2, 3};

int[] array2 = {1, 3};

int result = Arrays.compare(array1, array2);

System.out.println(result);
```

What is the output?

- A. -1
- B. 1
- C. 2
- D. 3

22

23

**Domain :** Programming Abstractly Through Interfaces

Given:

```
interface Foo {
    String name = "Foo";
}

class Bar implements Foo {
    static String name = "Bar";
}

public class Test extends Bar implements Foo {
    public static void main(String[] args) {
        Foo foo = new Bar();
        System.out.println(foo.name); // Line 1
        System.out.println(name); // Line 2
    }
}
```

What is the program's output?

- A. Foo<br>Foo
- B. Foo<br>Bar
- C. Bar<br>Foo
- D. Bar<br>Bar
- E. Compilation fails on line 1
- F. Compilation fails on line 2

24

**Domain :** Creating a Simple Java Program

Assuming required class files of a program are stored in two directories: *dir1* and *dir2*. Which of the following is a correct way to launch that program? (Assume Windows platform)

A. **java -classpath dir1 dir2 MainClass**

B. **java -classpath dir1:dir2 MainClass**

C. **java -classpath dir1;dir2 MainClass**

D. **java -classpath dir* MainClass**

E. **None of the above**

25



Given:

```
String string = "foo:and:bar";
String[] array = string.split(":", 2);
System.out.println(Arrays.toString(array));
```

What is the output?

- A. `[foo, and, bar]`
- B. `[foo:and, bar]`
- C. `[foo, and:bar]`
- D. `[foo:and:bar]`
- E. Compilation fails

26

Domain : Creating and Using Methods

Given:

```
class Foo {
    Foo() {
        System.out.println("Foo");
    }
}
public class Bar extends Foo {
    Bar() {
        super(); // Line 1
        this("Bar"); // Line 2
    }
    Bar(String arg) { // Line 3
        System.out.println(arg);
    }
    public static void main(String[] args) {
        new Bar();
    }
}
```

What is the program's output?

- A. Foo<br>Bar
- B. Bar<br>Foo
- C. Compilation fails on line 1
- D. Compilation fails on line 2
- E. Compilation fails on line 3

27

Domain : Using Operators and Decision Constructs

Given:

```
public class Test {
    public static void main(String[] args) {
        Test test = new Test();
        test.loop(0);
    }

    void loop(int number) {
        while (number < 5) {
            if (number % 2 == 0) break;
            System.out.print(number + " ");
            number++;
        }
    }
}
```

What is the program's output?

- A. 0
- B. 0 1 2 3 4
- C. 0 2 4
- D. 1
- E. 1 3
- F. Nothing
- G. Compilation fails

Given:

```
class Foo {
    String myField = "Foo";
}

public class Bar extends Foo {
    String myField = "Bar";

    void myMethod() {
        System.out.println(myField);
    }

    public static void main(String[] args) {
        Bar bar = new Bar();
        Foo foo = (Foo) bar; // Line 1
        foo.myMethod(); // Line 2
    }
}
```

What is the program's output?

- A. Foo
- B. Bar
- C. A ClassCastException is thrown
- D. Compilation fails on line 1
- E. Compilation fails on line 2

29

**Domain :** Programming Abstractly Through Interfaces

Given:

```
List list = new ArrayList<>(List.of("A", "B"));
list.addAll(1, List.of("A", "C"));
list.remove("A");
System.out.println(list);
```

What is the output of the given code fragment?

- A. `[A, B, C]`
- B. `[A, A, B, C]`
- C. `[A, C, B]`
- D. `[B, A, C]`
- E. `[B, C]`
- F. Compilation fails

30

**Domain :** Working with Java Arrays

Given:

```
String[] array = new String[5];

Arrays.fill(array, "Hello");

System.out.println(array[2]);
```

What is the output when executing the given code fragment?

A. l

B. Hello

C. null

D. An ArrayIndexOutOfBoundsException is thrown