Time left 0:03:26

**Question 1**

Not yet answered

Marked out of 1.00

How many dimensions does the array reference moreBools allow?

boolean[][] bools[], moreBools;

- a. Two dimensions
- b. One dimension
- c. Three dimensions
- d. None of the above

Clear my choice

**Question 2**

Not yet answered

Marked out of 1.00

What is the result of the following?

```
import java.time.*;
import java.util.*;
class Test {
      public static void main(String[] args)     {
List<String> hex = Arrays.asList("30", "8", "3A", "FF");
Collections.sort(hex);
int x = Collections.binarySearch(hex, "8");
int y = Collections.binarySearch(hex, "3A");
int z = Collections.binarySearch(hex, "4F");
System.out.println(x + " " + y + " " + z);
      }
}
```

- a. None of the above.
- b. The code doesn't compile.
- c. 2 1 –3
- d. 0 1 –2
- e. 0 1 –3
- f. 2 1 –2

Clear my choice

**Question 3**

Not yet answered

Marked out of 1.00

Which is the first line to prevent this code from compiling and running without error?

```
 char[][] ticTacToe = new char[3][3];
ticTacToe[0][0] = 'X';
ticTacToe[1][1] = 'X';
 ticTacToe[2][2] = 'X';
 // r1
 // r2
 System.out.println(ticTacToe.length + " in a row!"); // r3
```

- a. None of these
- b. Line r3
- c. Line r1
- d. Line r2

Clear my choice

**Question 4**

Not yet answered

Marked out of 1.00

What are the names of the methods to do searching and sorting respectively on arrays?

- a. Arrays.search() and Arrays.sort()
- b. Arrays.search() and Arrays.linearSort()
- c. Arrays.binarySearch() and Arrays.linearSort()
- d. Arrays.binarySearch() and Arrays.sort()

Clear my choice

**Question 5**

Not yet answered

Marked out of 1.00

```
interface I{
void m1();                                  //Line-2
}
class A implements I {                          //Line-4
        void m1(){ System.out.println(1);  }          //Line-5
}
class B extends A{
       public void m1(){ System.out.println(2);  }   //Line-8
       public static void main(String[] args){
        A a=new A();
        a.m1();                             //Line-11
        }
}
```

- a. 1
- b. compilation fails at Line-11
- c. 2
- d. compilation fails at Line-5
- e. compilation fails at Line-8
- f. compilation fails at Line-2
- g. compilation fails at Line-4

Clear my choice

**Question 6**

Not yet answered

Marked out of 1.00

```
class A {
        int x=10;
         A(){
                this(20);
                }
    A(int x){
                System.out.println(x);
        }
}
class Test {
                public static void main(String[] args)      {
                        A a=new A(30);
        }
}
```

- a. 10
- b. 20
- c. compilation fails
- d. none of these

- e. 30

Clear my choice

**Question 7**

Not yet answered

Marked out of 1.00

What does the following output?

```
String[] os = new String[] { "Mac", "Linux", "Windows" };
Arrays.sort(os);
System.out.println(Arrays.binarySearch(os, "RedHat"));
```

- a. The output is not defined
- b. -1
- c. -3
- d. -2

Clear my choice

**Question 8**

Not yet answered

Marked out of 1.00

Which line of code causes an ArrayIndexOutOfBoundsException?

```
String[][] matrix = new String[1][2];
matrix[0][0] = "Don't think you are, know you are.";
matrix[0][1] = "I'm trying to free your mind Neo";
matrix[1][0] = "Is all around you ";
// m1
// m2
// m3
matrix[1][1] = "Why oh why didn't I take the BLUE pill?";  // m4
```

- a. m1
- b. m4
- c. m2
- d. m3

Clear my choice

**Question 9**

Not yet answered

Marked out of 1.00

What is the result of the following code? (Choose all that apply)

```
StringBuilder numbers = new StringBuilder("0123456789");
numbers.delete(2, 8);
numbers.append("-").insert(2, "+");
System.out.println(numbers);
```

- [ ] a. An exception is thrown.
- [ ] b. 012+–9
- [ ] c. The code does not compile.
- [ ] d. 012+9–
- [x] e. 01+89–
- [ ] f. 0123456789

**Question 10**

Not yet answered

Marked out of 1.00

What will be the output of the following code?

```
import java.util.function.Predicate;

public class Main {
    public static void main(String[] args) {
        Predicate<String> startsWithA = str -> str.startsWith("A");
        Predicate<String> endsWithX = str -> str.endsWith("X");

        Predicate<String> combined = startsWithA.and(endsWithX);

        System.out.println(combined.test("AppleX"));
    }
}
```

a. Compilation error

b. True

c. False

d. Runtime exception

Clear my choice

**Question 11**

Not yet answered

Marked out of 1.00

Which of the following statements are true?

I. You can always change a method signature from call(String[] arg) to call(String... arg) without causing a compiler error in the calling code.

II. You can always change a method signature from call(String... arg) to call(String[] arg) without causing a compiler error in the existing code.

a. Both I and II

b. II

c. I

d. Neither I nor II

Clear my choice

**Question 12**

Not yet answered

Marked out of 1.00

What is the result of running the following program?

```
1: package fun;
2: public class Sudoku {
3:    static int[][] game = new int[6][6];
4:
5:    public static void main(String[] args) {
6:          game[3][3] = 6;
7:          Object[] obj = game;
8:          obj[3] = "X";
9:          System.out.println(game[3][3]);
10:   }
11: }
```

- a. The code does not compile.
- b. The code compiles but throws a NullPointerException at runtime.
- c. The code compiles but throws a different exception at runtime.
- d. X

Clear my choice

**Question 13**

Not yet answered

Marked out of 1.00

How many objects are created when running the following code?

```
Integer[] lotto = new Integer[4];
lotto[0] = new Integer(1_000_000);
lotto[1] = new Integer(999_999);
```

- a. Four
- b. Three
- c. Two
- d. Five

Clear my choice

**Question 14**

Not yet answered

Marked out of 1.00

Which is the first line to prevent this code from compiling and running without error?

```
char[][] ticTacToe = new char[3][3];
ticTacToe[1][3] = 'X';
ticTacToe[2][2] = 'X';
ticTacToe[3][1] = 'X';
// r1
// r2
System.out.println(ticTacToe.length + " in a row!"); // r3
```

- a. None of the above
- b. Line r3
- c. Line r1
- d. Line r2

Clear my choice

**Question 15**

Not yet answered

Marked out of 1.00

What is the result of running the following as java Copier?

```
package duplicate;
public class Copier {
public static void main(String... original) {
String... copy = original;
System.out.println(copy.length + " " + copy[0]);
}
}
```

- a. The code does not compile.
- b. 0 followed by an exception
- c. 1 followed by an exception
- d. 0

Clear my choice

**Question 16**

Not yet answered

Marked out of 1.00

```
class Test {
 final int x;
      Test(){
            x=10;
             System.out.print(x+" ");
            }
            Test(int y){  x=20;            System.out.print(x+" ");    }

public static void main(String[] args)     {
                  Test t=new Test(30);
      }
}
```

- a. compilation fails
- b. 20
- c. 10
- d. none of these
- e. 0
- f. 30

Clear my choice

**Question 17**

Not yet answered

Marked out of 1.00

Which of the following is NOT a valid lambda expression?

- a. () -> System.out.println("Hello");
- b. int x -> x * 2;
- c. (x, y) -> x + y
- d. (String s) -> { return s.length(); }

Clear my choice

**Question 18**

Not yet answered

Marked out of 1.00

Which of the following is correct about Java 8 lambda expression?

- a. Lambda expression eliminates the need of anonymous class and gives a very simple yet powerful functional programming capability to Java.
- b. None of these.
- c. Both of these.
- d. Lambda expressions are used primarily to define inline implementation of a functional interface.

Clear my choice

**Question 19**

Not yet answered

Marked out of 1.00

What is the correct syntax of a lambda expression for a Predicate<String> that checks if a string is empty?

- a. Predicate<String> p = s -> s.isEmpty();
- b. Predicate<String> p = (s) -> { return s.isEmpty(); }
- c. Predicate<String> p = (String s) -> s.isEmpty();
- d. All of these

Clear my choice

**Question 20**

Not yet answered

Marked out of 1.00

What is the output of the following application?

```
package beach;
import java.util.function.*;
class Tourist {
public Tourist(double distance) {
this.distance = distance;
}
public double distance;
}
public class Lifeguard {
private void saveLife(Predicate<Tourist> canSave, Tourist tourist) {
System.out.print(canSave.test(tourist) ? "Saved" : "Too far");  // y1
}
public final static void main(String... sand) {
new Lifeguard().saveLife(s -> s.distance<4, new Tourist(2));  // y2
}
}
```

- a. The code does not compile because of line y1.
- b. Too far
- c. The code does not compile because of line y2
- d. Saved

Clear my choice