MCQ: 2D Arrays

Haidian Foreign Language Tengfei School – Chris Nielsen

This quiz has 10 questions.

1. A two-dimensional array `myArray` is to be created with the following contents.

```
{ {0, 0, 3},
  {0, 0, 0},
  {7, 0, 0} }
```

Which of the following code segments can be used to correctly create and initialize `myArray`?

I.
```
int[][] myArray = new int[3][3];
myArray[0][2] = 3;
myArray[2][0] = 7;
```

II.
```
int[][] myArray = new int[3][3];
myArray[0][2] = 7;
myArray[2][0] = 3;
```

III.
```
int[][] myArray = {
    {0, 0, 3},
    {0, 0, 0},
    {7, 0, 0} };
```

Ⓐ I only

Ⓑ II only

Ⓒ III only

Ⓓ I and III only

Ⓔ II and III only

Ⓐ Ⓑ Ⓒ **Ⓓ** Ⓔ

2. Consider the following code segment, which is intended to create and initialize the two-dimensional (2D) integer array `num` so that columns with an even index will contain only even integers and columns with an odd index will contain only odd integers.

```
int[][] num = /* missing code */;
```

Which of the following initializer lists could replace `/* missing code */` so that the code segment will work as intended?

Ⓐ `{{0, 1, 2}, {4, 5, 6}, {8, 3, 6}}`

Ⓑ `{{1, 2, 3}, {3, 4, 5}, {5, 6, 7}}`

Ⓒ `{{1, 3, 5}, {2, 4, 6}, {3, 5, 7}}`

Ⓓ `{{2, 1, 4}, {5, 2, 3}, {2, 7, 6}}`

Ⓔ `{{2, 4, 6}, {1, 3, 5}, {6, 4, 2}}`

Ⓐ Ⓑ Ⓒ Ⓓ Ⓔ

3. Consider the following code segment, which is intended to display `"cat"`.

```
String[][] kb = {
  {"q", "w", "e", "r", "t"},
  {"a", "s", "d", "f", "g"},
  {"z", "x", "c", "v", "b"} };
System.out.println(
    /* missing expression */);
```

Which of the following can replace `/* missing expression */` so that the code segment works as intended?

Ⓐ `kb[12] + kb[5] + kb[4]`

Ⓑ `kb[13] + kb[6] + kb[5]`

Ⓒ `kb[2][2] + kb[1][0] + kb[0][4]`

Ⓓ `kb[2][2] + kb[0][1] + kb[4][0]`

Ⓔ `kb[3][3] + kb[2][1] + kb[1][5]`

Ⓐ Ⓑ **Ⓒ** Ⓓ Ⓔ

4. Consider the following code segment, where `twoD` is a two-dimensional (2D) `String` array. The code segment is intended to display `"JAVA"`.

```
System.out.print(twoD[2][1]);
System.out.print(twoD[3][2]);
System.out.print(twoD[1][1]);
```

Which of the following code segments properly declares and initializes `twoD` so that the code segment works as intended?

Ⓐ
```
String[][] twoD = {
    {"V", "AV", "J"},
    {"JA", "VA", "A"},
    {"JA", "J", "JAV"},
    {"AV", "V", "A"}};
```

Ⓑ
```
String[][] twoD = {
    {"VA", "J", "A", "V"},
    {"J", "A", "V", "A"},
    {"AV", "A", "JA", "V"}};
```

Ⓒ
```
String[][] twoD = {
    {"VA", "J", "V", "JA"},
    {"J", "JA", "A", "VA"},
    {"J", "VA", "A", "V"}};
```

Ⓓ
```
String[][] twoD = {
    {"A", "VA", "J", "V"},
    {"VA", "A", "JA", "V"},
    {"VA", "J", "A", "V"}};
```

Ⓔ
```
String[][] twoD = {
    {"A", "V"},
    {"VA", "J"},
    {"J", "A"},
    {"A", "AV"}};
```

Ⓐ Ⓑ Ⓒ Ⓓ Ⓔ

MCQ: 2D Arrays

5. Consider the following code segment.

```
int[][] multi = new int[4][4];
for (int rows=0; rows<4; rows++) {
    for (int cols=0; cols<4; cols++) {
        if (cols==0) {
            multi[rows][cols] = 0;
        } else if (cols==1) {
            multi[rows][cols] = 1;
        } else if (cols==2) {
            multi[rows][cols] = 2;
        }
        if ((rows%2==0) && (cols%2==0)){
            if ((rows>=2) && (cols<=2)) {
                multi[rows][cols] = 9;
            }
        }
    }
}
```

As a result of executing the code segment, how many elements in the two-dimensional (2D) array multi will store the value `9`?

Ⓐ `0`

Ⓑ `1`

Ⓒ `2`

Ⓓ `4`

Ⓔ `6`

Ⓐ Ⓑ **Ⓒ** Ⓓ Ⓔ

---

6. Consider the following code segment.

```
int[][] mat = {
  {10, 15, 20, 25},
  {30, 35, 40, 45},
  {50, 55, 60, 65}};
for (int[] row : mat) {
    for (int j=0; j<row.length; j+=2) {
        System.out.print(row[j] + " ");
    }
    System.out.println();
}
```

What, if anything, is printed as a result of executing the code segment?

Ⓐ
```
10 15 20 25
50 55 60 65
```

Ⓑ
```
10 20
30 40
50 60
```

Ⓒ
```
10 15 20 35
30 35 40 45
50 55 60 65
```

Ⓓ Nothing is printed, because an `ArrayIndexOutOfBoundsException` is thrown.

Ⓔ Nothing is printed, because it is not possible to use an enhanced for loop on a two-dimensional array.

Ⓐ **Ⓑ** Ⓒ Ⓓ Ⓔ

---

7. Consider the following code segment.

```
int[][] arr = {{3, 2, 1}, {4, 3, 5}};

for (int row=0; row<arr.length; row++) {
  for (int c=0; c<arr[row].length; c++) {
    if (c > 0) {
      if (arr[row][c]>=arr[row][c-1]){
        System.out.println("Condition one");
      }
    }
    if (arr[row][c]%2==0) {
      System.out.println("Condition two");
    }
  }
}
```

As a result of executing the code segment, how many times are `"Condition one"` and `"Condition two"` printed?

Ⓐ `"Condition one"` is printed twice, and `"Condition two"` is printed twice.

Ⓑ `"Condition one"` is printed twice, and `"Condition two"` is printed once.

Ⓒ `"Condition one"` is printed once, and `"Condition two"` is printed twice.

Ⓓ `"Condition one"` is printed once, and `"Condition two"` is printed once.

Ⓔ `"Condition one"` is never printed, and `"Condition two"` is printed once.

Ⓐ Ⓑ **Ⓒ** Ⓓ Ⓔ

---

8. Consider the following code segment.

```
int[][] arr = {{1, 3, 4}, {4, 5, 3}};
int max = arr[0][0];
for (int row=0; row<arr.length; row++) {
   for (int c=0; c<arr[row].length; c++) {
      int temp = arr[row][c];
      if (temp % 2 == 0) {
        arr[row][c] = temp+1;    // line 11
      }
      if (temp > max) {
        max = temp;
      }
   }
}
System.out.println(max);
```

How many times will the statement in ***`line 11`*** be executed as a result of executing the code segment?

Ⓐ 2

Ⓑ 3

Ⓒ 4

Ⓓ 5

Ⓔ 6

Ⓐ Ⓑ Ⓒ Ⓓ Ⓔ

MCQ: 2D Arrays

9. Consider the following method, sumRows, which is intended to traverse all the rows in the two-dimensional (2D) integer array `num` and print the sum of all the elements in each row.

```
public static void sumRows(int[][] num) {
   for (int[] r : num) {
      int sum = 0;
      for (int j=0; j<num.length; j++) {
         sum += r[j];
      }
      System.out.print(sum + " ");
   }
}
```

For example, if num contains:

```
{{3, 5}, {6, 8}}
```

then `sumRows(num)` should print `"8 14 "`.

The method does not always work as intended. For which of the following two-dimensional array input values does `sumRows` NOT work as intended?

Ⓐ `{{0, 1}, {2, 3}}`

Ⓑ `{{10, -18}, {48, 17}}`

Ⓒ `{{-5, 2, 0}, {4, 11, 0}}`

Ⓓ `{{4, 1, 7}, {-10, -11, -12}}`

Ⓔ `{{1, 2, 3}, {4, 5, 6}, {7, 8, 9}}`

Ⓐ Ⓑ Ⓒ **Ⓓ** Ⓔ

10. Consider the following code segment, where `num` is a properly declared and initialized integer variable. The following code segment is intended to set `foundRow` and `foundCol` to the row and column indexes of an array element containing `num`. The code segment does not work as intended.

```
int[][] arr = {
  {10, 11, 12, 13},
  {22, 24, 26, 28},
  {15, 16, 17, 18},
  {40, 41, 42, 43}};
int foundRow = -1;
int foundCol = -1;
for (int j=0; j<arr.length; j++) {
   for (int k=1; k<arr[0].length; k++) {
      if (arr[j][k] == num) {
         foundRow = j;
         foundCol = k;
      }
   }
}
```

Which of the following values for `num` can be used as a test case to show that the code segment does not work as intended?

Ⓐ `12`

Ⓑ `15`

Ⓒ `24`

Ⓓ `41`

Ⓔ `43`

Ⓐ **Ⓑ** Ⓒ Ⓓ Ⓔ