Unit 8 AP Computer Science A Practice Exam
Two-Dimensional Arrays

Section I – Multiple Choice
Optional Time – 20 minutes
10 Questions

1) Which of the following are true about two-dimensional arrays?

   I. All of the elements inside of an array must be of the same type or related types.
   II. When creating an array with the keyword *new*, integers are initialized to *0*.
   III. The first row of an array is located at the index of 1.

   (A) I only
   (B) II only
   (C) I and II
   (D) II and III
   (E) I, II, and III

2) Which of the following describes how two-dimensional arrays are configured?

   (A) Row-major order
   (B) Column-major order
   (C) Row-minor order
   (D) Column-minor order

3) Which of the following is the syntax to get an element out of a two-dimensional array at row `x` and column `y`?

   (A) `array[x[y]]`
   (B) `array[y][x]`
   (C) `array[x][y]`
   (D) `array.get(x,y)`

4) Which of the following is the correct declaration of a String two-dimensional array field meant to store sensitive data?

   (A) `String[][] data;`
   (B) `private String[][] data;`
   (C) `public String[][] data;`
   (D) `static String[][] data;`

5) Which of the following is the correct initialization of a two-dimensional array that should have 3 rows and 4 columns?

   (A) `String[][] data = new String[3][4]`
   (B) `String[][] data = new String[4][3]`
   (C) `String[3][4] data = new String[3][4]`
   (D) `String[4][3] data = new String[4][3]`

6) Which of the following is the correct code to store the number of columns in a two-dimensional array `a` into the `nC` variable?

   (A) `int nC = a.length;`
   (B) `int nC = a[0].length;`
   (C) `int nC = a.size();`
   (D) `int nC = a[0].size();`

Answer questions 7 and 8 based on the code below.

```
int[][] array = new int[4][4];

for(int i = 0; i < array.length; i++) {
   for(int j = array[0].length - 1; j >= 0; j--) {

      System.out.println(array[i][j]);
   }
}
```

7) Which of the following is accomplished by the code above?

(A) The elements of the two-dimensional array are printed, starting at the top left and ending in the bottom right
(B) The elements of the two-dimensional array are printed, starting at the bottom left and ending in the top right
(C) The elements of the two-dimensional array are printed, starting at the top right and ending in the bottom left
(D) The elements of the two-dimensional array are printed, starting at the bottom right and ending in the top left

8) What output will result from the code above?

(A) All of the output lines will be `1`.
(B) All of the output lines will be `nil`.
(C) All of the output lines will be `0`.
(D) All of the output lines will be `4`.
(E) There would be an error because the values inside the two-dimensional array were never set.

9) Which of the following is a valid way of traversing a two-dimensional array from the top left to the bottom right?

I.
```
int[][] array = new int[3][4];

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

      System.out.println(array[i][j]);
   }
}
```

II.
```
int[][] array = new int[3][4];

for(int[] i : array) {
   for(int j = 0; j < i.length; j++) {

      System.out.println(i[j]);
   }
}
```

III.
```
int[][] array = new int[3][4];

for(int[] i : array) {
   for(int j : i) {

      System.out.println(j);
   }
}
```

(A) I only
(B) II only
(C) III only
(D) II and III
(E) I, II, and III

10) Which initialization of the array variable would result in a integer two-dimensional array the same dimensions as the one in the code below?

```
array = {{1,2,3},{4,5,6}};
```

(A) `array = new int[3][6];`
(B) `array = new int[2][3];`
(C) `array = new int[3][2];`
(D) `array = new int[6][3];`

END OF SECTION I

Section II – Free Response Section

Optional Time – 15 minutes

1 Question

1. This question involves the representation of a battlefield by the following `Battlefield` class.

```
public class Battlefield
{
   /** Data fields. */
   private boolean[][] battlefieldGrid;

   /** Returns true if a location on the battlefield has a land mine, returns false
    *  otherwise.
    */
   public boolean hasMine(int row, int col)
   { /* to be implemented in part (a) */ }

   /** Returns true if a soldier can walk across the battlefield at a given row
    *  without hitting a land mine, returns false otherwise.
    */
   public boolean canSafelyCross(int rowToCross)
   { /* to be implemented in part (b) */ }


   // There may be instance variables, constructors, and methods not shown.

}
```

(a) Write the `Battlefield` method `hasMine`. This method will return `true` if there is a land mine at the row and column position in the `battlefieldGrid` field, and will return `false` otherwise.

The following is a representation of the `battlefieldGrid` array:

`true` – Land mine exists in area
`false` – Land mine does not exist in area

X	0	1	2	3	4
0	true	true	false	false	false
1	false	true	false	false	false
2	false	false	false	false	false
3	false	false	false	true	false
4	false	true	false	false	false

Class information for this question

```
public class Battlefield

private int[][] battlefieldGrid;

public boolean hasMine(int row, int col);
public boolean canSafelyCross(int row);
```

Complete the `hasMine` method below.

```
/** Returns true if a location on the battlefield has a land mine, returns false
*  otherwise.
*/
public boolean hasMine(int row, int col)
```

(b) Write the `Battlefield` method `canSafelyCross`. This method will return `true` if a soldier can safely cross the battlefield, represented by the `battlefieldGrid` field, without hitting any mines, and will return `false` otherwise.

NOTE: You must use the `hasMine` method appropriately to receive full credit.

For example:

X	0	1	2	3	4
0	`true`	`true`	`false`	`false`	`false`
1	`false`	`true`	`false`	`false`	`false`
2	`false`	`false`	`false`	`false`	`false`
3	`false`	`false`	`false`	`true`	`false`
4	`false`	`true`	`false`	`false`	`false`

In the example, row 2 is the only one that would return `true`, as it is the only row without any land mines.

Class information for this question

```
public class Battlefield

private int[][] battlefieldGrid;

public boolean hasMine(int row, int col);
public boolean canSafelyCross(int row);
```

Complete the `canSafelyCross` method below.

```
/** Returns true if a soldier can walk across the battlefield at a given row
 *  without hitting a land mine, returns false otherwise.
 */
public boolean canSafelyCross(int rowToCross)
```

END OF SECTION II