# CS 249: Assignment 4

Single and Multidimensional Arrays

## Theory Questions (24%)

1. (2%) Declare AND initialize an single-dimensional array of **5 doubles** with the name “list”. You do not need to set the values of the doubles.

2. (2%) If we had declared but NOT initialized “list”, what value would it have?

3. (2%) Write code **using a for loop** to set each value of “list” to 1. **Do NOT use the number 5 directly as the length!**

4. (2%) Write code **using a for-each loop** to print each value of “list” on each line.

5. (2%) I **CANNOT** use a for-each loop to modify values in the array.
   (a) True
   (b) False
6. (2%) I **CAN** use a for-each loop if I have to traverse the array in reverse.
   (a) True
   (b) False

7. (2%) Given the array “list”, “list” is a reference variable, because an array is implemented as a class.

   (a) True

   (b) False

8. (2%) Given the code below, what will the values in the array “powers” be?

```
public static void main(String [] args) {
        int [] powers = { 1, 1, 1, 1 };
        fillPowers(powers);
        System.out.println(Arrays.toString(powers));
}

public static void fillPowers(int [] powers) {
        powers[0] = 1;
        for(int i = 1; i < powers.length; i++) {
                powers[i] = powers[i-1]*2;
        }
}
```

9. (2%) If I want to copy the contents of “list2” into “list1”, should I use the code below? If not, why not?

```
list1 = list2;
```

10. (2%) If I call my Java program “TestArgs” on the command line as shown below, what would be the contents of **args[2]**?

```
java TestArgs Totoro "Spirited Away" "Howl's Moving Castle"
```

11. (2%) Declare and initialize a 2D doubles array of **3 rows** and **4 columns** named “M”. You do not need to set the values in the array.

12. (2%) Given the 2D array “M”, write code to store the length of the first row into an int variable “firstLen”. **Do NOT use the number 4 directly as the length!**

# Programming Assignments (76%)

**Where appropriate, use the Pseudocode Programming Process to implement!**

**For this assignment, use a SEPARATE Java file for each requirement (not sub-requirements)! Name each Java file "Assign4_N.java", where N is the requirement number.**

You can use the checkboxes to track whether you've met each requirement.

| # | *Questions* | |
|---|---|---|
| **1** | **Implement 7.25 from the book, BUT have it return the roots as an array (instead of passed in as a parameter). (If no roots, array has zero length.)** | |
| | The method will therefore have the following declaration:<br>**public static double[] solveQuadratic(double [] eqn)** | |
| | REMEMBER: Use epsilon check for double equality with zero!!! | |
| | INPUT: 1.0 3 1<br>OUTPUT: Two roots: -0.381966 and -2.61803 | |
| | INPUT: 1 2.0 1<br>OUTPUT: One root: -1 | |
| | INPUT: 1 2 3<br>OUTPUT: No real roots | |
| **2** | **You will implement an extremely bare-bones version of the game NetHack.** | |
| | Create two constants for the map rows and columns. Set them to 10 rows and 20 columns. | |
| | Create a 2D char array to hold the map. It will be filled with the dot character '.' | |
| | Create another 2D char array to hold the display buffer (same size as the map). Think of this as your canvas that will draw things to, and then you will print that canvas to the screen. | |
| | Create a 1D int array to hold the x and y coordinates of the player. | |
| | (Do-while) Do the following: | |
| | Copy the map into the display buffer. | |
| | Draw the player by putting an '@' sign at display[playerCoords[1]][playerCoords[0]]. | |

| | | |
|---|---|---|
| | “Render” the display by printing the characters in the display buffer to the screen. | |
| | Ask the user for input and grab the first character. | |
| | Move the player coordinates depending on the first character:<br>’a’ = move left<br>’d’ = move right<br>’s’ = move down<br>’w’ = move up<br>**IF THE MOVEMENT WOULD PUT THE PLAYER OUTSIDE THE BOUNDS OF THE MAP/ARRAY, DO NOTHING.** | |
| | ...while the input character is NOT ’q’. | |
| | Initial output would look like this:<br>@.....<br>.....<br>.....<br>.....<br>.....<br>.....<br>.....<br>.....<br>.....<br>.....<br>Enter action: | |

## Submission

You will submit the following items as a *.tar or *.zip file:

- A plaintext, Word doc, or PDF with your answers to any theory questions
- Your .java file(s)

Submit this tar/zip file on Blackboard under the appropriate assignment.

Do NOT submit:

- Your .class file(s)
- Your project files

## Grading

Below is a list of SOME of the grading penalties:

- Sloppy or poor coding style
- Bad coding design principles
- Code that does not compile
- Code that crashes, does not run, or takes a VERY long time to complete
- Using code from ANY source other than the course materials
- Collaboration on code of ANY kind; this is an INDIVIDUAL PROJECT
- Sharing code with other people in this class or using code from this or any other related class
- Output that is incorrect
- Algorithms/implementations that are incorrect
- Submitting improper files
- Failing to submit ALL required files