CIS351-ArrayPlay Lab

Submission Instructions

Submit the completed **ArrayPlay.java** file in Blackboard.

Objectives

Students will be able to:

- Create an array
- Initialize the array elements to an appropriate value
- Use an array to track a sequence of values
- Display the contents of the array

Key Terms

array.length
: an attribute of the array that is the number of elements in the array

initializer list
: a list of values used to instantiate and initialize an array to the values of the items in the list

ArrayIndexOutOfBoundsException
: an exception that is raised when a program tries to access an element using a subscript that either is negative or is equal to or larger than the number of elements in the array

Download Materials

- Download **Die.java** which you will need for this lab.

Additional Help

Fee free to watch the posted video on how Array works in Java.

Part 1 - Getting Started

You will write this program from scratch.

1. Create a new folder for this lab. Download a copy of Die.java to that folder.

2. Create a new Java file named **ArrayPlay.java**.

3. ArrayPlay will contain a single main method. For each section below, you will add code to the program, not replace prior code. When you are done, this program will provide you with a reference for working with arrays including initializing, printing, and manipulating entries.

Part 2 - Practice sequentially accessing an array

- 1. In the main method, declare and initialize a single array of integers that is 6 elements long.
- 2. **INITIALIZE:** First initialize each of the array elements to the value, -1. (use an appropriate loop to do so).
- 3. **PRINT:** Now write another loop to print each element of the array on a separate line as shown:

```
array[0] = -1
array[1] = -1
...
```

- 4. **STOP:** Now it is time to test if everything works so far. So, compile and execute to TEST IT!
- 5. **UPDATE:** Now write another loop to change the value of each element of the array to its subscript. For example, myArray[3] should hold the value 3.
- 6. **OPTIONAL PRINT:** If you wish, now you may print each element of the array (the one you updated in STEP 5) on a separate line. This will allow you to see the contents of the array. You may re-use the code from step 3.
- 6. **PUT ZERO:** Finally, assign 0 to all array positions. At this point, your array is empty, meaning all the positions has zero. **You can now use this array in Part 3.**

Part 3 - Randomly accessing an array

- Create a new `Die` object. Here is how to do this:
  `Die dice = new Die();`
  NOTE: make sure Die.java and arrayPlay.java is in the same Folder.
- Create a loop that will iterate 100 times. Withing the loop body, write instruction that will
  - Roll the die and the save the value of the rolled face in a variable. Here is how you can accomplish that
    `int face = dice.roll();` // roll() method rolls the die and returns the face value which is saved in the face variable.
  - Take the result of the roll, map it to the location/subscript/index in the array, and increment the value stored in that location/subscript/index. The die values will be in the range 1 - 6, your array subscripts are 0 - 5. Think about how you will map the die values to subscripts.

    At this point, the value stored at each array location will be a count of the number of times that roll is made. Hence, if 1 comes up three times, array[0] should have the value 3. If, 4 compes up eight times, array[3] will have value 8.
- After the loop finishes, print the results of the rolls as they correspond to die faces. In other words, how many times was a 1 rolled?, was a 2 rolled? etc.

  For example, if the array holds {20, 17, 19, 15, 12, 17} you would output:

```
1 was rolled 20 times.
2 was rolled 17 times.
...
```

Part 4 - Working with two arrays

- You will need two double arrays. Declare these and then....
  - initialize **one** of the arrays to 10 double values (of your choice).
  - Instantiate the other array, but do not initialize the individual contents.
  - Print the contents of both arrays. Print them side by side on your screen. Label this output **Before**
  - Copy the contents of the first array to the second array and again print their contents side by side. Label this output **After**
  - **NOTE: At this point, both of the Array should display same output.**

- Finally, change the content of one element of the first array and a different element of the second array. (Basically, assign a different number than before).
- Again, print the contents of the two arrays side by side. Label this output*After Change*.
- **NOTE: At this point, the contents of the two arrays will be indeed different, at least in two location. If not, try to figure out why they are not and make the appropriate corrections.**

Grading Criteria

Total points: 100 points

General Programming Correctness - 10pt
Part 2 - 30pt
Part 3 - 30pt
Part 4 - 30pt

Farzana Rahman / frahman@syr.edu