Objectives:
Practice using arrays to store and process values of different types.

Part I: Preparation
Submit through Blackboard by 8:00am the morning of Lab.

Simple array example:
```java
public class ArrayDemo {  
    public static void main(String[] args) {  
        int[] anArray; // declares an array of integers

        anArray = new int[10]; // allocates memory for 10 integers

        anArray[0] = 100; // initialize first element
        anArray[1] = 200; // initialize second element
        anArray[2] = 300; // and so forth
        anArray[3] = 400;
        anArray[4] = 500;
        anArray[5] = 600;
        anArray[6] = 700;
        anArray[7] = 800;
        anArray[8] = 900;
        anArray[9] = 1000;

        System.out.println("Element at index 0: " + anArray[0]);
        System.out.println("Element at index 1: " + anArray[1]);
        System.out.println("Element at index 2: " + anArray[2]);
        System.out.println("Element at index 3: " + anArray[3]);
        System.out.println("Element at index 4: " + anArray[4]);
        System.out.println("Element at index 5: " + anArray[5]);
        System.out.println("Element at index 6: " + anArray[6]);
        System.out.println("Element at index 7: " + anArray[7]);
        System.out.println("Element at index 8: " + anArray[8]);
        System.out.println("Element at index 9: " + anArray[9]);
    }
}
```

- Type in this program, compile, and run.

  - **Rewrite B:** (yes, it is recommended you start with B, not with A!) Modify the program so that it prints out the array values using a for-loop, instead of the repetitive code, above (i.e., rewrite the block of printing statements marked “B”).

  - **Rewrite A:** Modify the program so that it also initializes the array values using a for-loop, i.e., rewrite the block of assignment statements marked “A”.

  **NOTE:** Be sure you have two separate loops, DO NOT combine B with A.

- Submit your modified code for ArrayDemo though Blackboard.
Part A: Review & fine tune your work for preparation (ArrayDemo)

• Draw a picture of the array (be sure to label the elements with their indices).

• Compare your solution with your partner’s. Approve or correct each other’s solution, then sign each other’s worksheet.

  Lab partner’s signature (indicates approval): ______________________________

• Modify the code so that it uses a constant:
  ```java
  final int SIZE = 10;
  ```
  Use SIZE throughout your code instead of the number 10 (i.e., in the array declaration and both loops)

• What happens if you do not initialize the array’s values? (Comment out the initialization loop that sets the values to 100, 200, etc. Do you get an error?)
  
  Answer: ____________________________________________________________________________________

Part B: Creating arrays of different types

1. An array of double: Make a new version of your program from Part I and name it ArrayDouble.java. This program should create an array of 100 values of type double, set to random values in the range 0….1.

  • What happens if you do not initialize the array’s values?
  
  Answer: ____________________________________________________________________________________

2. An array of boolean: Implement a new version of your program ArrayBoolean.java that creates instead an array of 100 values of type boolean. The values should be set to alternating true/false, i.e., anArray[0] = true; anArray[1] = false, etc. (be sure to use a loop here too).

  • What happens if you do not initialize the array’s values?
  
  Answer: ____________________________________________________________________________________
3. **An array of char**: Implement a new version of your program `ArrayChar.java` that creates instead an array of 26 values of type `char`. The values should be set to the letter ‘a’ ... ‘z’.

- What happens if you do not initialize the array’s values?

  **Answer:**

4. **Reading values from the user and storing them in an array:**
Starting from `ArrayDouble.java`, implement a new version of your program `ArrayInput.java`, that uses Scanner to input values from the user. (Test it with smaller arrays, say with 5 entries, so you don’t need to type so much.)

5. **An array of String**: Implement a new version of your program `ArrayString.java` that creates an array of 8 values of type `String`. The values should be set to strings of your choice, with the purpose of implementing a Magic 8 Ball. For example, one of the strings might say “Signs point to YES”. Note that you do NOT want to use a loop to set the array values here, but they can still be displayed using a loop.

Your finished program should set up the array, display it, and proceed to input a question from the user to which it should produce a random answer from the array.

  **Hint:** Generate a random number and use it to index into the array (does that tell you what range of random values you need to generate?)

- What happens if you do not initialize the array’s values?

  **Answer:**

**Part C: Expand your knowledge of arrays and all things Java!**

The array exercise example is from the Java online tutorials:

http://docs.oracle.com/javase/tutorial/java/nutsandbolts/arrays.html

- Read the tutorial on arrays. This is one of many online tutorials that are available on Java programming. The Oracle tutorials is a great way to brush up on a topic or to learn more about specific topics of interest to you.

- Make a note here about something that you learned

  **Answer:**
Lab 10 Comments

Comments on this lab, please:

What was the most valuable thing you learned in this lab?

What did you like best about this lab?

Was there any particular problem?

Do you have any suggestions for improving this lab as an effective learning experience?