**CS 110 Programming Fundamentals I Lab 8: Arrays**

**upload to Canvas Computer Science**

The goal of this lab is to give you practice in using an array. You will create an array, into which you'll place randomly generated integers representing the exam scores for students in a class. You'll then access the array's values, and print the array's contents, as well the average and the maximum scores. Two sample invocations of the completed program are shown in Figure 1.

How many students are in the class? 14

... generating scores ... Student 1 has a score of 83

Student 2 has a score of 19

Student 3 has a score of 79

Student 4 has a score of 79

Student 5 has a score of 96

Student 6 has a score of 90

Student 7 has a score of 5

Student 8 has a score of 83

Student 9 has a score of 31

Student 10 has a score of 49

Student 11 has a score of 30

Student 12 has a score of 72

Student 13 has a score of 42

Student 14 has a score of 72

The average score is 59.285714285714285

The maximum score of 96 was attained by student 5

How many students are in the class? 9

... generating scores ... Student 1 has a score of 46

Student 2 has a score of 14

Student 3 has a score of 89

Student 4 has a score of 68

Student 5 has a score of 59

Student 6 has a score of 77

Student 7 has a score of 74

Student 8 has a score of 99

Student 9 has a score of 89

The average score is 68.33333333333333

The maximum score of 99 was attained by student 8

**Figure 1**: Two sample runs of the program *ArrayOfExamScores.java*

**I. Download skeleton file from Website**

Retrieve from the course webpage, the file *ArrayOfExamScores.java*. It is the skeleton of a java file for completing this lab. The comments in the skeleton file accompany the steps outlined below.

**II. Retrieve an integer from the keyboard**

Write code to retrieve from the keyboard, an integer, and save it into the variable numStudentsInClass. You've done this many times before. If you've forgotten how, then the code on page 85 of the textbook performs exactly this task (except the variable should be named "numStudentsInClass" instead of "number"). Be sure to ask the user, using a System.out.print statement, how many students are in the class.

**III. Creating an array**

Write code in which you create a reference variable, named arrayOfTestScores, of type int[]. Create a new array that has as many integers as the value that is saved in the variable numStudentsInClass. You can do this using two statements. Either use two statements, or use a single statement that achieves the same task. For example, if you wanted to create an array of 15 integers:

int numEntries = 15;

int[] arrayName;

arrayName = new int[numEntries];

or

int numEntries = 15;

int[] arrayName = new int[numEntries];

As another example, if you wanted to declare an array called arrayOfTestScores, that holds 6 integers, you would write:

int[] arrayOfTestScores = new int[6];

That would create an array with 6 elements, and each element would be initialized to the value of 0:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 0 | 0 | 0 | 0 | 0 | 0 |

**IV. Placing randomly generated test scores into your array**

Into each of the positions in the array named arrayOfTestScores, place a randomly generated integer between

0 and 100. To do this, you need to access each entry of the array, which has been initialized to hold all zeros. You can complete this task by using a for loop. For example if you had an array called myArray, that had 5 entries that looked like the following, where the index below a box indicates the index number of the array (and is here the number of the student) :

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 7 | 6 | 9 | 5 | 8 |

you could iterate over the array using the following for loop, and place the number 12 into each position:

for (int i = 0; i < 5; i++){

myArray[i] = 12;

}

In this case, the number of entries that are saved in your array is not a constant number, but instead is based on the input supplied by the user. You can use either the length field of the array (as was shown in lecture), or use the variable numStudentsInClass as the bound condition of the for loop.

In order to place a random number between 0 and 100 into each index of your array, use the Random class to create a random number generator. Use the nextInt method from the Random class at each iteration of the for

loop, to place a value between 0 and 100 into each index.

**V. Printing out the values in your array**

At this point, you have an array, with randomly generated numbers ranging from 0 to 100. For this step, again iterate through the array (see Step 3), and at each iteration of the “for” loop, print to the screen the student number (the array index) and that student's randomly generated test score. To do this, issue the following System.out.println statement during each iteration of the for loop:

System.out.println("Student " + (i+1)

+ " has a score of "

+ arrayOfTestScores[i]);

Because the first index of any array is at position 0, you are using "(i+1)" to print to the screen the array index, but incremented by 1, so that array index 0 corresponds to person 1, array index 1 corresponds to person 2, array index 3 corresponds to person 3, etc.

**VI. Calculating the average exam score**

Now, you want to calculate the average score of the randomly generated exam scores that you have placed into your array. To do this, you'll need to create a variable, of type integer, and name it runningTotal, and initialize it to the value 0. Use the variable runningTotal to keep the running sum total of the scores as you iterate through your array using the same for loop as in step 3. At each iteration of the for loop, increment the value of runningTotal by the value of the integer at position i of the array arrayOfTestScores.

To this this, the code that you'd write inside the body of your for loop is the following:

runningTotal += arrayOfTestScores[i];

**VII. Calculate and output the average exam score**

Print to the screen, the average of the exam scores in your array. Remember that the variables runningTotal and numStudentsInClass are integers, and if you perform integer division, then you'll loose the decimal values. To retain the decimal place, when you perform the calculation for the average exam score, cast either runningTotal or numStudentsInClass as a double. Converting (or casting) from one primitive data type to another was explained in class, and is discussed in section 2.7 of the textbook. Page 67 gives you sample code that you can use to make sure that division that you perform with integers retains the decimal value. For

example, if you have two variables of type integer, var1 and var2, and you want to performed division and not lose the decimal point, you'd write:

double result = (double)var1/var2;

After you calculate the average score, use a System.out.println statement to print that value to the screen.

**VIII. Calcluate the maximum exam score**

Print to the screen, the maximum exam score in your array, as well as the person (index) who has the highest score. To do this, first define two variables of type int. Name one of those variables maximumScore, and the other maximumScoreStudent. Then, again iterate through the elements of your array using the for-loop that has been given to you. In the body of the for loop, which will be executed during each iteration of the for loop, check if the value at position i is greater than the currently seen largest value stored in the variable maximumScore. If yes, then update the variables maximumScore and maximumScoreStudent. The code that you want to write into the body of your for loop is the following:

if (arrayOfTestScores[i] > maximumScore){ maximumScore = arrayOfTestScores[i]; maximumScoreStudent = i+1;

}

**IX. Printing out the value of the maximum score**

After you've determined the maximumScore and maximumScoreStudent, use a System.out.println statement to print the maximum score to the screen.

**What to hand in**

Make sure to upload the following file to Canvas:

*“ArrayOfExamScores.java”*

*“ArrayOfExamScores output.docx”*

Be sure that each .java file is commented, and be sure to include your name, date, honor code at the top of each file. Code must be indented, so that it is easy to read. Finally, make sure that you have given your variables good names.

**Rubric**

|  |  |
| --- | --- |
| **File / task** | **Points** |
| No name, and or date and or honor code | -100 |
| *ArrayOfExamScores.java* compiles and runs to completion, included output file | 50 |
| Program accurately calculates random scores | 10 |
| Program accurately calculates average and maximum of scores | 10 |
| Variable names are adequate and descriptive in all java files | 10 |
| Program Correctly formatted, fully commented, good comments, plenty of blank lines | 20 |
| Total | 100 |