

Chapter 4 Lab Loops and Files

Lab Objectives

- Be able to convert an algorithm using control structures into Java
- Be able to write a while loop
- Be able to write an do-while loop
- Be able to write a for loop
- Be able to use the Random class to generate random numbers.
- Be able to use file streams for I/O
- Be able to write a loop that reads until end of file
- Be able to implement an accumulator and a counter

Introduction

This is a simulation of rolling dice. Actual results approach theory only when the sample size is large. So we will need to repeat rolling the dice a large number of times (we will use 10,000). The theoretical probability of rolling doubles of a specific number is 1 out of 36 or approximately 278 out of 10,000 times that you roll the pair of dice. Since this is a simulation, the numbers will vary a little each time you run it.

Check out how to use the random number generator (introduced in section 4.11 of the text) to get a number between 1 and 6 to create the simulation.

We will continue to use control structures that we have already learned, while exploring control structures used for repetition. We shall also continue our work with algorithms, translating a given algorithm to java in order to complete our program. We will start with a while loop, then use the same program, changing the while loop to a do-while loop, and then a for loop.

We will be introduced to file input and output. We will read a file, line by line, converting each line into a number. We will then use the numbers to calculate the mean and standard deviation.

First we will learn how to use file output to get results printed to a file. Next we will use file input to read the numbers from a file and calculate the mean. Finally, we will see that when the file is closed, and then reopened, we will start reading from the top of the file again so that we can calculate the standard deviation.

Task #1 While loop

1. Copy the file DiceSimulation.java (see code listing 4.1) from the Student CD or as directed by your instructor. DiceSimulation.java is incomplete. Since there is

a large part of the program missing, the output will be incorrect if you run DiceSimulation.java.

2. I have declared all the variables. You need to add code to simulate rolling the dice and keeping track of the doubles. Convert the algorithm below to Java and place it in the main method after the variable declarations, but before the output statements. You will be using several control structures: a **while** loop and an if-else-if statement nested inside another if statement. Use the indenting of the algorithm to help you decide what is included in the loop, what is included in the if statement, and what is included in the nested if-else-if statement.
3. To “roll” the dice, use the nextInt method of the random number generator to generate an integer from 1 to 6.

Repeat while the number of dice rolls are less than the number of times the dice should be rolled.

Get the value of the first die by “rolling” the first die

Get the value of the second die by “rolling” the second die

If the value of the first die is the same as the value of the second die

If value of first die is 1

Increment the number of times snake eyes were rolled

Else if value of the first die is 2

Increment the number of times twos were rolled

Else if value of the first die is 3

Increment the number of times threes were rolled

Else if value of the first die is 4

Increment the number of times fours were rolled

Else if value of the first die is 5

Increment the number of times fives were rolled

Else if value of the first die is 6

Increment the number of times sixes were rolled

Increment the number of times the dice were rolled

4. Compile and run. You should get numbers that are somewhat close to 278 for each of the different pairs of doubles. Run it several times. You should get different results than the first time, but again it should be somewhat close to 278.

Task #2 Using Other Types of Loops

1. Change the while loop to a **do-while** loop. Compile and run. You should get the same results.
2. Change the do loop to a **for** loop. Compile and run. You should get the same results.

Task #3 Writing Output to a File

1. Copy the files StatsDemo.java (see code listing 4.2) and Numbers.txt from the Student CD or as directed by your instructor.
2. First we will write output to a file:
 - a) Create a FileWriter object passing it the filename “Results.txt” (Don’t forget the needed import statement).
 - b) Create a PrintWriter object passing it the FileWriter object.
 - c) Since you are using a FileWriter object, add a throws clause to the main method header.
 - d) Print the mean and standard deviation to the output file using a three decimal format, labeling each.
 - e) Close the output file.
3. Compile, debug, and run. You will need to type in the filename **Numbers.txt**. You should get no output to the console, but running the program will create a file called **Results.txt** with your output. The output you should get at this point is: mean = 0.000, standard deviation = 0.000. This is not the correct mean or standard deviation for the data, but we will fix this in the next tasks.

Task #4 Calculating the Mean

1. Now we need to add lines to allow us to read from the input file and calculate the mean.
 - a) Create a FileReader object passing it the filename.
 - b) Create a BufferedReader object passing it the FileReader object.
2. Write a priming read to read the first line of the file.
3. Write a loop that continues until you are at the end of the file.
4. The body of the loop will
 - a) convert the line into a double value and add the value to the accumulator
 - b) increment the counter
 - c) read a new line from the file
5. When the program exits the loop close the input file.
6. Calculate and store the mean. The mean is calculated by dividing the accumulator by the counter.
7. Compile, debug, and run. You should now get a mean of 77.444, but the standard deviation will still be 0.000.

Task #5 Calculating the Standard Deviation

1. We need to reconnect to the file so that we can start reading from the top again.
 - a) Create a FileReader object passing it the filename.
 - b) Create a BufferedReader object passing it the FileReader object.
2. Reinitialize sum and count to 0.
3. Write a priming read to read the first line of the file.
4. Write a loop that continues until you are at the end of the file.
5. The body of the loop will

- a) convert the line into a double value and subtract the mean, store the result in difference
 - b) add the square of the difference to the accumulator
 - c) increment the counter
 - d) read a new line from the file
6. When the program exits the loop close the input file.
 7. The variance is calculated by dividing the accumulator (sum of the squares of the difference) by the counter. Calculate the standard deviation by taking the square root of the variance (Use Math.sqrt () to take the square root).
 8. Compile, debug, and run. You should get a mean of 77.444 and standard deviation of 10.021.

Code Listing 4.1 (DiceSimulation.java)

```

/**
 This class simulates rolling a pair of dice 10,000 times and
 counts the number of times doubles of are rolled for each different
 pair of doubles.
 */

import java.util.Random;           //to use the random number generator
public class DiceSimulation
{
    public static void main(String[] args)
    {
        final int NUMBER = 10000; //the number of times to roll the dice

        //a random number generator used in simulating rolling a dice
        Random generator = new Random();

        int die1Value;           // number of spots on the first die
        int die2Value;           // number of spots on the second die
        int count = 0;           // number of times the dice were rolled
        int snakeEyes = 0;       // number of times snake eyes is rolled
        int twos = 0;            // number of times double two is rolled
        int threes = 0;          // number of times double three is rolled
        int fours = 0;           // number of times double four is rolled
        int fives = 0;           // number of times double five is rolled
        int sixes = 0;           // number of times double six is rolled

        //ENTER YOUR CODE FOR THE ALGORITHM HERE

        System.out.println ("You rolled snake eyes " + snakeEyes +
            " out of " + count + " rolls.");
        System.out.println ("You rolled double twos " + twos +
            " out of " + count + " rolls.");
    }
}

```

```

        System.out.println ("You rolled double threes " + threes +
            " out of " + count + " rolls.");
        System.out.println ("You rolled double fours " + fours +
            " out of " + count + " rolls.");
        System.out.println ("You rolled double fives " + fives +
            " out of " + count + " rolls.");
        System.out.println ("You rolled double sixes " + sixes +
            " out of " + count + " rolls.");
    }
}

```

Code Listing 4.2 (StatsDemo.java)

```

import java.text.DecimalFormat; //for number formatting
import java.util.Scanner;       //for keyboard input
//ADD AN IMPORT STATEMENT HERE //for using files

public class StatsDemo
{
    public static void main(String [] args)//ADD A THROWS CLAUSE HERE
    {
        double sum = 0;    //the sum of the numbers
        int count = 0;     //the number of numbers added
        double mean = 0;   //the average of the numbers
        double stdDev = 0; //the standard deviation of the numbers
        String line;       //a line from the file
        double difference;  //difference between the value and the mean

        //create an object of type Decimal Format
        DecimalFormat threeDecimals = new DecimalFormat("0.000");
        //create an object of type Scanner
        Scanner keyboard = new Scanner (System.in);
        String filename;    // the user input file name

        //Prompt the user and read in the file name
        System.out.println("This program calculates statistics"
            + "on a file containing a series of numbers");
        System.out.print("Enter the file name: ");
        filename = keyboard.nextLine();

        //ADD LINES FOR TASK #4 HERE
        //Create a FileReader object passing it the filename
        //Create a BufferedReader object passing it the FileReader object.
        //priming read to read the first line of the file
        //create a loop that continues until you are at the end of the file
        //convert the line to double value, add the value to the sum
    }
}

```

```

        //increment the counter
        //read a new line from the file
    //close the input file
    //store the calculated mean

    //ADD LINES FOR TASK #5 HERE
    //create a FileReader object passing it the filename
    //create a BufferedReader object passing it the FileReader object.
    //reinitialize the sum of the numbers
    //reinitialize the number of numbers added
    //priming read to read the first line of the file
    //loop that continues until you are at the end of the file
        //convert the line into a double value and subtract the mean
        //add the square of the difference to the sum
        //increment the counter
        //read a new line from the file
    //close the input file
    //store the calculated standard deviation

    //ADD LINES FOR TASK #3 HERE
    //create an object of type FileWriter using "Results.txt"
    //create an object of PrintWriter passing it the FileWriter object.
    //print the results to the output file
    //close the output file
}
}

```