**Programming Assignment #2**

**Question 1:**

a) Write a program RollDice11.java that takes an integer argument n, and rolls 11 fair six-sided dice, n times (You may select n as 3000). Use an integer array X to tabulate the number of times each possible total (between 11 and 66) occurs. Then print a text histogram of the results, as illustrated below.

<Output>

11:

12:

13:

14:

15:

16:

17:

18: \*

19: \*\*\*\*

20:

21: \*\*\*

22: \*\*\*\*\*\*

23: \*\*\*\*\*\*\*\*

24: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

25: \*\*\*\*\*\*\*\*\*\*\*\*\*

26: \*\*\*\*\*\*\*\*\*\*

27: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

28: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

29: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

30: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

31: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

32: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

33: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

34: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

35: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

36: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

37: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 38: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

39: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

40: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

41: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

42: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

43: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

44: \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

45: \*\*\*\*\*\*\*\*\*

46: \*\*\*\*\*\*\*\*\*\*\*

47: \*\*\*\*\*\*\*

48: \*\*\* 49: \*\*

50:

51:

52: \*

53:

54:

55:

56:

57:

58:

59:

60:

61:

62:

63:

64:

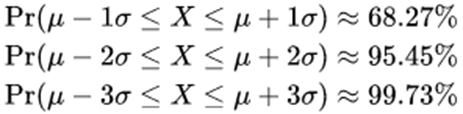
65:

66:

<End Output>

1. Find the average (μ) and the standard deviation (σ) for the set of values in X.
2. Find whether the distribution sample in X comply with the 68–95–99.7 rule

[(https://en.wikipedia.org/wiki/68%E2%80%9395%E2%80%9399.7\_rule)](https://en.wikipedia.org/wiki/68%E2%80%9395%E2%80%9399.7_rule). You must find the number of samples and their percentages in the following intervals. Check whether you get the similar percentages or not.



1. Make a normality test using Shapiro–Wilk test. Find, download and use a java library that includes the Shapiro–Wilk test code [(https://github.com/elcronos/shapiro-wilk](https://github.com/elcronos/shapiro-wilk)[, https://sourceforge.net/projects/jdistlib/,](https://sourceforge.net/projects/jdistlib/) or [https://github.com/datumbox/datumbox-framework.](https://github.com/datumbox/datumbox-framework) )

**Question 2:**

In this question, you are going to write methods that operate on matrices. The program reads values of matrices A, B, and C stored in a file called inputs.txt. This file should be placed under “src” directory. The first line before each matrix contains the number of rows and the number of columns as shown below. <Input>

4 4

56 57 58 56

66 66 66 67

77 77 77 78

88 88 88 89

4 4

[4 5 6 7 2 2 2 2](#_Toc8903)

[3 3 3 3](#_Toc8904)

[4 4 4 4](#_Toc8905)

[415 16 17 18 4](#_Toc8906)

22 24 24 25

33 34 35 36

44 45 46 47

<End Input>

As a first step, the program reads data for matrices A, B and C from inputs.txt file and write them into console. A forth matrix D is generated randomly. The program code for reading and writing matrices, and generating a matrix D is given below.

<Code> **import** java.io.File;

**import** java.io.FileNotFoundException; **import** java.util.Scanner; **public class** Matrices { **static** Scanner *scanner*;

**public static void** main(String[] args) {

// Read data from inputs.txt file placed in src directory(package)

File inputFile = **new** File(".//src//inputs.txt"); **try** {

*scanner* = **new** Scanner(inputFile); } **catch** (FileNotFoundException e) {

System.***out***.println("inputs.txt file not found");

e.printStackTrace();

}

// Read Matrices from inputs.txt file

System.***out***.println("Reading data from inputs.txt file placed in src directory(package)");

// Read first matrix // Read the number of rows **int** rows, columns; rows = *scanner*.nextInt(); // Read the number of columns columns = *scanner*.nextInt(); // Read first Matrix

**int**[][] a = *readMatrix*(rows, columns); // Read the second matrix rows = *scanner*.nextInt(); // Read the number of columns columns = *scanner*.nextInt(); // Read first Matrix

**int**[][] b = *readMatrix*(rows, columns);

// Read the third matrix rows = *scanner*.nextInt(); // Read the number of columns columns = *scanner*.nextInt(); // Read first Matrix

**int**[][] c = *readMatrix*(rows, columns);

// Generate a forth matrix with random number generator **int**[][] d = *generateMatrix*(rows, columns);

// Print input Matrices

System.***out***.println();

System.***out***.println(" \*\*\*\*\*\*\* Matrix A \*\*\*\*\*\*\*");

*printMatrix*(a); System.***out***.println();

System.***out***.println(" \*\*\*\*\*\*\* Matrix B \*\*\*\*\*\*\*");

*printMatrix*(b); System.***out***.println();

System.***out***.println(" \*\*\*\*\*\*\* Matrix C \*\*\*\*\*\*\*");

*printMatrix*(c); System.***out***.println();

System.***out***.println(" \*\*\*\*\*\*\* Matrix D \*\*\*\*\*\*\*"); *printMatrix*(d); }

// reads a matrix

**public static int**[][] readMatrix(**int** rows, **int** columns) { **int**[][] result = **new int**[rows][columns]; **for** (**int** i = 0; i < rows; i++) { **for** (**int** j = 0; j < columns; j++) { result[i][j] = *scanner*.nextInt();

} } **return** result;

}

// generate and return a random M-by-N matrix with values between 1 and 100 **public static int**[][] generateMatrix(**int** rows, **int** columns) { **int**[][] result = **new int**[rows][columns]; **for** (**int** i = 0; i < rows; i++) { **for** (**int** j = 0; j < columns; j++) { result[i][j] = (**int**) (Math.*random*() \* 100) + 1;

} } **return** result;

}

// prints a matrix

**public static void** printMatrix(**int**[][] matrix) { **int** rows = matrix.length; **int** columns = matrix[0].length; **for** (**int** i = 0; i < rows; i++) { **for** (**int** j = 0; j < columns; j++) { System.***out***.printf("%6d ", matrix[i][j]);

}

System.***out***.println();

}

}

// returns a \* b

**public static int**[][] multiply(**int**[][] a, **int**[][] b) {

}

// returns the transpose of a

**public static int**[][] transpose(**int**[][] a) {

}

// returns a + b

**public static int**[][] add(**int**[][] a, **int**[][] b) {

}

// returns a - b

**public static int**[][] subtract(**int**[][] a, **int**[][] b) {

}

}

<End code>

You are going to write the code for the following methods:

**public static int**[][] multiply(**int**[][] a, **int**[][] b) **public static int**[][] transpose(**int**[][] a) **public static int**[][] add(**int**[][] a, **int**[][] b) **public static int**[][] subtract(**int**[][] a, **int**[][] b)

By using these methods, your program will perform several matrix operations and print the results as shown below. Complete the code given above so that it will produce an output as follows (The numbers in the folllowing output may not be correct.)

<Output>

Reading data from inputs.txt file placed in src directory(package)

\*\*\*\*\*\*\* Matrix A \*\*\*\*\*\*\*

56 57 58 56

66 66 66 67

77 77 77 78

88 88 88 89

\*\*\*\*\*\*\* Matrix B \*\*\*\*\*\*\*

4 5 6 7

2 2 2 2

# 3 3 3 3

# 4 4 4 4

\*\*\*\*\*\*\* Matrix C \*\*\*\*\*\*\*

# 15 16 17 18

22 24 24 25

33 34 35 36

44 45 46 47

\*\*\*\*\*\*\* Matrix D \*\*\*\*\*\*\*

55 55 100 18

33 85 67 1

92 55 82 16

57 11 11 11

\*\*\*\*\* Transpose of C \*\*\*\*\*

1. 22 33 44
2. 24 34 45
3. 24 35 46
4. 25 36 47

\*\*\*\*\* A – B + D \*\*\*\*\*

56 57 58 60

68 68 68 69

80 80 80 81

92 92 92 93

\*\*\*\*\* (A – B + D) \* (Transpose of C) \*\*\*\*\*

156 157 58 60

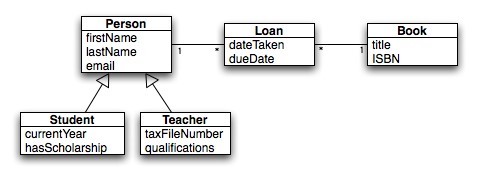
168 268 68 69

180 280 81 81

192 592 92 93

**Question 3:**

The following is UML diagram that shows the relationships amoung several classes.



1. Implement the classes and their relationships.
2. Write constructors, setter and getter methods for each class.
3. Write toString() method for each class.
4. Write a class to test the classes.