**LAB 14**

**Driver:** [Nachiketh Mamidi](mailto:nachiketh.mamidi@sjsu.edu)

**Scriber: Ayush Shresth**

**Date: 05/14/2021**

PART A

1. We will have an class with methods and the instance variables with an arraylist such as-
   1. borrow/add
   2. Remove
   3. collect fine
   4. reservations.
2. and we have to create another class to create an object to read the variables with parameters such as-
   1. check in date
   2. Check out date
   3. Item’s name

PART B

1. The input and output occurs in dataAnalyzer1. DataAnalyzer uses the methods from dataset, scanner and printstream
2. The dataAnalyzer2 class does not use coupling. The major disadvantage to this is that if we want to add any method or change any method we will have to change and create all of the variables in the main method.

public static void main(String[] args)

{

Scanner in = new Scanner(System.in);

DataSet data = new DataSet();

DataSet data2 = new DataSet();

boolean done = false;

while (!done)

{

System.out.print("Enter value, Q to quit: ");

String input = in.next();

if (input.equalsIgnoreCase("Q"))

done = true;

else

{

double x = Double.parseDouble(input);

data.add(x);

}

}

boolean done2 = false;

while (!done2)

{

System.out.print("Enter value, Q to quit: ");

String input = in.next();

if (input.equalsIgnoreCase("Q"))

done2 = true;

else

{

double x = Double.parseDouble(input);

data2.add(x);

}

}

System.out.println("Average\_1 = " + data.getAverage());

System.out.println("Maximum\_1 = " + data.getMaximum());

System.out.println("Average\_2 = " + data2.getAverage());

System.out.println("Maximum\_2 = " + data2.getMaximum());

}

5.

public static void main(String[] args)

{

Scanner in = new Scanner(System.in);

double sum = 0;

double count = 0;

double maximum = 0;

boolean done = false;

while (!done)

{

System.out.print("Enter value, Q to quit: ");

String input = in.next();

if (input.equalsIgnoreCase("Q"))

done = true;

else

{

double x = Double.parseDouble(input);

sum = sum + x;

if (count == 0 || maximum < x) maximum = x;

count++;

}

}

double average;

if (count == 0) average = 0;

else average = sum / count;

double sum2 = 0;

double count2 = 0;

double maximum2 = 0;

boolean done2 = false;

while (!done2)

{

System.out.print("Enter value, Q to quit: ");

String input = in.next();

if (input.equalsIgnoreCase("Q"))

done2 = true;

else

{

double x = Double.parseDouble(input);

sum2 = sum2 + x;

if (count2 == 0 || maximum2 < x) maximum2 = x;

count2++;

}

}

double average2;

if (count2 == 0) average2 = 0;

else average2 = sum2 / count2;

System.out.println("Average\_1 = " + average);

System.out.println("Maximum\_1 = " + maximum);

System.out.println("Average\_2 = " + average2);

System.out.println("Maximum\_2 = " + maximum2);

}

6. The advantage in dataAnalyzer1 is that we don’t have to create all of the new variables as it uses the dataSet class.

PART C

1. We can use seed to generate the same results.
2. Use sides as a seed to generate the same results.

public Die(int s)

{

sides = s;

generator = new Random(s);

}

1. We use a double variable to get 6 random numbers and then use another int variable to generate numbers from 1 to 6 by casting it to int.

/\*\*

\* Write a description of class MyRandom here.

\*

\* @author (your name)

\* @version (a version number or a date)

\*/

public class MyRandom

{

// instance variables - replace the example below with your own

private int num;

/\*\*

\* Constructor for objects of class MyRandom

\*/

public MyRandom(int var)

{

num = var;

}

/\*\*

\* Generating an integer between m and n (inclusive)

\*

\* @param

\* @return

\*/

public String randomInt(int m, int n)

{

Return 0;

}

/\*\*

\* Generating a floating-point number between x (inclusive) and y (exclusive)

\*

\* @param

\* @param

\* @return

\*/

public String randomDouble(double x, double y)

{

Return 0;

}

}

1. We can implement MyRandom class in the die class.

public class Die

{

private int sides;

MyRandom random;

/\*\*

Constructs a die with a given number of sides.

@param s the number of sides, e.g. 6 for a normal die

\*/

public Die(int s)

{

sides = s;

random = new MyRandom(s);

}

/\*\*

Simulates a throw of the die

@return the face of the die

\*/

public String cast()

{

return random.randomInt(1, sides);

}

}