Informatics Institute of Technology

Department of Computing

Module:4COSC006C- Programming Principles 01

Module Leader- Mr. Guganathan Poravi

Coursework Details: 2017/18 Practical Exercises - Referral/Deferral

Date of submission: 9th July 2018

Method of Submission: Submitted online via Blackboard

Deferral Coursework

Student ID: **2016053**

UoW ID: **w1673610**

Student First Name: **Dharshi**

Student Last Name: **Gunawardana**

Table of Contents

1. Introduction

This Assignment is Practical Exercises- Referral/Deferral Coursework for the module Programming Principles 01 (4COSC006C).

This practical assignment covers how to Identify program requirements and select appropriate algorithms to implement them; Represent algorithms in a structured manner (e.g., by use of flow diagrams); Implement algorithms using an algorithmic, strongly typed programming language, and design and run appropriate tests on the resulting code; Write program code that conforms to norms of good style and meets generally accepted referencing criteria (Purdy, 2018).

This program is a Command Line (console) based computer program and was made using the Java Programming Language. This program has two Mini-assignments they are part1 and part2.

Part 1; Mini-Assignment has to create Grocery Discount application. A supermarket they want to award coupons for their customers how much spend to the groceries. Supermarket gives coupon percentage for every single spent money so then this system wants to calculate that percentage to those supermarket groceries and get total discount for per person. That program can display the amount spent and discount for individual customer; total spent and total discount for all customers. And that program has good Programming style, Referencing, Validation for I use Input Mismatch Exception, good Formatted results using (printf()), using loops Exit loop (-99) (Purdy, 2018); etc

Part 2; Mini-Assignment) has to create Sequence of Values application. In that wants to use loops; Statistics display, Formatted results, good Programming style, use recommended Referencing; and for extra requirements are Values entered 0.0 to 50.0 inclusive, Limited to 10 entered values (Purdy, 2018); etc

In this report there have five main topics for those two-mini assignments. In that main topics there have sub two topics for those two mini assignments. The two mini topics belongs these main topics those are;

Functional requirements, algorithm for each functional requirement separately, draw flow charts for the functional requirements (how the functional decomposition should be done in the overall application), and non-functional requirements. Next implement java code, screenshot of displays output from running applications, testing with black-box testing and white-box testing which should include valid and invalid scenarios covering all functional requirements. Lastly conclusion and references.

Tip: Get this document as pdf file or MS Word file then open “navigation panel” then it’s very helpful to find all headings and can identify all requirements.

2. Analysis

2.1. *Functional requirements*

Part 1 - Mini-Assignment: Grocery Discount

1) Program will prompt cashier input values.
2) Program will provide cashier to execute or terminate (-99).
3) Program will loop until the cashier enter -99.
4) Program will check that cashier enter the amount is greater than £0.
5) Program will validate cashier-inputs are in correct format (numeric values).
6) Find discount for individual customer (amount * coupon percentage / 100).
7) Find the amount spent to the groceries for individual person (given money spent with coupon percentage from the scenario).
8) Find the total money spent to the supermarket (adding amount to the loop).
9) Find the total discount spent during the program run (adding discounts to the loop).
10) View the total discount spent, total money spent, discount and coupon for a person (display to use printf() format).

Part 2 - Mini-Assignment: Sequence of Values (loop)

1) Program will prompt user input set of floating points values.
2) Program will provide a non-numeric character to terminate the program.
3) Program will loop until the user prompt values 10.
4) Program will check that user enter 0.0 to 50.0 inclusive.
5) Program will validate user-inputs are in correct format (one decimal point values).
6) Find the average of the values (using for loop).
7) Find the smallest of the values (using sorted array list).
8) Find the largest of the values (using sorted array list).
9) Find the range, that is the difference between the smallest and largest (largest - smallest).
10) View the average value, the smallest values, the largest values, the difference (display to use printf() format).

2.2. *Non-Functional requirements*

1) Program Can easily Maintain.
2) Has a good Capacity to store the data.
3) User friendly Performance in my program.

3. Design

Flow chart and Algorithms

Part 1 - Mini-Assignment: Grocery Discount

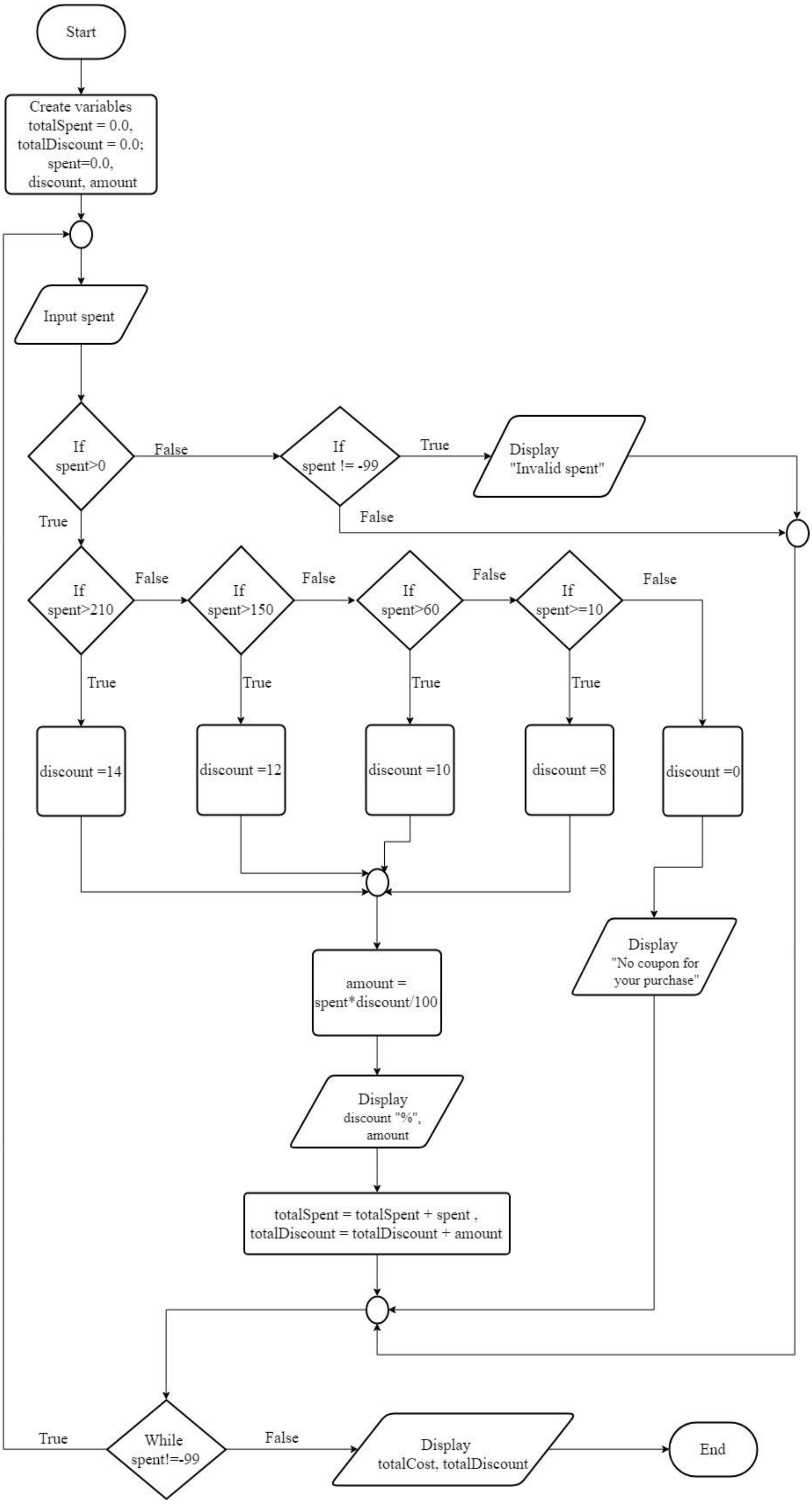
I. Calculate & print for individual customer that they received value of the coupon and the discount.

```
BEGIN
  SET spent, discount, amount
  PROMPT spent
  GET spent
    IF spent > 0 THEN
      IF spent > 210 THEN
        discount =14
      ELSE
        IF spent > 150 THEN
          discount =12
        ELSE
          IF spent > 60 THEN
            discount =10
          ELSE
            IF spent >= 10 THEN
              discount =8
            ELSE
              discount =0
              DISPLAY "No coupon for your purchase"
            END IF
          END IF
        END IF
      END IF
      amount = spent * discount /100
      DISPLAY discount “%’, amount
    END IF
END
```

II. Loop until user enter -99.

```
BEGIN
  TotalSpent=0.0, totalDiscount=0.0
  DOWHILE spent != -99
        PROMPT spent
        totalSpent = totalSpent + spent
        totalDiscount = totalDiscount + amount
  ENDDO
  Display totalSpent, totalDiscount
END
```

Above algorithm amount is customer received value of the coupon.



This flow chart is the structured algorithm. There includes cashier to execute or terminate

(-99), discount for individual customer and coupon percentage, total Discount and spends. Input mismatched exception is doesn’t considered this flow chart.

Part 2 - Mini-Assignment: Sequence of Values (loop)

I. Basic program

User have to entered floating point values and a non-numeric character to end the program

```
BEGIN
    SET character, decimal
    PROMPT value
    GET value
        decimal= [\\d*\\.\\d{1}]
        character = \\D
        IF input.match (decimal)
            SET value
        ELSE
            IF input.match (character)
                break
            END IF
        END IF
END
```

II. Extra requirements

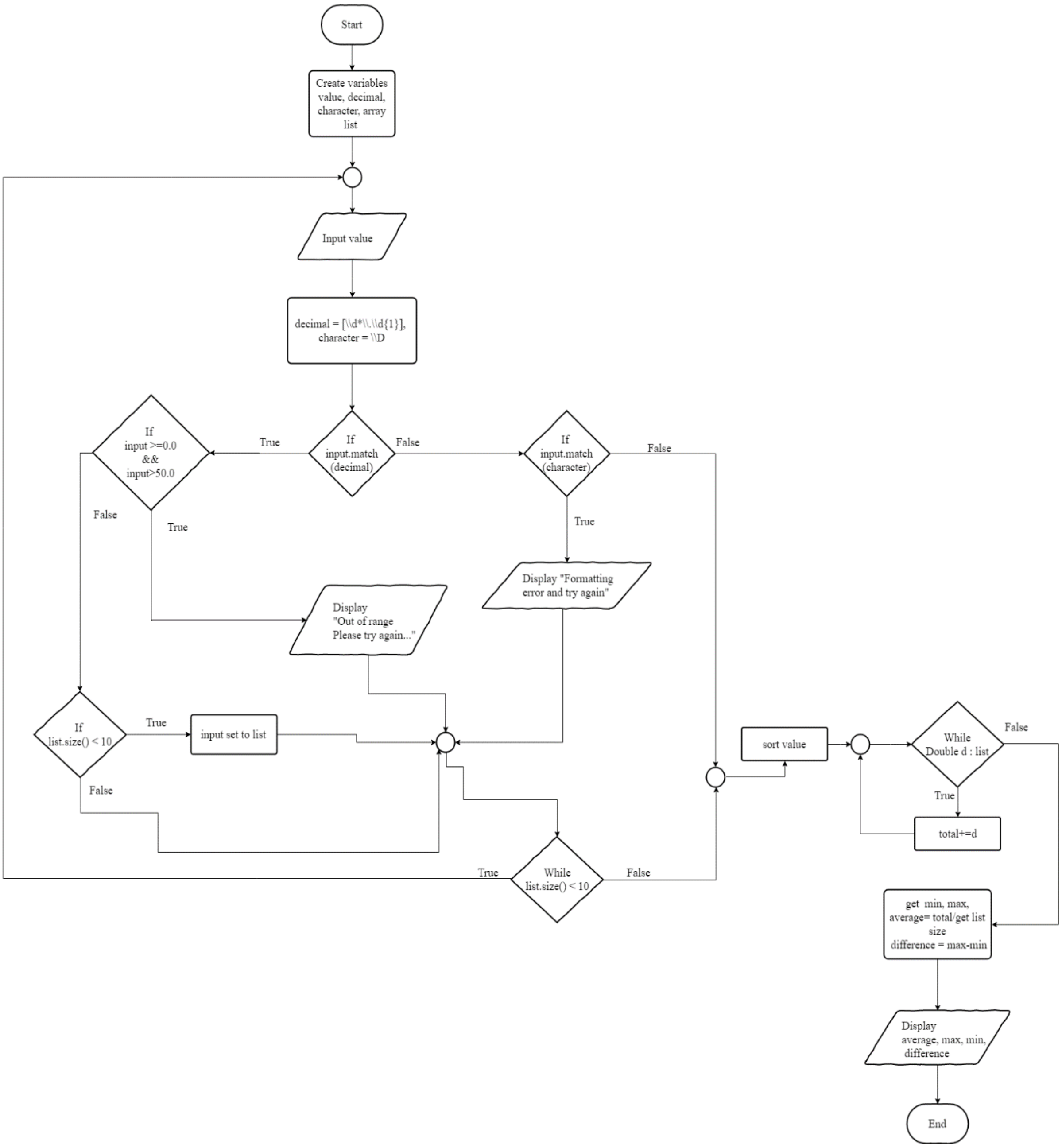
In this algorithm limited to 10 entered values

```
BEGIN
    DOWHILE list.size() < 10
        PROMPT value
    ENDDO
    Sort value
    For (d: list)
        total+=d
    Get lowest value
    Get highest value
    average = total/Get list size
    difference = highest -lowest
    DISPLAY lowest, highest, average, difference
END
```

In this algorithm for values that user entered 0.0 to 50.0 inclusive

```
BEGIN
    SET value
    PROMPT value
    GET value
        If value >=0.0 && value>50.0
            DISPLAY “error”
        ELSE
            Return value
        END IF
END
```

Below flow chart is cover 10 floating points (0.0-50.0 inclusive) values, non-numeric character to get the smallest, largest average, difference and sort the values that user enters.


Start
Create variables value, decimal, character, array list
Input value
decimal = [\\d*\\.\\d{1}], character = \\D
If input.match (decimal)
True
False
If input >=0.0 && input>50.0
If input.match (character)
False
False
True
True
Display "Formatting error and try again"
Display "Out of range Please try again..."
If list.size() < 10
True
input set to list
False
sort value
While Double d : list
False
True
total+=d
True
While list.size() < 10
False
get min, max, average= total/get list size difference = max-min
Display average, max, min, difference
End

When read the flow chart will helpful the Pseudocode (algorithm).

4. Implementation

Part 1 - Mini-Assignment: Grocery Discount

```
/**
 * Designed and implemented by A.A.D.M Gunawardana
 * Practical Exercises - Deferral
 * This is the Part 1 - Mini-Assignment: Grocery Discount system
 */
package GroceryDiscount.Part_1.CWK.Def;

import java.util.InputMismatchException;
import java.util.Scanner;

public class GroceryDiscount{

    public static void main(String[] args) {
        System.out.printf("****** Grocery Discount ******");
        new GroceryDiscount().execute();    //create an object for Grocery Discount
    }

    /**
     * Main method for Discount Calculations
     */
    private void execute() {
        Double totalCost = 0.0;
        Double totalDiscount = 0.0;
        Double cost = 0.0;
        int totalCustomer =0;
        do {
            System.out.print("\nPlease Enter the cost of your groceries");
            Scanner scanner = new Scanner(System.in);
            try{
                cost = scanner.nextDouble();

                if (cost > 0) {
          //To check that the amount entered is greater than £0.
                    Double discountPercent = getDiscountPercent(cost);
Double discountAmount = cost * discountPercent / 100;       // Calculating discount amount
                    totalCost += cost;              // updating total cost
                    totalDiscount += discountAmount;   // updating total Discount
                    totalCustomer++;     //update total customer
                    if (discountPercent == 0.0) {
                        System.out.printf("No coupon for your purchase.");
                    } else {
                        System.out.printf(" You win a discount coupon of £%.2f (%.0f%% of your purchase).",discountAmount,discountPercent);
    // (oracle, 2017)
                    }
                    System.out.printf("%n----------------------------------------------------------------%n");

                } else if (cost != -99) {    //To check that the cashier not wants to exit the program
                    System.out.printf("Invalid cost !%n");
                }

            }catch (InputMismatchException ime){        // (oracle, 2018)    //To check that the amount entered is non-numeric
                System.out.printf("%nError!!!%n");
            }
        } while (cost != -99);      // loop for any cost amount other than -99 (total summary display command)
        System.out.printf("%n==========Summary of %d customers==========%n%n",totalCustomer);    //to get valid total customers
        System.out.printf("Total cost: £%.2f  %n" ,totalCost);
        System.out.printf("Total Discount : £%.2f  %n" , totalDiscount);
        // Exit system
    }

    /**
     * Returns DiscountPercent for a given cost
     *
     * @param cost
     * @return discountPercent
     */
    private Double getDiscountPercent(Double cost) {
        if (cost > 210) {
            return 14.0;
        } else if (cost > 150) {
            return 12.0;
        } else if (cost > 60) {
            return 10.0;
        } else if (cost >=10) {
            return 8.0;
        } else {
            return 0.0;
        }
    }
}
```

Part 2 - Mini-Assignment: Sequence of Values (loop)

```
/**
 * Designed and implemented by A.A.D.M Gunawardana
 * Practical Exercises - Deferral
 * This is the Part 2 - Mini-Assignment: Sequence Of Values (loop)
 * Used Array List, do-while loop, if-else conditions, regular expressions to manipulating strings
 */
package SequenceOfValue.Part_2.CWK.Def;

import java.awt.List;
import java.util.ArrayList;
import java.util.Scanner;

public class SequenceOfValue {

    public static void main(String[] args) {
            ArrayList<Double> list = new ArrayList<>();
            Scanner scanner = new Scanner(System.in);
            System.out.printf("****** Sequence Of Values ******");
            System.out.printf("\n"); // \n to find a new line

            do {
                    System.out.println("\nPlease Enter a value");
                    String inputVal = scanner.next();
                    boolean isCharacter = inputVal.matches("\\D");      // \D for find non-digit character

                    if (inputVal.matches("\\d*\\.\\d{1}")) {        //to check floating point // (zeroturnaround, n.d.)
                            double formatedInput = Double.parseDouble(inputVal);
            //to convert String to double for the further calculations, conditions

                            if(formatedInput>=0.0 && formatedInput>50.0) {
                                    //not 0.0 to 50.0
                                    System.out.printf("Out of range...Please try again...%n");

                            }else if(list.size() < 10){  //list doesn't have 10 values
                                    list.add(Double.parseDouble(inputVal)); // add value to the list
                            }

                    }else { // above conditions false user enter value check exit the terminate or execute  the program.

                            if(isCharacter) {  //user enter non-numeric character
                                    System.out.printf("........... END THE PROGRAM ...........%n ");
                                    break;          //to break the loop
```

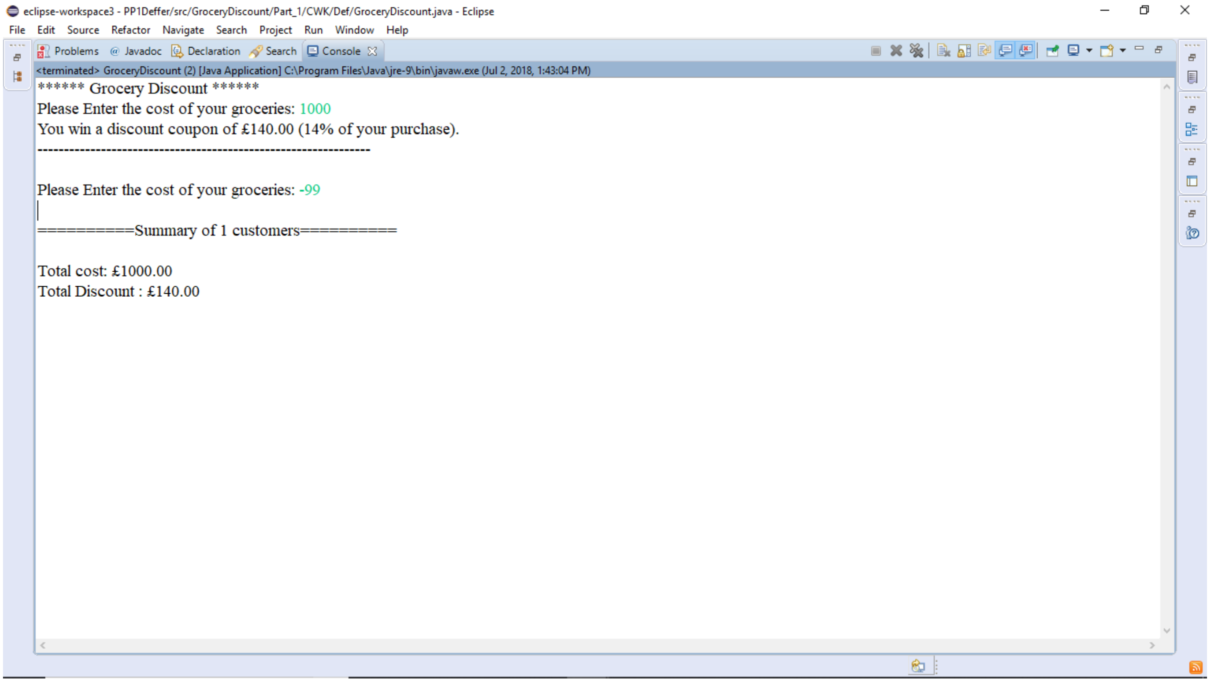
```
                }else { //user enter numeric character or mixed
                    System.out.printf("Formatting Error...Please try again...%n");
                }
            }

        }while (list.size() < 10);    //User can enter max 10 values (Summary display command)
        System.out.printf("%n============== Summary ==============%n");
        /**
         * sort values, calculate total using enhanced for loop
         */
        list.sort(null);                    //default
        double total=0.0;
        for (Double d : list) {
            total+=d;        //updating total
        }
        if(list.size()>0) {                            //To check that the list is greater than 0.
            System.out.printf("%n------------------------------------------------%n");
            System.out.printf("User entered %d set of valid floating-point values",list.size());
            System.out.printf("%n------------------------------------------------%n");
            /**
             * print Min value by sorted 0th value, Max value by sorted list size-1, Average by total of the sorted
             * values/list size, Difference by max-min, listed values
             */
            System.out.printf("The smallest value: %.1f %n",list.get(0));            //took one decimal points
            System.out.printf("The largest value: %.1f %n",list.get(list.size()-1));        //took one decimal points
            System.out.printf("The average of the values: %.2f %n",total/list.size());            //took two decimal points because for the accurate
            System.out.printf("Difference between the smallest and largest: %.2f%n",list.get(list.size()-1)-list.get(0));            //took one decimal points

            System.out.printf("%n Sorted values: ");
            for (int i=0; i<list.size(); i++)     //to show sorted values
                System.out.printf(+list.get(i) + "  ");
        }else {            //To check that the array list is empty
            System.out.printf("%n------------------------------------%n");
            System.out.printf("User entered all values are invalid %n");
            System.out.printf("------------------------------------");
        }
    }
}
```

5. Screenshots

Part 1 - Mini-Assignment: Grocery Discount

5.1.1. Prompt cashier input values.

Program will allow to Cashier to enter amounts are grater than 0 and “-99” to exit the program.

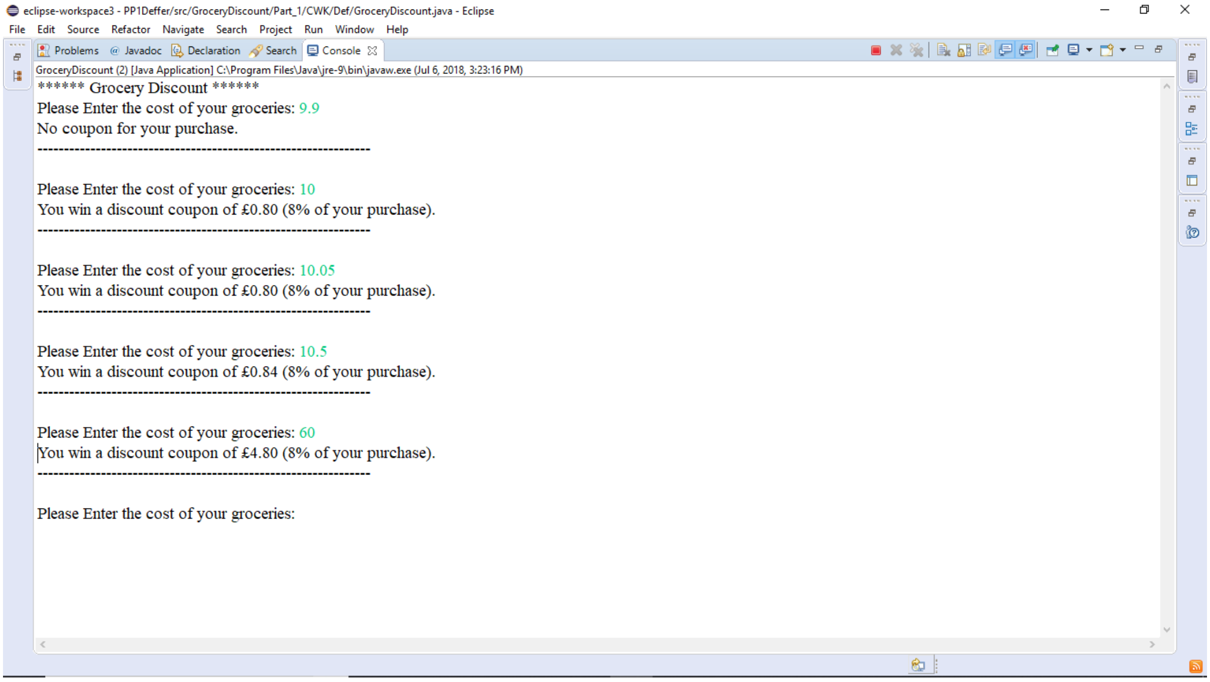


eclipse-workspace3 - PP1Deffer/src/GroceryDiscount/Part_1/CWK/Def/GroceryDiscount.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Problems @ Javadoc Declaration Search Console

```
<terminated> GroceryDiscount (2) [Java Application] C:\Program Files\Java\jre-9\bin\javaw.exe (Jul 2, 2018, 1:43:04 PM)
****** Grocery Discount ******
Please Enter the cost of your groceries: 1000
You win a discount coupon of £140.00 (14% of your purchase).
-------------------------------------------------------------

Please Enter the cost of your groceries: -99

==========Summary of 1 customers==========

Total cost: £1000.00
Total Discount : £140.00
```

Figure 1 Grocery Discount

In my program cashier enter real spends, so my program will take sterling pounds notes & coins amounts. it simply means my program can allow to cashier to enter decimal numbers or whole numbers both are taken to the program.

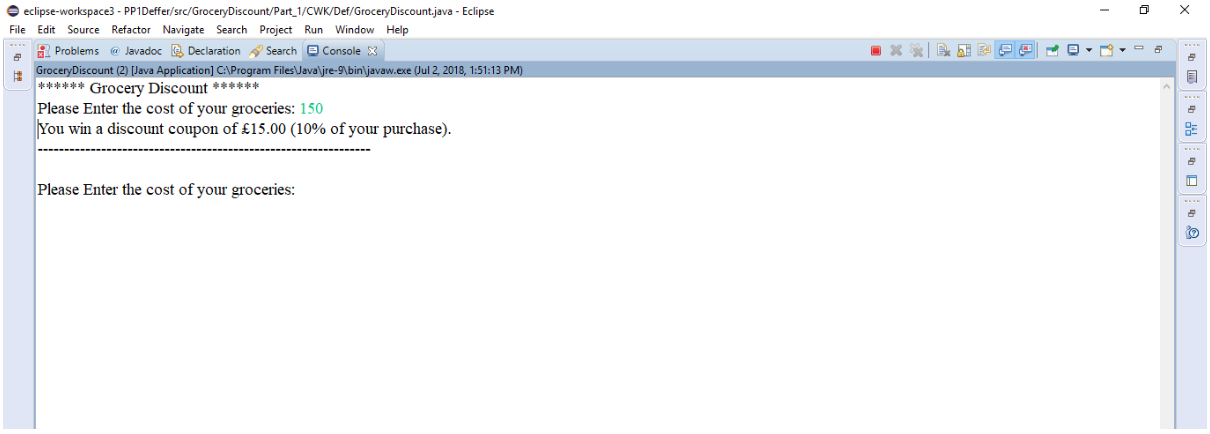


eclipse-workspace3 - PP1Deffer/src/GroceryDiscount/Part_1/CWK/Def/GroceryDiscount.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Problems @ Javadoc Declaration Search Console

```
GroceryDiscount (2) [Java Application] C:\Program Files\Java\jre-9\bin\javaw.exe (Jul 6, 2018, 3:23:16 PM)
****** Grocery Discount ******
Please Enter the cost of your groceries: 9.9
No coupon for your purchase.
-------------------------------------------------------------

Please Enter the cost of your groceries: 10
You win a discount coupon of £0.80 (8% of your purchase).
-------------------------------------------------------------

Please Enter the cost of your groceries: 10.05
You win a discount coupon of £0.80 (8% of your purchase).
-------------------------------------------------------------

Please Enter the cost of your groceries: 10.5
You win a discount coupon of £0.84 (8% of your purchase).
-------------------------------------------------------------

Please Enter the cost of your groceries: 60
You win a discount coupon of £4.80 (8% of your purchase).
-------------------------------------------------------------

Please Enter the cost of your groceries:
```

Figure 2 Grocery Discount

When cashier enter valid value then program automatically shows a single customer detail.



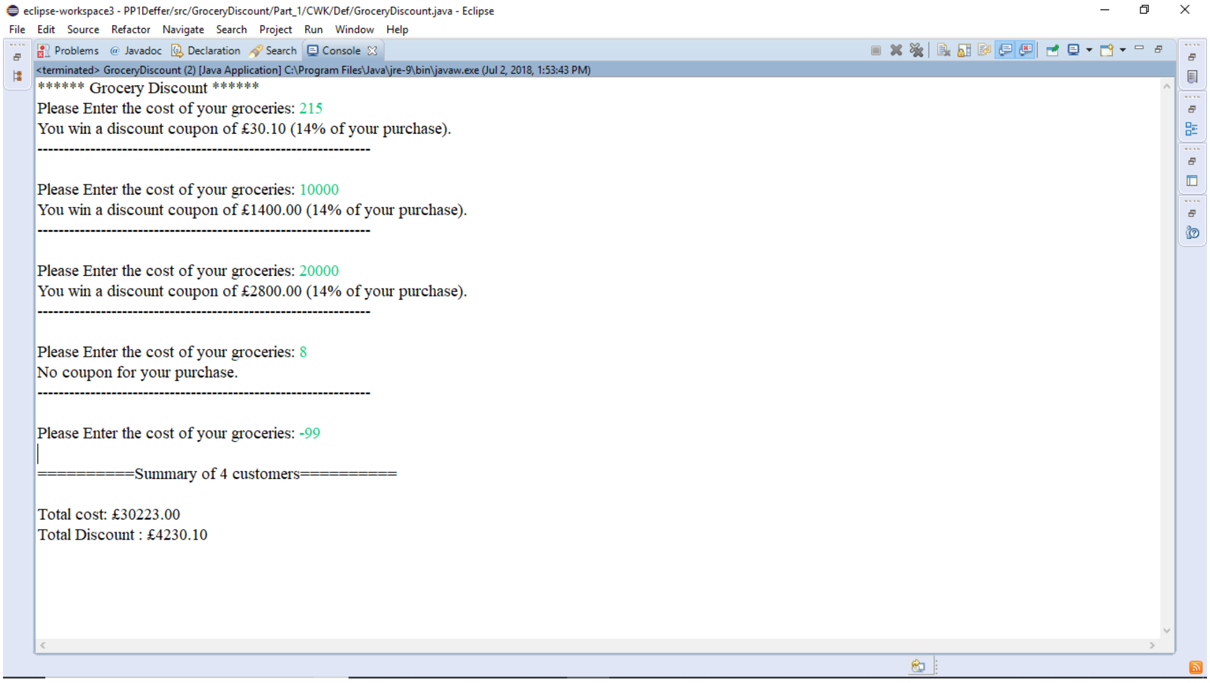
eclipse-workspace3 - PP1Deffer/src/GroceryDiscount/Part_1/CWK/Def/GroceryDiscount.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Problems Javadoc Declaration Search Console

```
GroceryDiscount (2) [Java Application] C:\Program Files\Java\jre-9\bin\javaw.exe (Jul 2, 2018, 1:51:13 PM)
****** Grocery Discount ******
Please Enter the cost of your groceries: 150
You win a discount coupon of £15.00 (10% of your purchase).
-------------------------------------------------------------

Please Enter the cost of your groceries:
```

Figure 3 Grocery Discount

5.1.2. Cashier to get total summary.

When cashier enter “-99” then terminate the program. So, break at the do loop; get sum of the discounts and money spent during the program run. I create two variables called total_Cost and total_Discount and loop them at the values are valid condition and update them separately I also get total of customers (at valid values). In my program shows how many customers are spends on the grocery, it is an extra feature.



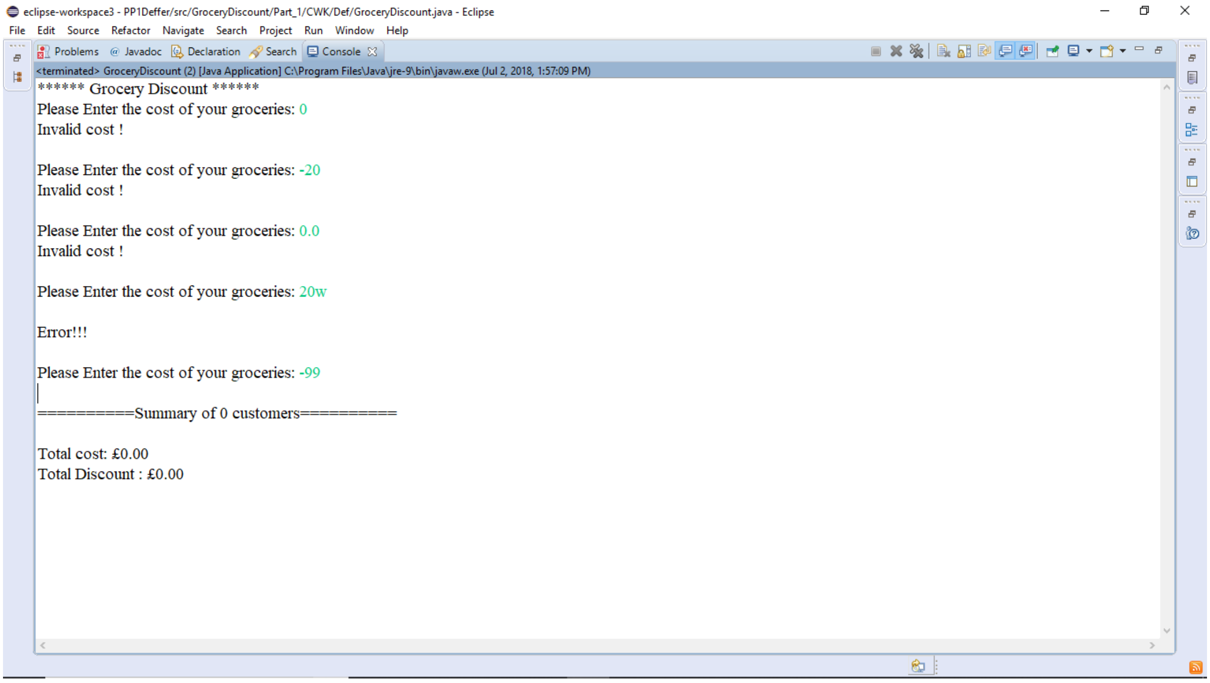
eclipse-workspace3 - PP1Deffer/src/GroceryDiscount/Part_1/CWK/Def/GroceryDiscount.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Problems Javadoc Declaration Search Console

```
<terminated> GroceryDiscount (2) [Java Application] C:\Program Files\Java\jre-9\bin\javaw.exe (Jul 2, 2018, 1:53:43 PM)
****** Grocery Discount ******
Please Enter the cost of your groceries: 215
You win a discount coupon of £30.10 (14% of your purchase).
-------------------------------------------------------------

Please Enter the cost of your groceries: 10000
You win a discount coupon of £1400.00 (14% of your purchase).
-------------------------------------------------------------

Please Enter the cost of your groceries: 20000
You win a discount coupon of £2800.00 (14% of your purchase).
-------------------------------------------------------------

Please Enter the cost of your groceries: 8
No coupon for your purchase.
-------------------------------------------------------------

Please Enter the cost of your groceries: -99

==========Summary of 4 customers==========

Total cost: £30223.00
Total Discount : £4230.10
```

Figure 4 Grocery Discount

5.1.3. Cashier entered value validation.

Cashier must enter valid numeric values and cashier enter positive values. It means catch user enter non-numeric character also check the condition cashier enter negative values for both given another chance to cashier enter a value.

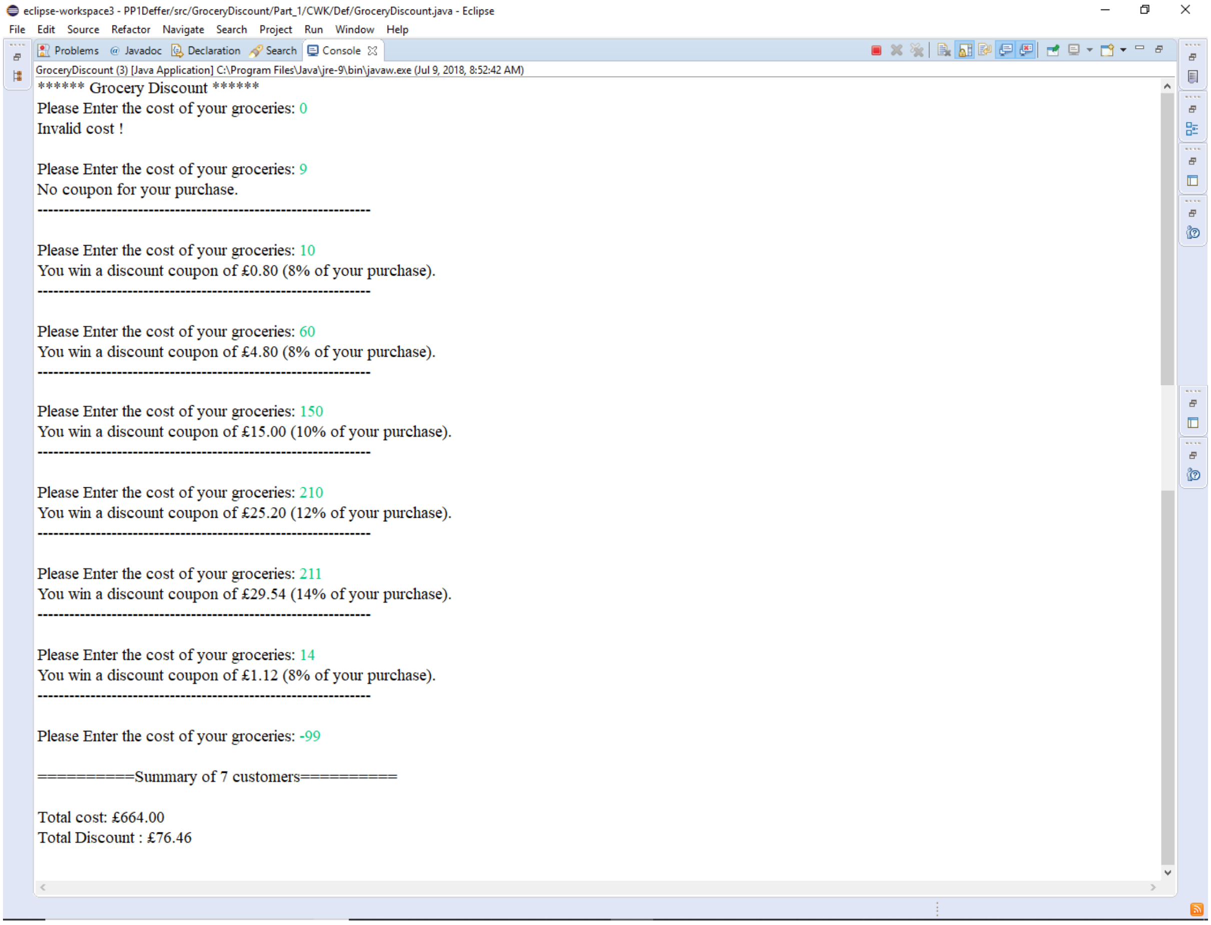

eclipse-workspace3 - PP1Deffer/src/GroceryDiscount/Part_1/CWK/Def/GroceryDiscount.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Problems @ Javadoc Declaration Search Console

```
<terminated> GroceryDiscount (2) [Java Application] C:\Program Files\Java\jre-9\bin\javaw.exe (Jul 2, 2018, 1:57:09 PM)
****** Grocery Discount ******
Please Enter the cost of your groceries: 0
Invalid cost !

Please Enter the cost of your groceries: -20
Invalid cost !

Please Enter the cost of your groceries: 0.0
Invalid cost !

Please Enter the cost of your groceries: 20w

Error!!!

Please Enter the cost of your groceries: -99

==========Summary of 0 customers==========

Total cost: £0.00
Total Discount : £0.00
```

Figure 5 Grocery Discount

5.1.4. <u>*Coupon that 8% 10% 12% 14% offered values*</u>

Less than or equal to £9, from £10 to £60, £61 to £150, £151 to £210, more than £211.


eclipse-workspace3 - PP1Deffer/src/GroceryDiscount/Part_1/CWK/Def/GroceryDiscount.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Problems @ Javadoc Declaration Search Console

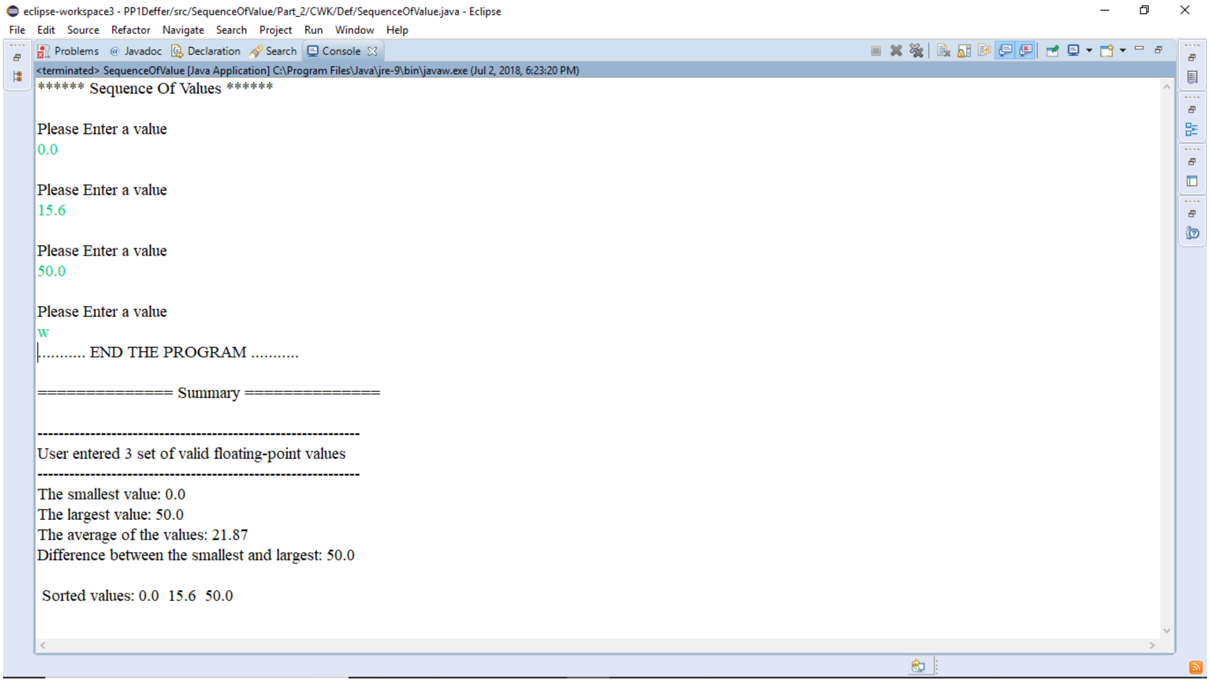
```
GroceryDiscount (3) [Java Application] C:\Program Files\Java\jre-9\bin\javaw.exe (Jul 9, 2018, 8:52:42 AM)
****** Grocery Discount ******
Please Enter the cost of your groceries: 0
Invalid cost !

Please Enter the cost of your groceries: 9
No coupon for your purchase.
--------------------------------------------------------------

Please Enter the cost of your groceries: 10
You win a discount coupon of £0.80 (8% of your purchase).
--------------------------------------------------------------

Please Enter the cost of your groceries: 60
You win a discount coupon of £4.80 (8% of your purchase).
--------------------------------------------------------------

Please Enter the cost of your groceries: 150
You win a discount coupon of £15.00 (10% of your purchase).
--------------------------------------------------------------

Please Enter the cost of your groceries: 210
You win a discount coupon of £25.20 (12% of your purchase).
--------------------------------------------------------------

Please Enter the cost of your groceries: 211
You win a discount coupon of £29.54 (14% of your purchase).
--------------------------------------------------------------

Please Enter the cost of your groceries: 14
You win a discount coupon of £1.12 (8% of your purchase).
--------------------------------------------------------------

Please Enter the cost of your groceries: -99

==========Summary of 7 customers==========

Total cost: £664.00
Total Discount : £76.46
```

Figure 6 Grocery Discount

Part 2 - Mini-Assignment: Sequence of Values (loop)

5.2.1. *Prompt user input set of floating points values.*

Program will allow to user to enter 0.0 to 50.0 inclusive set of floating-point values and any non-numeric character to exit the program. Floating point value checked by using if condition I take a String variable called inputVal then I checked (inputVal.matches("\\d*\\.\\d{1}")). It tests whether the regular expression matches the any digit before decimal point after decimal point there can be one digit. Then I convert that value string to double check the (0.0 - 50.0).



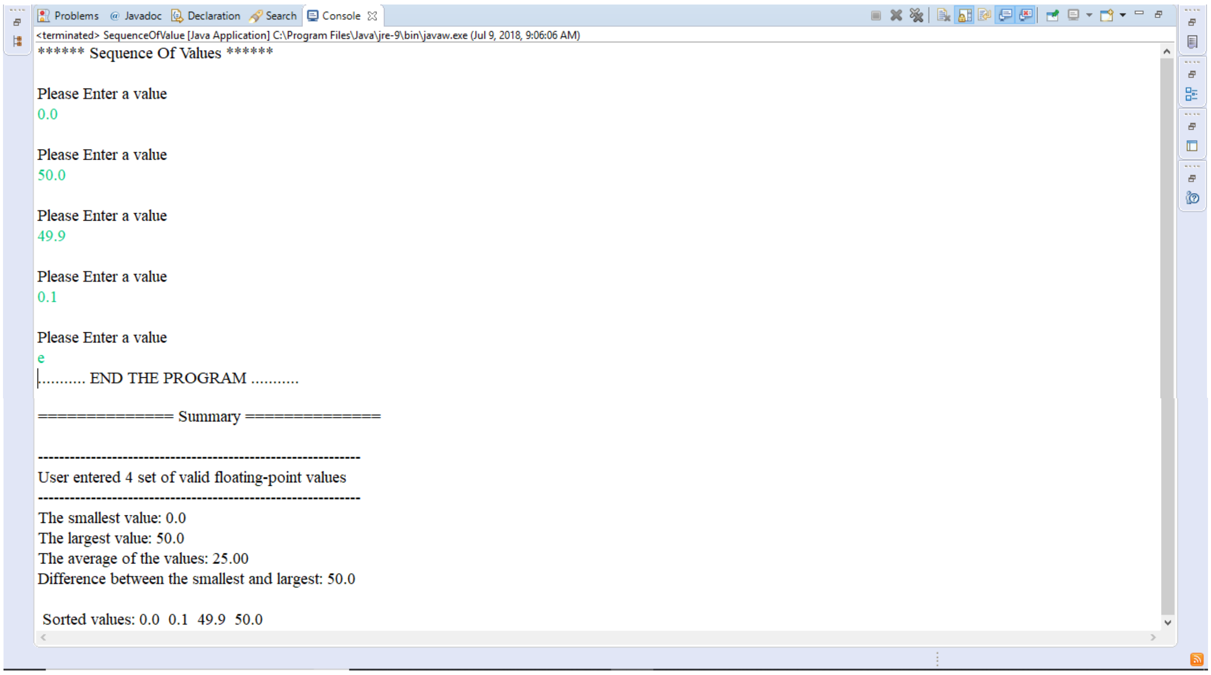
eclipse-workspace3 - PP1Deffer/src/SequenceOfValue/Part_2/CWK/Def/SequenceOfValue.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Problems @ Javadoc Declaration Search Console

```
<terminated> SequenceOfValue [Java Application] C:\Program Files\Java\jre-9\bin\javaw.exe (Jul 2, 2018, 6:23:20 PM)
****** Sequence Of Values ******

Please Enter a value
0.0

Please Enter a value
15.6

Please Enter a value
50.0

Please Enter a value
w
.......... END THE PROGRAM ..........

============== Summary ==============

------------------------------------------------------------
User entered 3 set of valid floating-point values
------------------------------------------------------------
The smallest value: 0.0
The largest value: 50.0
The average of the values: 21.87
Difference between the smallest and largest: 50.0

 Sorted values: 0.0  15.6  50.0
```

Figure 7 Sequence of Values

5.2.2. *From 0.0 to 50.0 inclusive*

Include 0.0 and 50.0 and all one decimal point floating points to that.



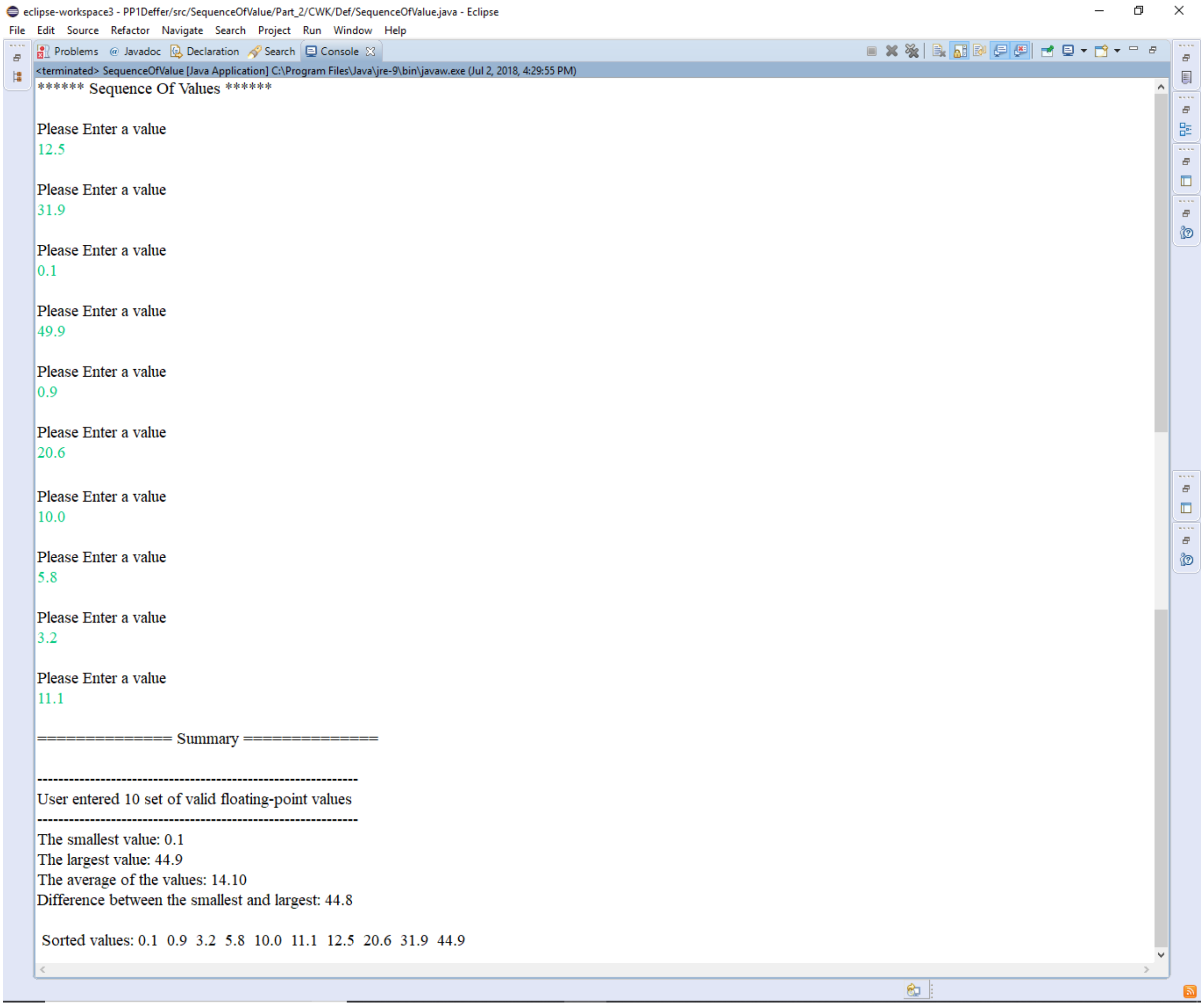
```
<terminated> SequenceOfValue [Java Application] C:\Program Files\Java\jre-9\bin\javaw.exe (Jul 9, 2018, 9:06:06 AM)
****** Sequence Of Values ******

Please Enter a value
0.0

Please Enter a value
50.0

Please Enter a value
49.9

Please Enter a value
0.1

Please Enter a value
e
.......... END THE PROGRAM ..........

============== Summary ==============

------------------------------------------------------------
User entered 4 set of valid floating-point values
------------------------------------------------------------
The smallest value: 0.0
The largest value: 50.0
The average of the values: 25.00
Difference between the smallest and largest: 50.0

 Sorted values: 0.0  0.1  49.9  50.0
```

Figure 8 Sequence of Values

5.2.3. *User to end the number series.*

The program has 2ways to terminate. *Figure 6* is the non-numeric character to exit. Then below way is another one.

Program will allow user to enter maximum 10 set of values. After that program will terminate; I used do-while loop to program to read only 10 valid values.


eclipse-workspace3 - PP1Deffer/src/SequenceOfValue/Part_2/CWK/Def/SequenceOfValue.java - Eclipse
File
Edit
Source
Refactor
Navigate
Search
Project
Run
Window
Help
Problems
Javadoc
Declaration
Search
Console

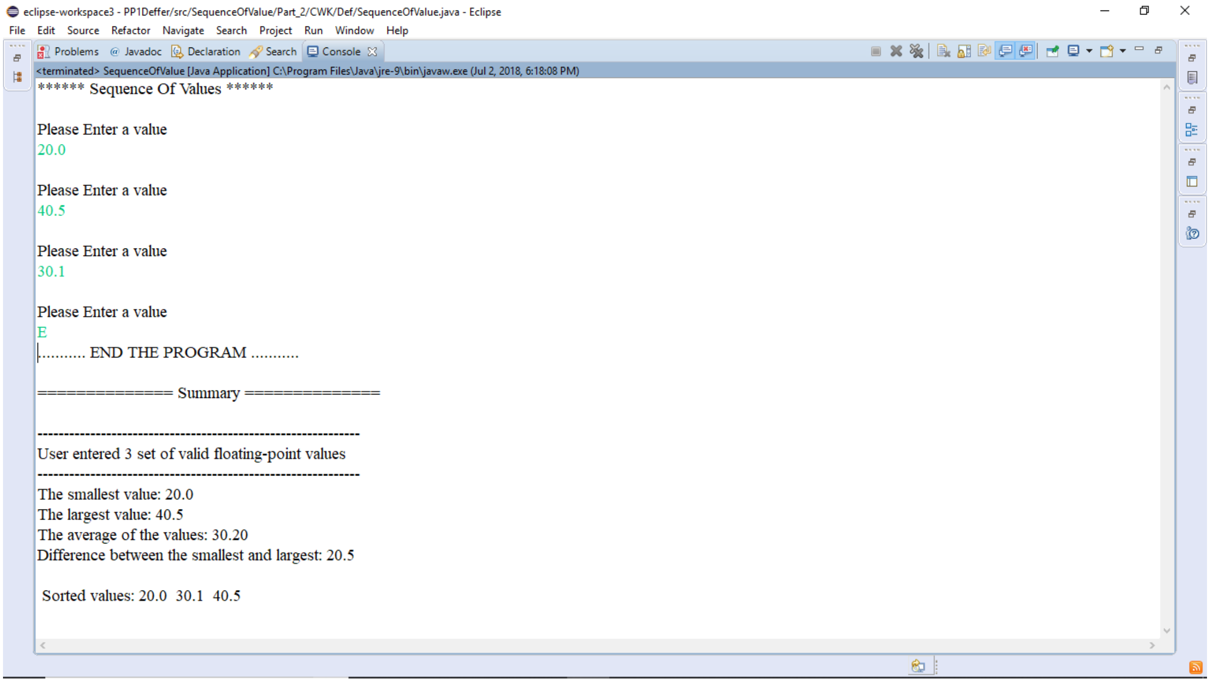
```
<terminated> SequenceOfValue [Java Application] C:\Program Files\Java\jre-9\bin\javaw.exe (Jul 2, 2018, 4:29:55 PM)
****** Sequence Of Values ******

Please Enter a value
12.5

Please Enter a value
31.9

Please Enter a value
0.1

Please Enter a value
49.9

Please Enter a value
0.9

Please Enter a value
20.6

Please Enter a value
10.0

Please Enter a value
5.8

Please Enter a value
3.2

Please Enter a value
11.1

============== Summary ==============

-----------------------------------------------------------
User entered 10 set of valid floating-point values
-----------------------------------------------------------
The smallest value: 0.1
The largest value: 44.9
The average of the values: 14.10
Difference between the smallest and largest: 44.8

 Sorted values: 0.1  0.9  3.2  5.8  10.0  11.1  12.5  20.6  31.9  44.9
```

Figure 9 Sequence of Values

5.2.4. *User to get summary of the floating-point values.*

There can 2ways to show summery. A non-numeric character entered and enter 10 set of values. In the scenario program can maximumly read 10 set of values. It should be the terminate (exit) point to the program refer *figure 7*. A non-numeric character to end the programme.

So, I use array list to store the data and sort them then take the smallest value which is the 0th value, largest value which is the last element of the list, then average which is sum of all elements divide by list size, lastly difference between them (largest – smallest). Then I get extra two features that are system shows sored values and how many values entered. I use for loop to show sorted values and the list size to get the how many values entered.

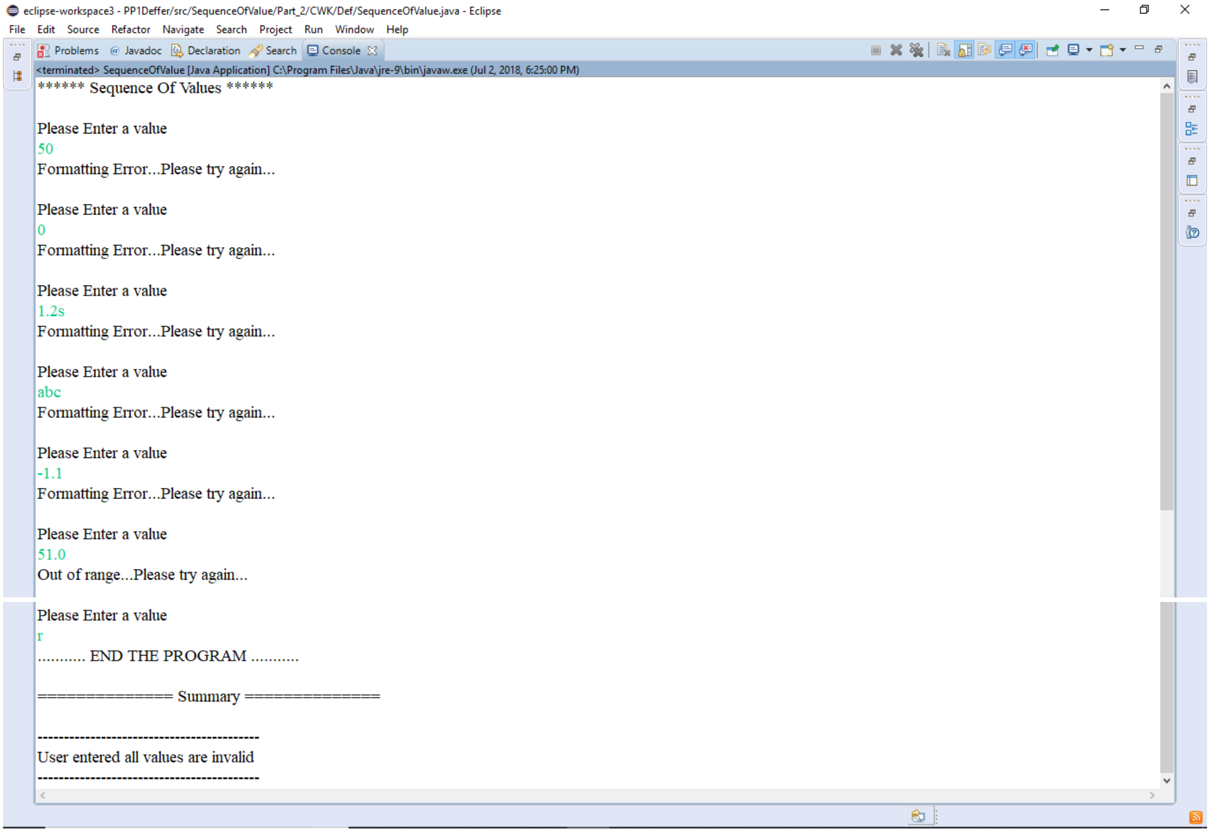

eclipse-workspace3 - PP1Deffer/src/SequenceOfValue/Part_2/CWK/Def/SequenceOfValue.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Problems Javadoc Declaration Search Console

```
<terminated> SequenceOfValue [Java Application] C:\Program Files\Java\jre-9\bin\javaw.exe (Jul 2, 2018, 6:18:08 PM)
****** Sequence Of Values ******

Please Enter a value
20.0

Please Enter a value
40.5

Please Enter a value
30.1

Please Enter a value
E
|.......... END THE PROGRAM ..........

============== Summary ==============

-------------------------------------------------------------
User entered 3 set of valid floating-point values
-------------------------------------------------------------
The smallest value: 20.0
The largest value: 40.5
The average of the values: 30.20
Difference between the smallest and largest: 20.5

 Sorted values: 20.0  30.1  40.5
```

Figure 10 Sequence of Values

5.2.5. User entered values validation.

There have 4 ways to validate the program. 1st is the range that user can entered is valid or not include the negative values, 2nd user can have entered maximum 10 set of values, 3rd user can have entered one non-numeric character to terminate the program, 4th user can only enter one decimal point values. Without 2nd and 3rd other options must be executing gracefully.


eclipse-workspace3 - PP1Deffer/src/SequenceOfValue/Part_2/CWK/Def/SequenceOfValue.java - Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Problems Javadoc Declaration Search Console

```
<terminated> SequenceOfValue [Java Application] C:\Program Files\Java\jre-9\bin\javaw.exe (Jul 2, 2018, 6:25:00 PM)
****** Sequence Of Values ******

Please Enter a value
50
Formatting Error...Please try again...

Please Enter a value
0
Formatting Error...Please try again...

Please Enter a value
1.2s
Formatting Error...Please try again...

Please Enter a value
abc
Formatting Error...Please try again...

Please Enter a value
-1.1
Formatting Error...Please try again...

Please Enter a value
51.0
Out of range...Please try again...

Please Enter a value
r
.......... END THE PROGRAM ..........

============== Summary ==============

------------------------------------------
User entered all values are invalid
------------------------------------------
```

Figure 11 Sequence of Values

6. Evaluation (Testing)

6.1. *Black Box Testing*

Part 1 - Mini-Assignment: Grocery Discount

Below test plan is to show grocery discount program's decision points there result in expected view and actually program views.

S. No	Test Description	Test Data	Expected	Actual	Status
		cost	result	Result	
1	Check the cost greater than 0	0	Ask user from another input	Show an error message and ask an input	Pass
2		-20			Pass
3	Check different values entered and get that discounts and coupons	3	No coupon	Show No coupon for your purchase	Pass
4		10	coupon: 8% discount: 0.8	You win a discount coupon of £0.80 (8% of your purchase).	Pass
5		60	coupon: 8% discount: 4.8	You win a discount coupon of £4.80 (8% of your purchase).	Pass
6		99	coupon: 10% discount: 9.9	You win a discount coupon of £9.90 (10% of your purchase).	Pass
7		150	coupon: 10% discount: 15	You win a discount coupon of £15.00 (10% of your purchase).	Pass
8		210	coupon: 12% discount: 25.2	You win a discount coupon of £25.20 (12% of your purchase).	Pass
9		211	coupon: 14% discount: 29.54	You win a discount coupon of £29.54 (14% of your purchase).	Pass
10	Check non-numeric character entered	abc	Ask user from another input	Show an error massage and ask an input	Pass
11	Check system end	-99	Total cost:743 Total discount: 85.24	Total cost: £743.00 Total discount: £85.24	Pass
12	Check cost enter -99 (without any previous values)	-99	Total cost:0 Total discount: 0	Total cost: £0.00 Total discount: £0.00	Pass

Table 1 Grocery Discount B/B

Part 2 - Mini-Assignment: Sequence of Values (loop)

Below test plan is to show sequence of values program's decision points there result in expected view and actually program views.

1. Test for floating point values

Test Case ID	Test Description	Test Data	Expected Result	Actual Result	Status
		value			
1	Check floating point value	2.4	Nothing	Get the input	Pass
2		2.04	Ask user from another input	Show an error message and ask an input	Pass

Table 2 Sequence of values B/B

2. Test for 10 values with 0.0-50.0 inclusive.

Test Case ID	Test Description	Test Data	Expected Result	Actual Result	Status
		value			
1	Check different 10 values entered	0.0	Get those values after 50.0 enter Smallest Value: 0.0 Largest Value: 50.0 Average of them: 24.50 Difference between largest and the smallest: 50.0	The smallest value: 0.0 The largest value: 50.0 The average of the values: 24.50 Difference between the smallest and largest: 50.0	Pass
2		0.1			Pass
3		1.1			Pass
4		33.3			Pass
5		25.7			Pass
6		31.6			Pass
7		5.5			Pass
8		47.8			Pass
9		49.9			Pass
10		50.0			Pass

3. Test for sorting the above values

	Sort		0.0, 0.1, 1.1, 5.5, 25.7, 31.6, 33.3, 47.8, 49.9, 50.0	Sorted values: 0.0 0.1 1.1 5.5 25.7 31.6 33.3 47.8 49.9 50.0	Pass

Table 3 Sequence of values B/B

4. Test for non-numeric character and numeric character

Test Case ID	Test Description	Test Data	Expected Result	Actual Result	Status
		value			
1	Check numeric and non-numeric character entered	abc	Ask user from another input	Show an error massage and ask an input	Pass
2		12.4	Nothing	Ask another input	Pass
3		-2.6	Ask user from another input	Show an error massage and ask an input	Pass
4		20			Pass
5		z	End the program & Smallest Value: 12.4 Largest Value: 12.4 Average of them: 12.4 Difference between largest and the smallest: 0.0	The smallest value: 12.4 The largest value: 12.4 The average of the values: 12.40 Difference between the smallest and largest: 0.0	Pass

Table 4 Sequence of values B/B

6.2. *White Box testing*

Part 1 - Mini-Assignment: Grocery Discount

cost	Branch coverage								Expected output	Actual output
	If (cost>0)					if (discount == 0.0)	Else if (cost! = -99)	While (cost! = -99)		
	if (cost > 0)	if (cost > 210)	if (cost > 150)	if (cost > 60	if (cost >=10)					
0	F						T	T	Ask user to input	Ask Input
5	T	-	-	-	-	T	-	T	No coupon Ask Input	Ask Input
10	T	F	F	F	T	F	-	T	8% Ask Input	Ask Input
60	T	F	F	F	T	F	-	T	8% Ask Input	Ask Input
150	T	F	F	T		F	-	T	10% Ask Input	Ask Input
210	T	F	T	-	-	F	-	T	12% Ask Input	Ask Input
211	T	T	-	-	-	F	-	T	14% Ask Input	Ask Input
-99	F						F	F	Total discount	Total discount

Table 5 Grocery Discount W/B

Part 2 - Mini-Assignment: Sequence of Values (loop)

value	Branch coverage					Expected output	Actual output
	if (inputVal.matches("\\d*\\.\\d {1}")			if(isCharacter)	While (list.size() < 10)		
	If (InpuVal match)	If (Input >=0.0 && Input> 50.0)	ELSE if (list.size() < 10)				
20	F		F	F	T	Ask user to input	Ask Input
0.0	T	T	T	-	T	Get the value Ask Input	Ask Input
3.3	T	T	T	-	T	Get the value Ask Input	Ask Input
50.0	T	T	T	-	T	Get the value Ask Input	Ask Input
5.06	F	-	-	F	T	Ask user for input	Ask Input
51.5	T	F	F	-	T	Ask user for input	Ask Input
-1.2	F	-	-	-	T	Ask user for input	Ask Input
abc	F	-	-	F	T	Ask user for input	Ask Input
a	F	-	-	T	F	Get results	Results
After enter 9 values… 10th value below							
5.8	T	T	T	-	F	Get results	Results

Table 6 Sequence of values W/B

For both part 1 and 2 below are very important.

Ask Input- “Please Enter a value” / “Enter groceries”

Results- smallest, largest, average and difference

Those white boxes are test program’s conditions and decision points effectively. (program path way). In white box testing actual output I show the java program console output. When I doing the white box testing I referred so many sites. (Themes, 2010)

6. Conclusion

In this assignment belongs two mini topics those are grocery management system and sequence value checker.

So, in that first I would like to give you a summery about grocery discount;

In that I shown analysis the specification in that I show all possible functional requirements and those requirements how to design. In design I show how to identify algorithms (pseudocode) and combination those requirements in one flow chart (algorithm in a structured manner). Then I show testing in the testing I do black box and white box testing. In the black box testing I basically shows how the functional requirements are testing and white box testing I basically shows how the code (in the code conditions) are testing (program decision points).

Secondly, I would like to give a summary about Sequence of Values;

The way I did this is same to the grocery discount's analysis, design, testing (black box and white box)

I would like to talk about how the implementation done to the grocery discount system…In that I covered calculates discount coupon, good programming style, use valid referencing, validation to the program, and formatted results using spec given way. Requirements are get that greater than 0 values, -99 to exit the program and display smallest, largest, average, difference between smallest and largest vales. Extra I shows how many values are read by the system.

Then Sequence of Values I covered loops with effectively using, statistics display, formatted results using spec given way, good programming style, use valid referencing criteria. Requirements are read floating point values entered 0.0 to 50.0 inclusive, limited to 10 entered values, non-numeric to exit the program and display total spent, total discount. Extra I shows how many values are read by the system and those values display small value to large value.

In these two assignments I got much deeper understand how to work with for-loop, do-while loops, if-else and else-if conditions, methods, formatting print statement, static display using regex, how to validate user input using try-catch etc. I believe that the program and the report are done by the Coursework Specifications.

7. References

oracle, 2017. *oracle java documentation.* [Online]
Available at: https://docs.oracle.com/javase/tutorial/java/data/numberformat.html
[Accessed 07 2018].

oracle, 2018. *Java SE Documentation.* [Online]
Available at:
https://docs.oracle.com/javase/8/docs/api/java/util/InputMismatchException.html#InputMismatchException-java.lang.String-
[Accessed 7 2018].

Purdy, W., 2018. *4COSC00W: Programming Principles I Assignment Specification (2017/8),* University of Westminster : Department of Computer Science .

S., C., 2012. *JAVA METHODS.* UK: Java Secrets.

Themes, E., 2010. *QA.* [Online]
Available at: http://softwaretestingtimes.com/2010/04/white-box-testing-simplified.html
[Accessed 7 2018].

zeroturnaround, n.d. *zeroturnaround regex sheet.* [Online]
Available at: http://files.zeroturnaround.com/pdf/zt_regular-expressions-cheat-sheet.pdf
[Accessed 07 2018].