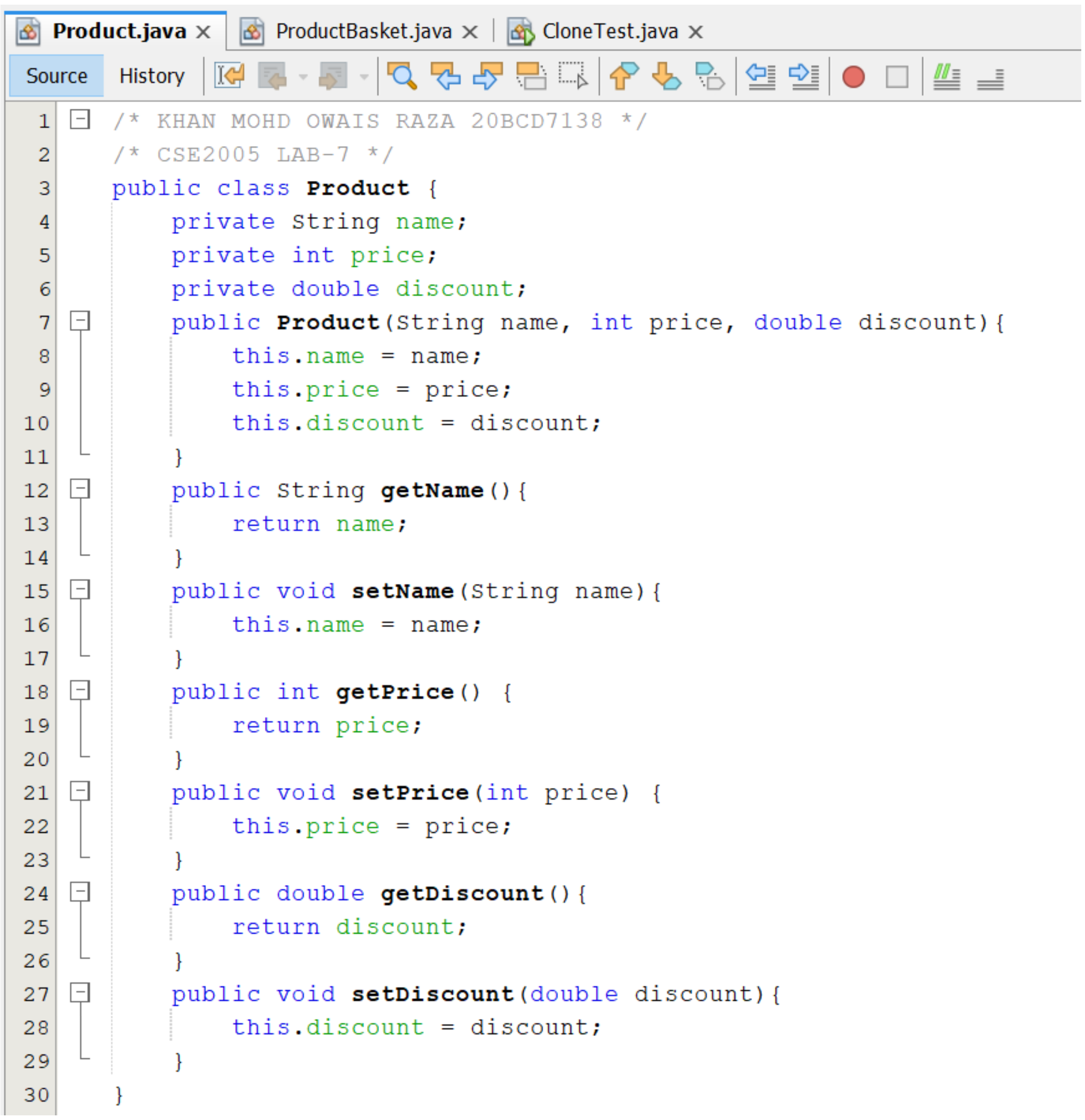
**CSE2005 (OOP in Java) Lab – 7** **09/12/2021**

**KHAN MOHD. OWAIS RAZA**
**20BCD7138**

Q.1] Consider a supermarket and develop following classes –
Product: It has three attributes name, price, and discount. Supply appropriate getter and setter methods.
ProductBasket: It is used to hold bunch of products before purchase of products.
Sales supervisor observes that two product baskets are very similar and he want to duplicate the bill generation of first basket to avoid re-entering the products of second basket for bill generation.
Apply suitable Object Cloning to generate different bills even if few items are deleted in generation of second bill. Also ensure that second bill modifications will not affect first bill.

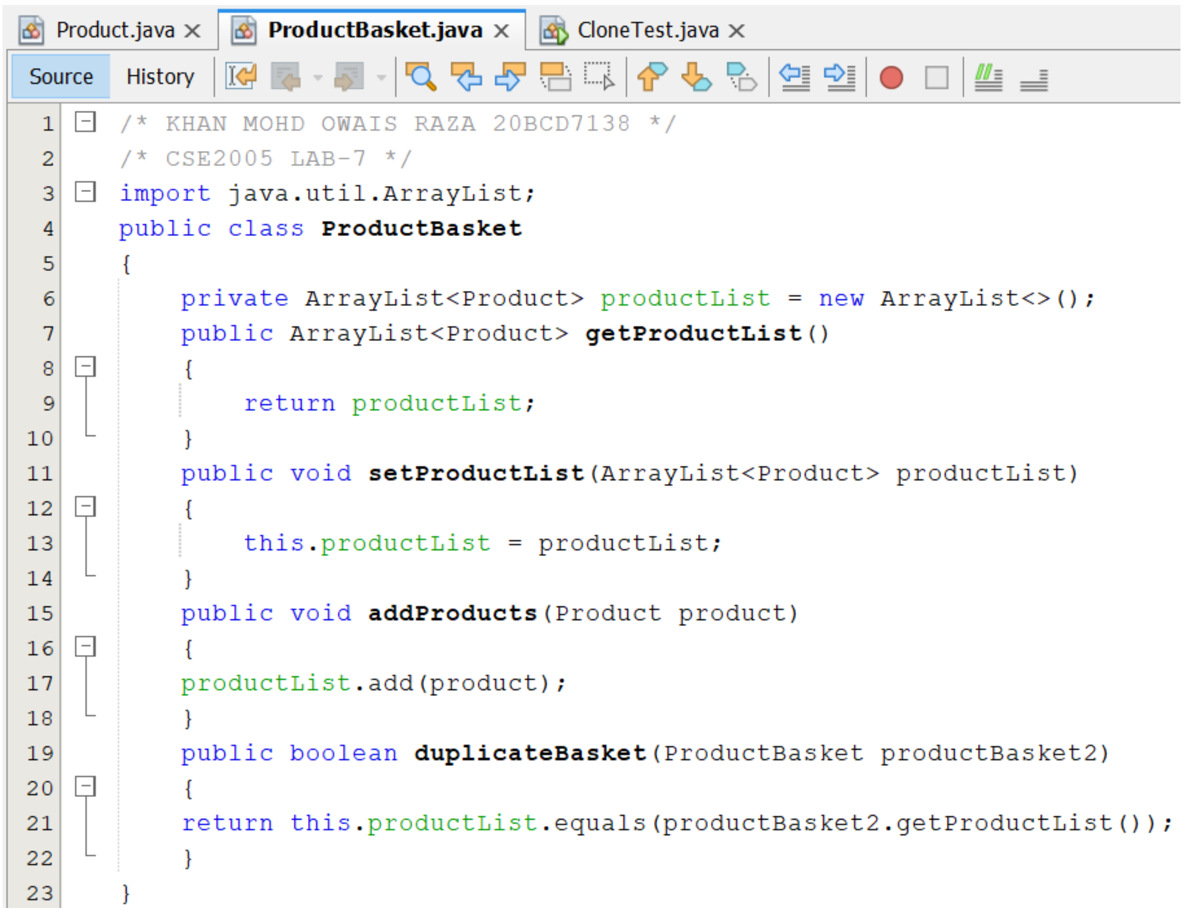
Write a CloneTest class and check if the applied cloning works or not with two objects.

Product.java –



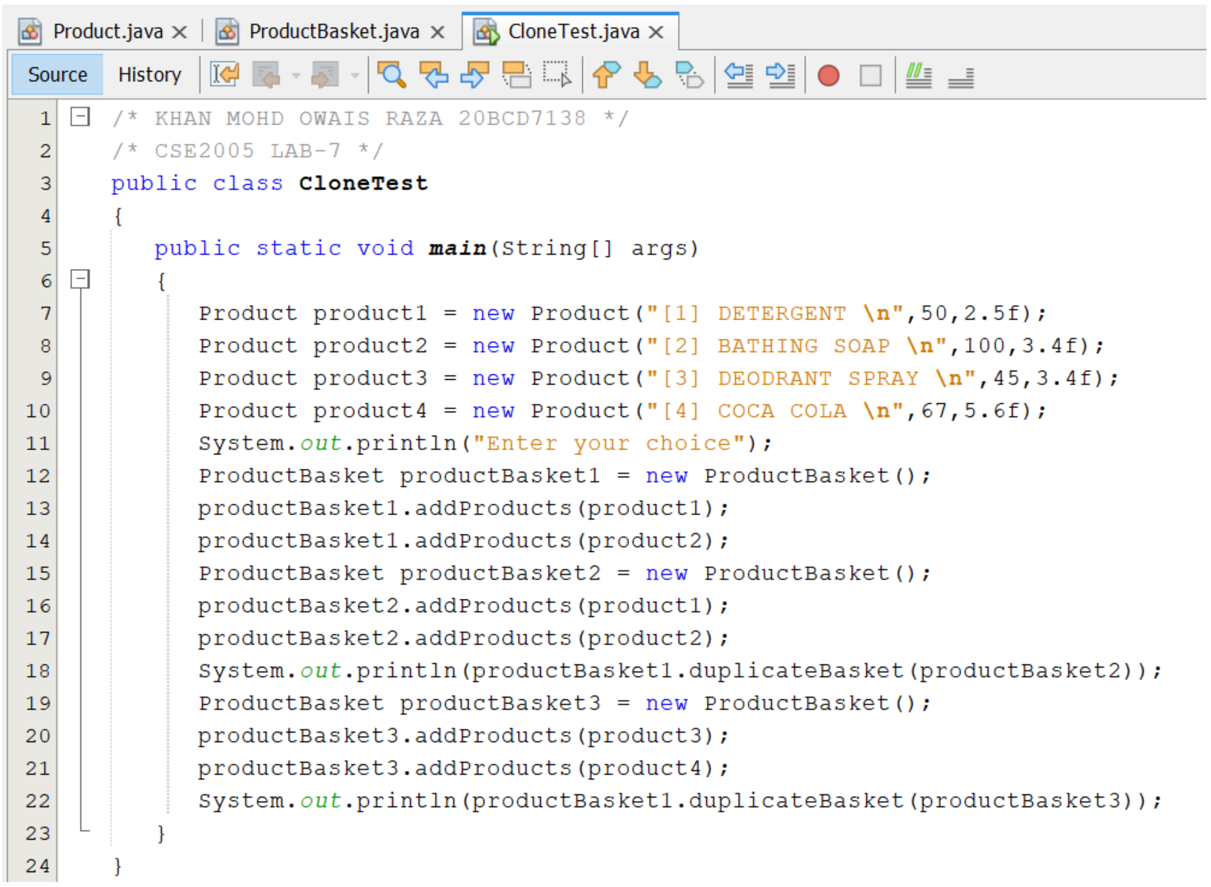
```java
/* KHAN MOHD OWAIS RAZA 20BCD7138 */
/* CSE2005 LAB-7 */
public class Product {
    private String name;
    private int price;
    private double discount;
    public Product(String name, int price, double discount){
        this.name = name;
        this.price = price;
        this.discount = discount;
    }
    public String getName(){
        return name;
    }
    public void setName(String name){
        this.name = name;
    }
    public int getPrice() {
        return price;
    }
    public void setPrice(int price) {
        this.price = price;
    }
    public double getDiscount(){
        return discount;
    }
    public void setDiscount(double discount){
        this.discount = discount;
    }
}
```

ProductBasket.java –



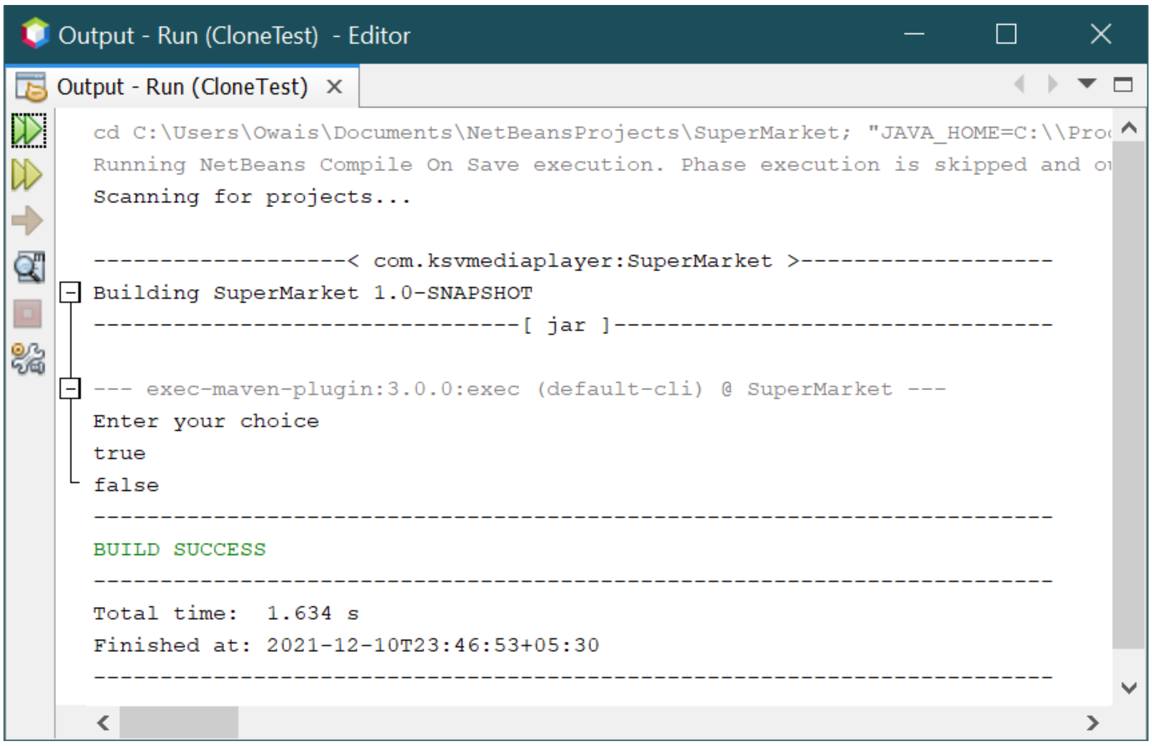
```
/* KHAN MOHD OWAIS RAZA 20BCD7138 */
/* CSE2005 LAB-7 */
import java.util.ArrayList;
public class ProductBasket
{
    private ArrayList<Product> productList = new ArrayList<>();
    public ArrayList<Product> getProductList()
    {
        return productList;
    }
    public void setProductList(ArrayList<Product> productList)
    {
        this.productList = productList;
    }
    public void addProducts(Product product)
    {
    productList.add(product);
    }
    public boolean duplicateBasket(ProductBasket productBasket2)
    {
    return this.productList.equals(productBasket2.getProductList());
    }
}
```

CloneTest.java –



```
/* KHAN MOHD OWAIS RAZA 20BCD7138 */
/* CSE2005 LAB-7 */
public class CloneTest
{
    public static void main(String[] args)
    {
        Product product1 = new Product("[1] DETERGENT \n",50,2.5f);
        Product product2 = new Product("[2] BATHING SOAP \n",100,3.4f);
        Product product3 = new Product("[3] DEODRANT SPRAY \n",45,3.4f);
        Product product4 = new Product("[4] COCA COLA \n",67,5.6f);
        System.out.println("Enter your choice");
        ProductBasket productBasket1 = new ProductBasket();
        productBasket1.addProducts(product1);
        productBasket1.addProducts(product2);
        ProductBasket productBasket2 = new ProductBasket();
        productBasket2.addProducts(product1);
        productBasket2.addProducts(product2);
        System.out.println(productBasket1.duplicateBasket(productBasket2));
        ProductBasket productBasket3 = new ProductBasket();
        productBasket3.addProducts(product3);
        productBasket3.addProducts(product4);
        System.out.println(productBasket1.duplicateBasket(productBasket3));
    }
}
```

Output console of CloneTest class to check the cloning –



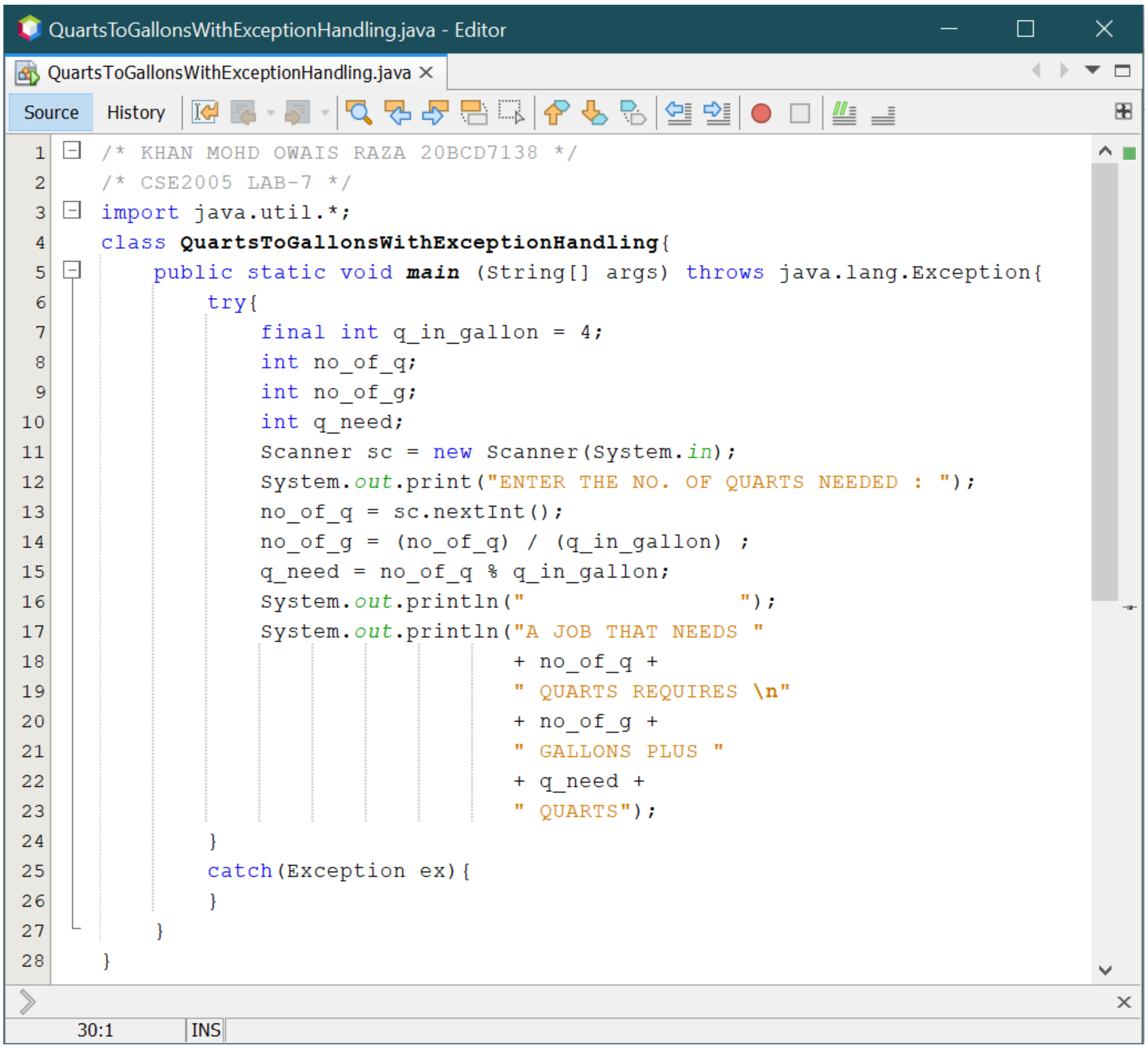
```
cd C:\Users\Owais\Documents\NetBeansProjects\SuperMarket; "JAVA_HOME=C:\\Pro
Running NetBeans Compile On Save execution. Phase execution is skipped and o
Scanning for projects...

-------------------< com.ksvmediaplayer:SuperMarket >-------------------
Building SuperMarket 1.0-SNAPSHOT
--------------------------------[ jar ]---------------------------------

--- exec-maven-plugin:3.0.0:exec (default-cli) @ SuperMarket ---
Enter your choice
true
false
------------------------------------------------------------------------
BUILD SUCCESS
------------------------------------------------------------------------
Total time:  1.634 s
Finished at: 2021-12-10T23:46:53+05:30
------------------------------------------------------------------------
```

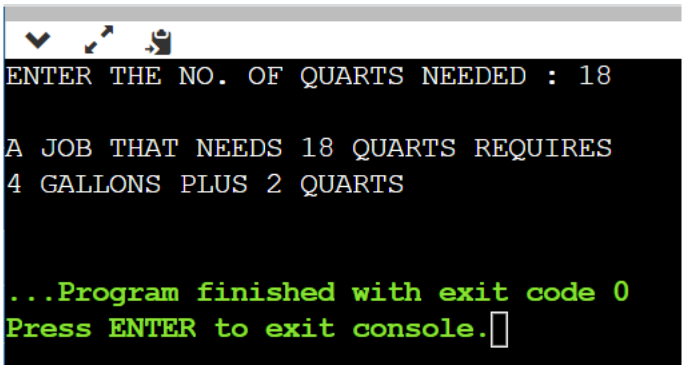
Q.2] Write a program that declares a named constant to hold the number of quarts in a gallon (4). Also declare a variable to represent the number of quarts needed for a painting job, and assign an appropriate value for example, 18 (default value). Compute and display the number of gallons and quarts needed for the job. Display explanatory text with the values for example, a job that needs 18 quarts requires 4 gallons plus 2 quarts. Instead of assigning a value to the number of quarts, accept the value from the user as input. Now, add exception-handling capabilities to this program and continuously re-prompt the user while any non-numeric value is entered. Save the file as QuartsToGallonsWithExceptionHandling.java.

QuartsToGallonsWithExceptionHandling.java –


QuartsToGallonsWithExceptionHandling.java - Editor
QuartsToGallonsWithExceptionHandling.java
Source
History
30:1
INS

```java
/* KHAN MOHD OWAIS RAZA 20BCD7138 */
/* CSE2005 LAB-7 */
import java.util.*;
class QuartsToGallonsWithExceptionHandling{
    public static void main (String[] args) throws java.lang.Exception{
        try{
            final int q_in_gallon = 4;
            int no_of_q;
            int no_of_g;
            int q_need;
            Scanner sc = new Scanner(System.in);
            System.out.print("ENTER THE NO. OF QUARTS NEEDED : ");
            no_of_q = sc.nextInt();
            no_of_g = (no_of_q) / (q_in_gallon) ;
            q_need = no_of_q % q_in_gallon;
            System.out.println("                ");
            System.out.println("A JOB THAT NEEDS "
                                   + no_of_q +
                                   " QUARTS REQUIRES \n"
                                   + no_of_g +
                                   " GALLONS PLUS "
                                   + q_need +
                                   " QUARTS");
        }
        catch(Exception ex){
        }
    }
}
```

Output console –



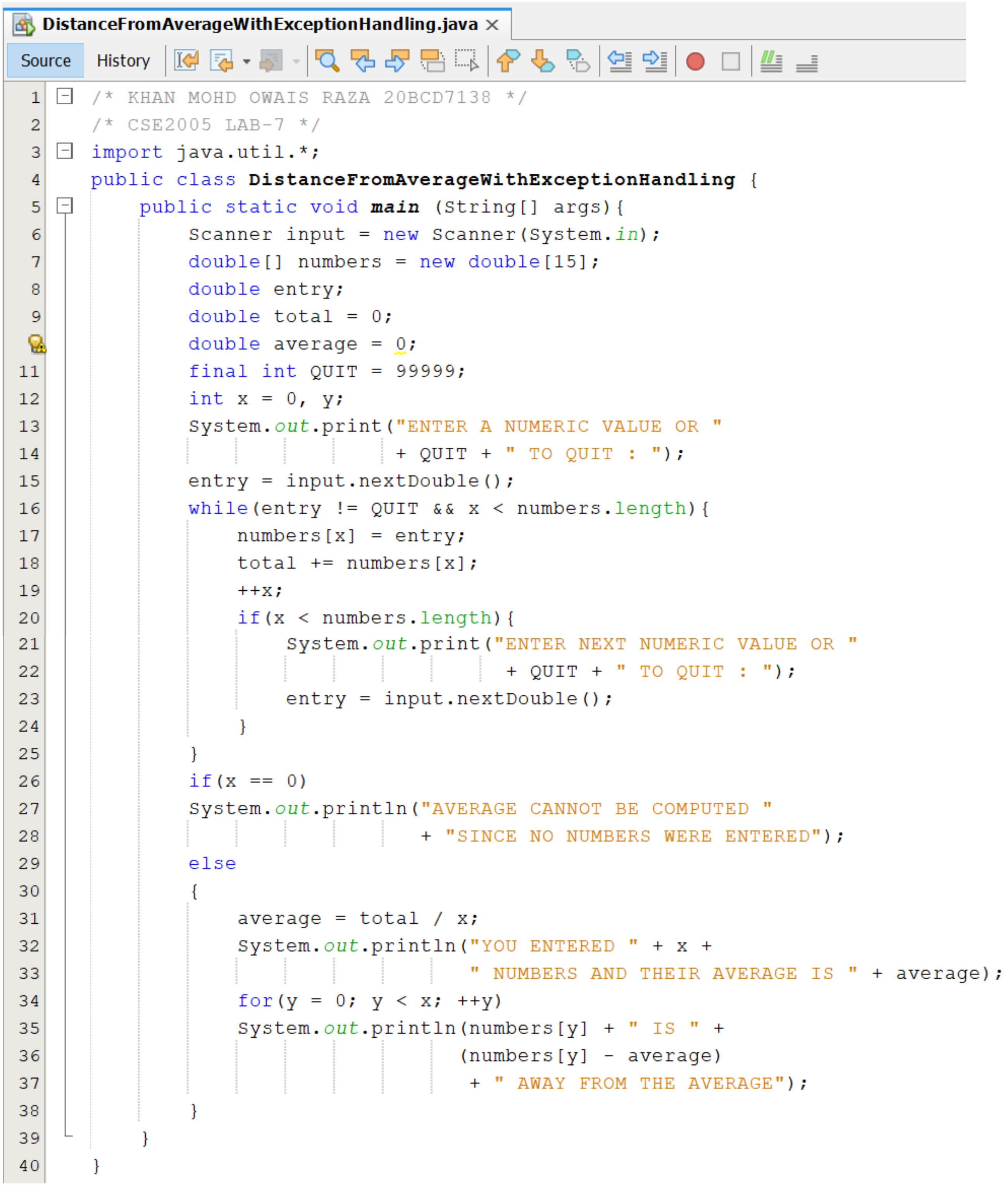
```
ENTER THE NO. OF QUARTS NEEDED : 18

A JOB THAT NEEDS 18 QUARTS REQUIRES
4 GALLONS PLUS 2 QUARTS


...Program finished with exit code 0
Press ENTER to exit console.
```

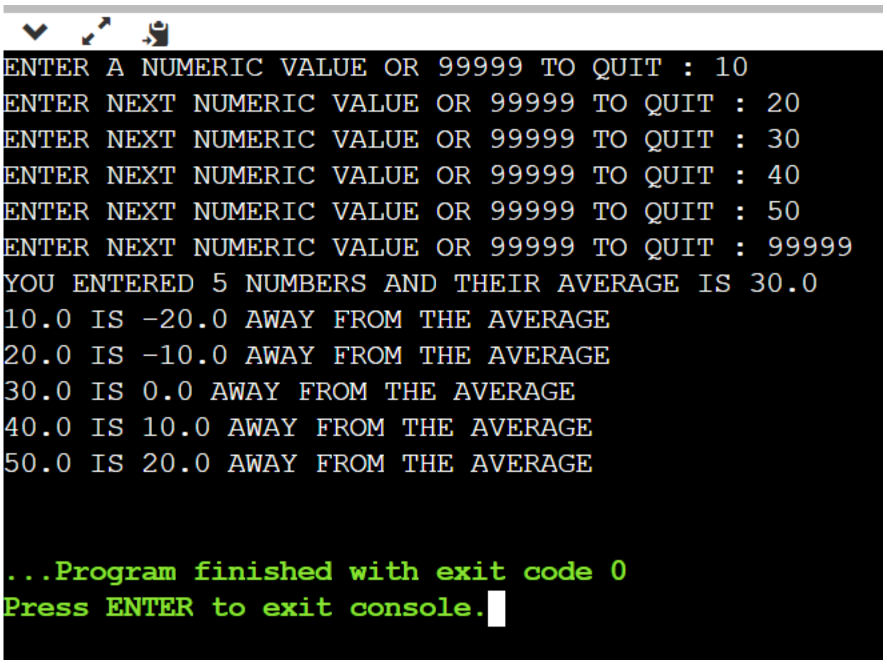
Q.3] Allow a user to enter an integer to declare an array of double with specific size. Java generates a NumberFormatException if you attempt to enter a noninteger value; handle this exception by displaying an appropriate error message. Create an array using the integer entered as the size. Java generates a NegativeArraySizeException if you attempt to create an array with a negative size; handle this exception by setting the array size to a default value of five. If the array is created successfully, use exception-handling techniques to ensure that each entered array value is a double or not. If not re-prompt the user to take proper input with suitable message. The program should display each entered value and its distance from the average.
Save the file as DistanceFromAverageWithExceptionHandling.java .

DistanceFromAverageWithExceptionHandling.java : –



```java
/* KHAN MOHD OWAIS RAZA 20BCD7138 */
/* CSE2005 LAB-7 */
import java.util.*;
public class DistanceFromAverageWithExceptionHandling {
    public static void main (String[] args){
        Scanner input = new Scanner(System.in);
        double[] numbers = new double[15];
        double entry;
        double total = 0;
        double average = 0;
        final int QUIT = 99999;
        int x = 0, y;
        System.out.print("ENTER A NUMERIC VALUE OR "
                         + QUIT + " TO QUIT : ");
        entry = input.nextDouble();
        while(entry != QUIT && x < numbers.length){
            numbers[x] = entry;
            total += numbers[x];
            ++x;
            if(x < numbers.length){
                System.out.print("ENTER NEXT NUMERIC VALUE OR "
                                 + QUIT + " TO QUIT : ");
                entry = input.nextDouble();
            }
        }
        if(x == 0)
        System.out.println("AVERAGE CANNOT BE COMPUTED "
                           + "SINCE NO NUMBERS WERE ENTERED");
        else
        {
            average = total / x;
            System.out.println("YOU ENTERED " + x +
                               " NUMBERS AND THEIR AVERAGE IS " + average);
            for(y = 0; y < x; ++y)
            System.out.println(numbers[y] + " IS " +
                               (numbers[y] - average)
                               + " AWAY FROM THE AVERAGE");
        }
    }
}
```

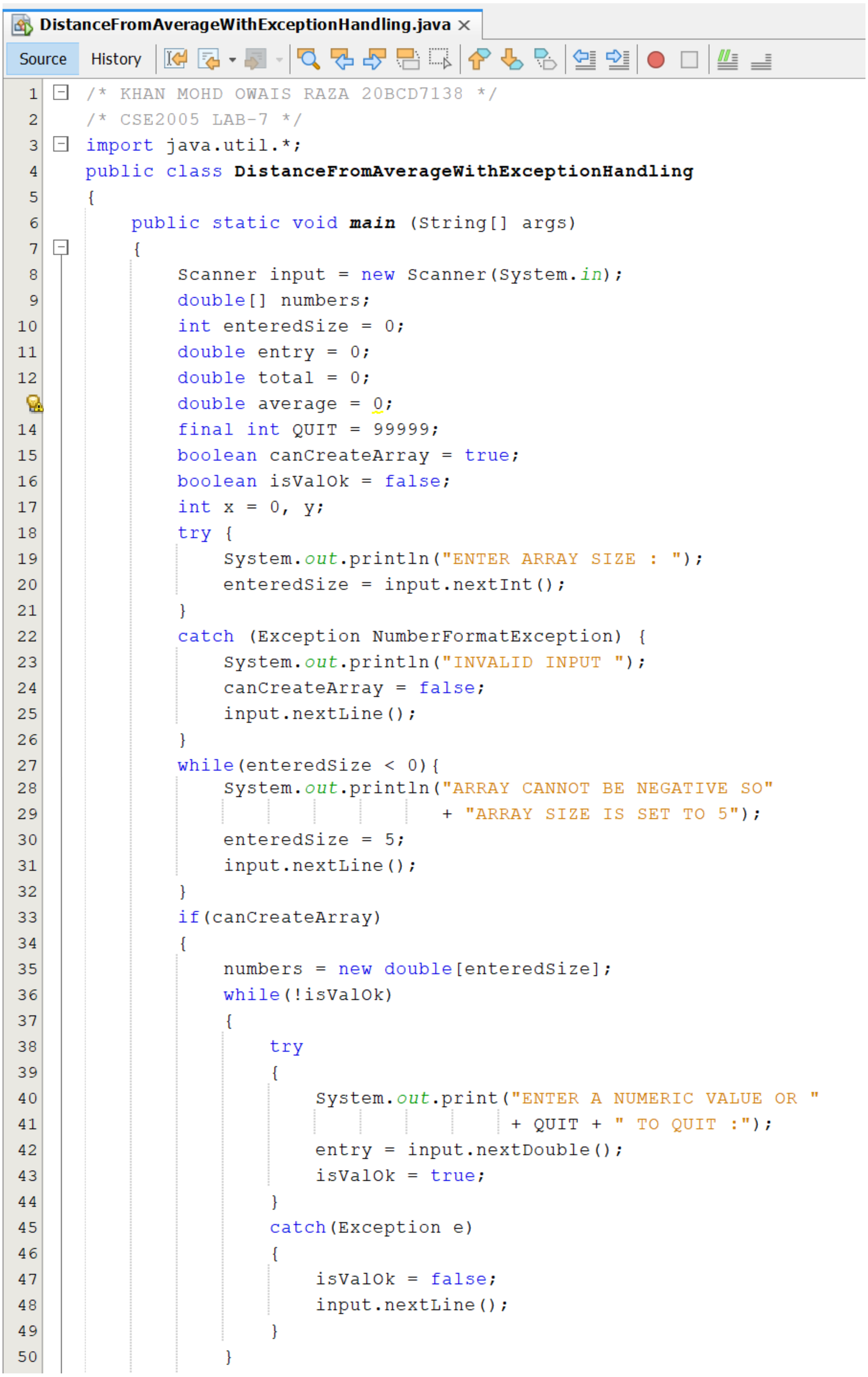
Output console –



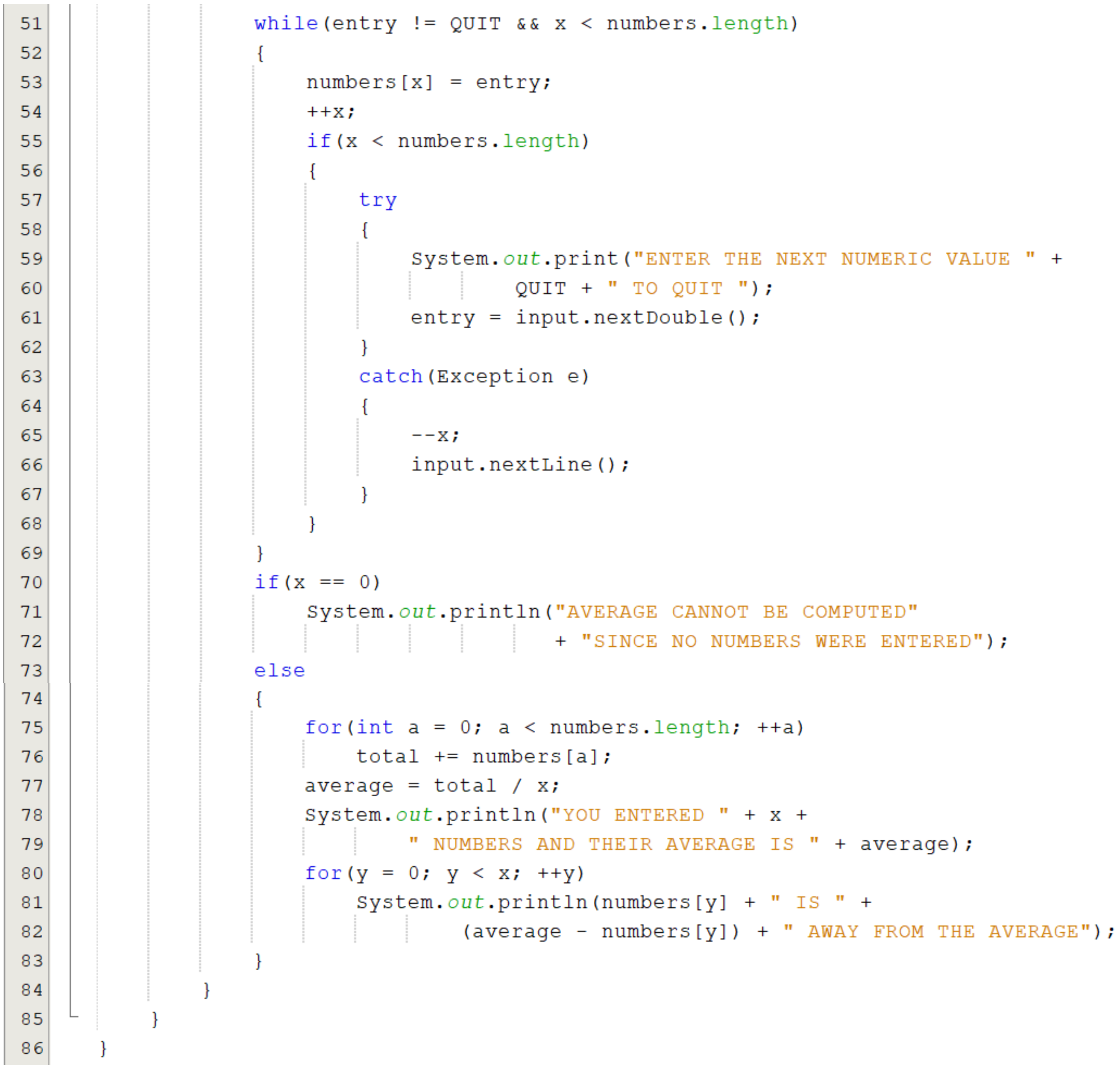
```
ENTER A NUMERIC VALUE OR 99999 TO QUIT : 10
ENTER NEXT NUMERIC VALUE OR 99999 TO QUIT : 20
ENTER NEXT NUMERIC VALUE OR 99999 TO QUIT : 30
ENTER NEXT NUMERIC VALUE OR 99999 TO QUIT : 40
ENTER NEXT NUMERIC VALUE OR 99999 TO QUIT : 50
ENTER NEXT NUMERIC VALUE OR 99999 TO QUIT : 99999
YOU ENTERED 5 NUMBERS AND THEIR AVERAGE IS 30.0
10.0 IS -20.0 AWAY FROM THE AVERAGE
20.0 IS -10.0 AWAY FROM THE AVERAGE
30.0 IS 0.0 AWAY FROM THE AVERAGE
40.0 IS 10.0 AWAY FROM THE AVERAGE
50.0 IS 20.0 AWAY FROM THE AVERAGE


...Program finished with exit code 0
Press ENTER to exit console.
```

Using another technique (assigning value to array ) –

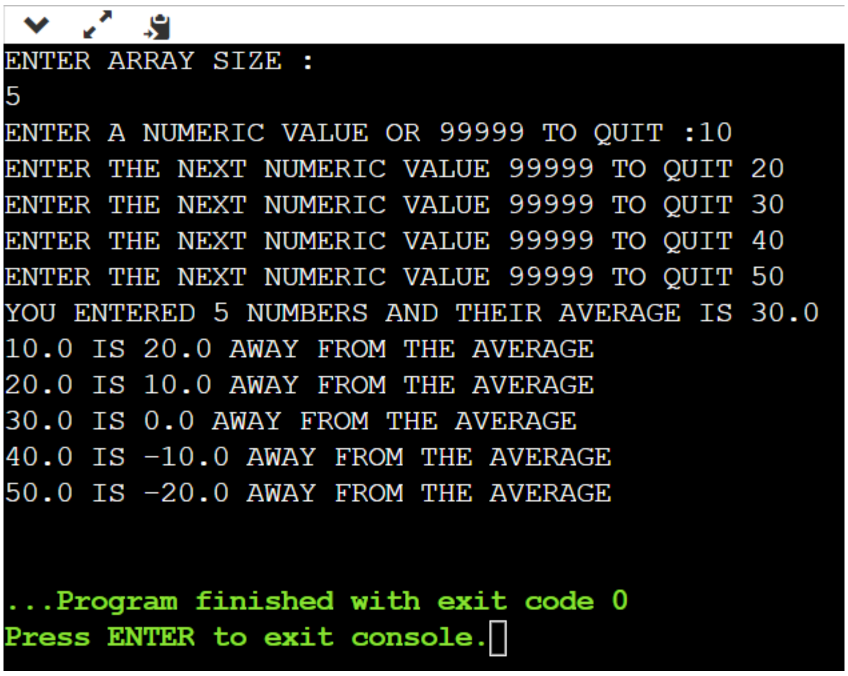


```java
/* KHAN MOHD OWAIS RAZA 20BCD7138 */
/* CSE2005 LAB-7 */
import java.util.*;
public class DistanceFromAverageWithExceptionHandling
{
    public static void main (String[] args)
    {
        Scanner input = new Scanner(System.in);
        double[] numbers;
        int enteredSize = 0;
        double entry = 0;
        double total = 0;
        double average = 0;
        final int QUIT = 99999;
        boolean canCreateArray = true;
        boolean isValOk = false;
        int x = 0, y;
        try {
            System.out.println("ENTER ARRAY SIZE : ");
            enteredSize = input.nextInt();
        }
        catch (Exception NumberFormatException) {
            System.out.println("INVALID INPUT ");
            canCreateArray = false;
            input.nextLine();
        }
        while(enteredSize < 0){
            System.out.println("ARRAY CANNOT BE NEGATIVE SO"
                                    + "ARRAY SIZE IS SET TO 5");
            enteredSize = 5;
            input.nextLine();
        }
        if(canCreateArray)
        {
            numbers = new double[enteredSize];
            while(!isValOk)
            {
                try
                {
                    System.out.print("ENTER A NUMERIC VALUE OR "
                                    + QUIT + " TO QUIT :");
                    entry = input.nextDouble();
                    isValOk = true;
                }
                catch(Exception e)
                {
                    isValOk = false;
                    input.nextLine();
                }
            }
```



```
            while(entry != QUIT && x < numbers.length)
            {
                numbers[x] = entry;
                ++x;
                if(x < numbers.length)
                {
                    try
                    {
                        System.out.print("ENTER THE NEXT NUMERIC VALUE " +
                                QUIT + " TO QUIT ");
                        entry = input.nextDouble();
                    }
                    catch(Exception e)
                    {
                        --x;
                        input.nextLine();
                    }
                }
            }
            if(x == 0)
                System.out.println("AVERAGE CANNOT BE COMPUTED"
                                   + "SINCE NO NUMBERS WERE ENTERED");
            else
            {
                for(int a = 0; a < numbers.length; ++a)
                    total += numbers[a];
                average = total / x;
                System.out.println("YOU ENTERED " + x +
                        " NUMBERS AND THEIR AVERAGE IS " + average);
                for(y = 0; y < x; ++y)
                    System.out.println(numbers[y] + " IS " +
                            (average - numbers[y]) + " AWAY FROM THE AVERAGE");
            }
        }
    }
}
```

Output console –



```
ENTER ARRAY SIZE :
5
ENTER A NUMERIC VALUE OR 99999 TO QUIT :10
ENTER THE NEXT NUMERIC VALUE 99999 TO QUIT 20
ENTER THE NEXT NUMERIC VALUE 99999 TO QUIT 30
ENTER THE NEXT NUMERIC VALUE 99999 TO QUIT 40
ENTER THE NEXT NUMERIC VALUE 99999 TO QUIT 50
YOU ENTERED 5 NUMBERS AND THEIR AVERAGE IS 30.0
10.0 IS 20.0 AWAY FROM THE AVERAGE
20.0 IS 10.0 AWAY FROM THE AVERAGE
30.0 IS 0.0 AWAY FROM THE AVERAGE
40.0 IS -10.0 AWAY FROM THE AVERAGE
50.0 IS -20.0 AWAY FROM THE AVERAGE


...Program finished with exit code 0
Press ENTER to exit console.
```