# Oops
# Lab 7

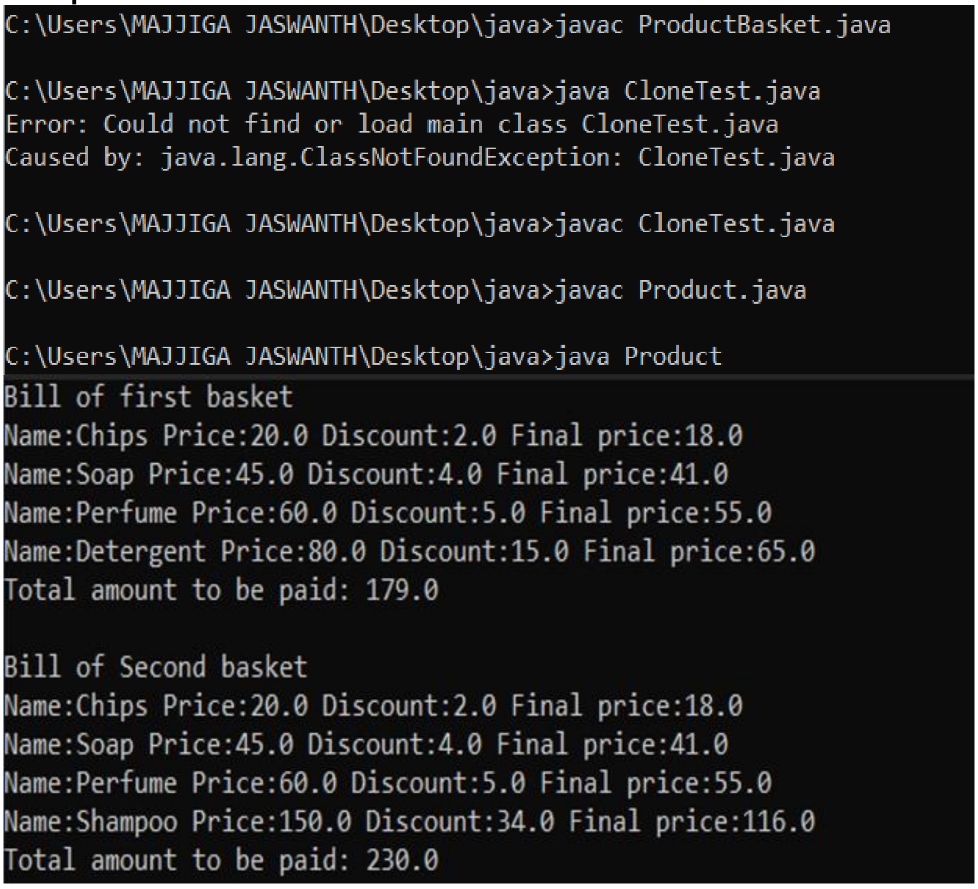
MAJJIGA JASWANTH
20BCD7171

1)
Consider a super market 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 reentering the products of second basket for bill generation. Apply suitable Object Cloning to generate different bills even if few items are deleted I n 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.
Code:
```
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()
);
    }

}
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()
);
    }

}
public class CloneTest {

    public static void main(String[] args) {

        Product product1 = new
Product("Watch",50,2.5f);
        Product product2 = new
Product("Lamp",100,3.4f);
        Product product3 = new
Product("Teddy",45,3.4f);
```

```
        Product product4 = new
Product("Frame",67,5.6f);

        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(produ
ctBasket2));

        ProductBasket productBasket3 = new
ProductBasket();
        productBasket3.addProducts(product3);
        productBasket3.addProducts(product4);


System.out.println(productBasket1.duplicateBasket(produ
ctBasket3));

    }
}

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()
);
    }

}
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;
    }
```

```
}
```

Output:



```
C:\Users\MAJJIGA JASWANTH\Desktop\java>javac ProductBasket.java

C:\Users\MAJJIGA JASWANTH\Desktop\java>java CloneTest.java
Error: Could not find or load main class CloneTest.java
Caused by: java.lang.ClassNotFoundException: CloneTest.java

C:\Users\MAJJIGA JASWANTH\Desktop\java>javac CloneTest.java

C:\Users\MAJJIGA JASWANTH\Desktop\java>javac Product.java

C:\Users\MAJJIGA JASWANTH\Desktop\java>java Product
Bill of first basket
Name:Chips Price:20.0 Discount:2.0 Final price:18.0
Name:Soap Price:45.0 Discount:4.0 Final price:41.0
Name:Perfume Price:60.0 Discount:5.0 Final price:55.0
Name:Detergent Price:80.0 Discount:15.0 Final price:65.0
Total amount to be paid: 179.0

Bill of Second basket
Name:Chips Price:20.0 Discount:2.0 Final price:18.0
Name:Soap Price:45.0 Discount:4.0 Final price:41.0
Name:Perfume Price:60.0 Discount:5.0 Final price:55.0
Name:Shampoo Price:150.0 Discount:34.0 Final price:116.0
Total amount to be paid: 230.0
```

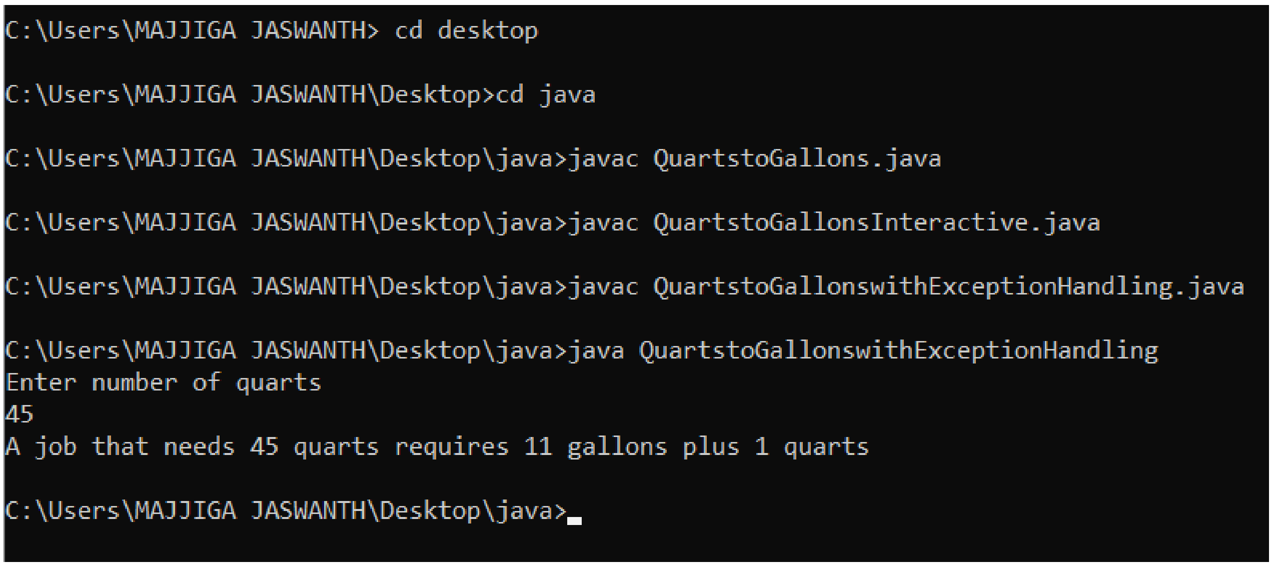
2.
Write a program that declares a named constant to hold the number of quarts in a gallon (4). Also declare a variable to represent the number of quarts needed for a painting job, and assign an appropriate value—for example, 18. Compute and display the number of gallons and quarts needed for the job.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

Code:

```
public class QuartstoGallons {
     public static void main(String args[])
     {
          final int
          quarts=4;
          int
          noofquart
          s=39;
          System.out.println("A job that needs
"+noofquarts+" quarts requires "+(noofquarts/quarts)+"
gallons plus "+(noofquarts%quarts)+" quarts");
     }
}

import java.util.*;
public class QuartstoGallonsInteractive
{
 public static void main(String args[])
 {
Scanner sc=new
Scanner(System.in);
final int quarts=18;
  System.out.println("Enter the
  number of quarts : "); int
  noofquarts=sc.nextInt();
  System.out.println("A job that needs "+noofquarts+"
quarts requires "+(noofquarts/quarts)+" gallons plus
"+(noofquarts%quarts)+" quarts");
```

```
 }
}

import java.util.Scanner;
class QuartstoGallonswithExceptionHandling
{
 public static void main(String[] args)
 {
   Scanner input = new
   Scanner(System.in);
   final int quarts = 4;
   int
  noofqua
  rts;
  int
  i=0;
  while(i
  ==0)
  {
    System.out.println("Enter
    number of quarts"); try
    {
     noofquarts =
     Integer.parseInt(input.nex
     tLine()); int gallons =
     noofquarts/quarts;
     System.out.println("A job that needs
"+noofquarts+" quarts requires "+gallons+" gallons
plus "+(noofquarts%quarts)+" quarts");
     i=1;
    }
    catch(Exception e)
    {
     System.out.println("Exception: " + "
     NumberFormatException");
    }

  }
 }
}
```

Output:



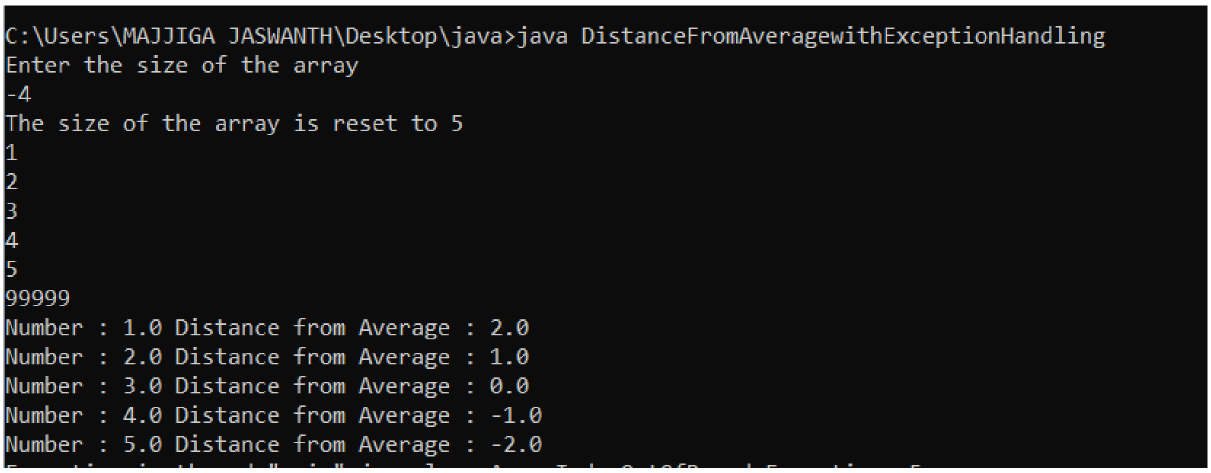
```
C:\Users\MAJJIGA JASWANTH> cd desktop

C:\Users\MAJJIGA JASWANTH\Desktop>cd java

C:\Users\MAJJIGA JASWANTH\Desktop\java>javac QuartstoGallons.java

C:\Users\MAJJIGA JASWANTH\Desktop\java>javac QuartstoGallonsInteractive.java

C:\Users\MAJJIGA JASWANTH\Desktop\java>javac QuartstoGallonswithExceptionHandling.java

C:\Users\MAJJIGA JASWANTH\Desktop\java>java QuartstoGallonswithExceptionHandling
Enter number of quarts
45
A job that needs 45 quarts requires 11 gallons plus 1 quarts

C:\Users\MAJJIGA JASWANTH\Desktop\java>
```

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 nonintegervalue; handle this exception by displaying an appropriateerror message. Create an array using the integer entered as the size. Java generates a NegativeArraySizeException if you attempt to create an arraywith a negative size; handle this exception by setting the array size to a defaultvalue of five.If the array is created successfully, use exception-handlingtechniques 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 itsdistance from the average. Save the file asDistanceFromAverageWithExceptionHandling.java.

Code:
```
import java.util.*;
public class DistanceFromAveragewithExceptionHandling
{
 public static void main(String args[])
 {
  Scanner sc=new
  Scanner(System.in);
  System.out.println("Enter
  the size of the array");
  int n;
  doub
  le
  a[];
  try
  {
   n=Integer.parseInt(sc.nextLine());
   try
   {
    a=new double[n];
   }
   catch(NegativeArraySizeException w)
   {
     a=new
    double[5
    ]; n=5;
    System.out.println("The size of the array is reset
    to "+n);
  }
double
s=0,count=
0; for(int
i=0;i<=n;i
++)
{
  double
  k=sc.nextDoub
  le();
  if(k==99999)
  {
    break;
  }
  else
  {
    a[i]=k;
    count=c
    ount+1;
    s=s+a[i
    ];
  }
}
try
{
  if(s==0)
  {
    throw new NotenteredanyException();
  }
  else
  {
    double
    avg=s/coun
    t; for(int
    i=0;i<15;i
    ++)
    {
      if(a[i]!=0)
      {
        System.out.println("Number : "+a[i]+" Distance
        from Average : "+(avg-a[i]));
      }
      else
      {
        System.out.println("Furthur values are
        not given by the user"); break;
      }
```

```
        }
      }
    }
    catch(NotenteredanyException e)
    {
     System.out.println(e);
    }
    }
    catch(NumberFormatException n1)
    {
     System.out.println("Input should be an integer");
    }
  }
}
class NotenteredanyException extends Exception
{
 public String toString()
 {
   return "Error : Please enter atleast one element in
   the array other than 99999";
 }
}
```

Output:



```
C:\Users\MAJJIGA JASWANTH\Desktop\java>java DistanceFromAveragewithExceptionHandling
Enter the size of the array
-4
The size of the array is reset to 5
1
2
3
4
5
99999
Number : 1.0 Distance from Average : 2.0
Number : 2.0 Distance from Average : 1.0
Number : 3.0 Distance from Average : 0.0
Number : 4.0 Distance from Average : -1.0
Number : 5.0 Distance from Average : -2.0
```

--------------------------------------------------------------

END