CSC 151 Assignment #1

1. Honor Code

A. For individual assignments: Jane Doe and John Doe will be replaced by your full name(s)
I affirm that I have carried out my academic endeavors with full academic honesty.
[Signed, Manav Bilakhia]

Resources:
Textbook
Geeksforgeeks

2. Java files and outputs

A. Java files

Class: Product.java

```java
/*
 * I affirm that I have carried out the attached academic endeavors with full academic
honesty.
 * Manav Bilakhia (MB)
 */
package assignment;
/**
 * Product class for your receipts.
 * Each Product has a name, unit price and quantity
 * Total cost is calculated as unit price x quantity
 *
 * @Manav Bilakhia

 */
public class Product
{

    //Limit values as constants
    public static final int MAX_UNIT_PRICE = 100;
    public static final int MIN_UNIT_PRICE = 0;
    public static final int MAX_QUANTITY = 10;
    public static final int MIN_QUANTITY = 0;

    /**
     * Instance variables for unit price, quantity, product name
     */
    private int unitPrice;
    private int quantity;
    private String pName;

    /**
     * Constructor with no parameter
     * int instance variables are set to 0
     * String instance variable is set to NO NAME
     */
    public Product()
    {
        unitPrice = 0;
        quantity = 0;
        pName = "NO NAME";
    }
    /**
     * Constructor with 3 parameters
     * @param unitPrice initial unit price
     * @param quantity initial quantity
     * @param pName initial name of the product
     */
    public Product(int unitPrice,int quantity,String pName)
    {
        this.setUnitPrice(unitPrice);
        this.setQuantity(quantity);
        this.setPName(pName);
    }
    /**
     * get method
     * @return unit price as integer
     */
    public int getUnitPrice()
    {
        return unitPrice;
    }


    /**
     * set method
     * @param unitPrice to set
     */
    public void setUnitPrice(int unitPrice)
    {
        if (unitPrice>MAX_UNIT_PRICE||unitPrice<MIN_UNIT_PRICE)
        {
            this.unitPrice = 0;
        }
        else
            this.unitPrice = unitPrice;
    }
    /**
     * get method
     * @return quantity as integer
     */
    public int getQuantity()
    {
        return quantity;
    }

    /**
     * set method
     * @param quantity to set
     */
    public void setQuantity(int quantity)
    {
        if (quantity>MAX_QUANTITY||quantity<MIN_QUANTITY)
        {
            this.quantity = 0;
        }
        else
            this.quantity = quantity;
    }

    /**
    * get method
    * @return product name as String
    */
    public String getPName()
    {
        return pName;
    }

    /**
     * set method
     * @param pName the pName to set
     */
    public void setPName(String pName)
    {
        this.pName = pName;
    }

    /**
     * Override equals method
     * @param obj second product for comaprison
     * @return
     */
    @Override
    public boolean equals(Object obj) {
        if (this == obj)
            return true;
        if (obj == null)
            return false;
        if (getClass() != obj.getClass())
            return false;
        Product other = (Product) obj;
        if (pName == null){
            if (other.pName !=null)
                return false;
        } else if (!pName.equals(other.pName))
            return false;
        if (quantity != other.quantity)
            return false;
        if (unitPrice != other.unitPrice)
            return false;
        return true;
    }

    /**
     * total method for calculating total
     * @return total
     */
    public int total()
    {
        return this.getQuantity()*this.getUnitPrice();
    }

    /**
     * override tostring method
     * @return string
     */
    @Override
    public String toString()
    {
        return this.getPName()+":"+this.getUnitPrice()+" x "+this.getQuantity()+ " = "+
total();
    }

    /**
     * main method for testing,
     * @param args default java main parameter
     */
```

```java
    public static void main(String[] args)
    {
        Product p1=new Product();
        Product p2=new Product(100,10,"Chocolate");
        Product p3=new Product(100,10,"Chocolate");
        Product p4=new Product(0,0,"shampoo");
        Product p5=new Product(10,5,"soap");
        int total=p1.total()+p2.total()+p4.total()+ p5.total();
        System.out.println(p1);
        System.out.println(p2);
        System.out.println(p4);
        System.out.println(p5);
        System.out.println(total);
        System.out.println(p3.equals(p4));
        System.out.println(p3.equals(p2));
    }
}
```

Class: Receipt.java

```java
/*
 * I affirm that I have carried out the attached academic endeavors with full academic
honesty.
 * Manav Bilakhia (MB)
 */
package assignment;

/**
 * Receipt class for your receipts of Products.
 * Each Receipt has a collection of Product objects kept as array
 * Total cost is calculated as unit price x quantity for all
 * Product instances in the array
 *
 * @Manav Bilakhia
 */
public class Receipt
{
    /**
     * Limit value for max items as a constant
     */
    public static final int MAX_ITEMS = 100;
    /**
     * private instance variables for receipt as array and item count
     */
    private int itemCount;
    private Product [] receipt;

    /**
     * Constructor with no parameter
     */
    public Receipt()
    {
        receipt = new Product[MAX_ITEMS];
        itemCount = 0;
    }

    /**
     * addItem method for adding a Product to the array
     * @param product product to be added
     * @return itemCount
     */
    public int addItem(Product product)
    {
        if (itemCount<100) {
            receipt[itemCount] = product;
            itemCount++;
        }
        return itemCount;
    }

    /**
     * calcTotal method for calculating the total cost of items
     * @return total
     */
    public int calcTotal()
    {
        int total = 0;
        for(int i = 0; i< itemCount;i++)
        {
            total = total + receipt[i].total();
        }
        return total;
    }
    /**
     * override tostring method
     * @return string
     */
    @Override
    public String toString()
    {
        String str = "";
        for(int i = 0;i<itemCount;i++)
        {
            str = str + ""+ receipt[i] + "\n";
        }
        return str;
    }

    /**
     * main method for testing,
     * @param args default java main parameter
     */
    public static void main(String[] args)
    {
        // TODO Auto-generated method stub
        Receipt mR = new Receipt();
        mR.addItem(new Product(1,  3, "p1"));
        mR.addItem(new Product(70,  5, "p2"));
        mR.addItem(new Product(950,  20, "p3"));
        System.out.println(mR + " = " + mR.calcTotal());
    }
}
```

B. Sample output 1

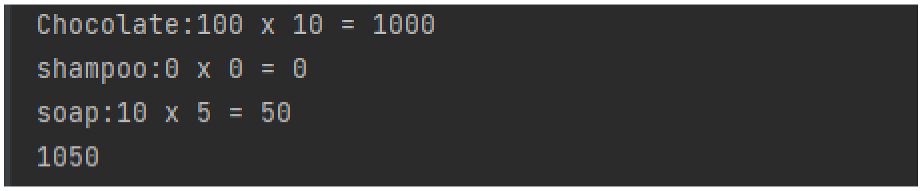
I. Describe your test 1:To see if total method in product class works

II. Text output 1:

```
Chocolate:100 x 10 = 1000
shampoo:0 x 0 = 0
soap:10 x 5 = 50
```

1050
false
true

III. Screenshot 1:



```
Chocolate:100 x 10 = 1000
shampoo:0 x 0 = 0
soap:10 x 5 = 50
1050
```

C. Sample output 2

I. Describe your test 2: to see if equals method in product class recognizes two same items with same name, same item price and same quantity

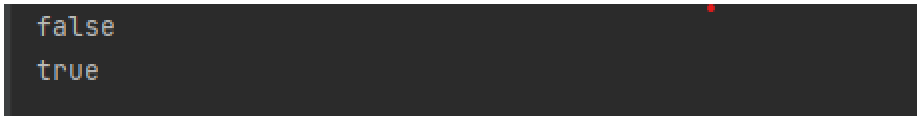
Test case:

```
Product p2=new Product(100,10,"Chocolate");
Product p3=new Product(100,10,"Chocolate");
System.out.println(p3.equals(p4));
System.out.println(p3.equals(p2));
```

II. Text output 2:

false
true

III. Screenshot 2:



```
false
true
```

D. Sample output 3

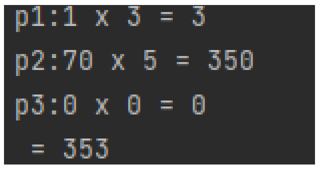
I. Describe your test 3: to see what happens when we try to input items beyond the max price and quantity in the receipt class

```
mR.addItem(new Product(1,  3, "p1"));
mR.addItem(new Product(70,  5, "p2"));
mR.addItem(new Product(950,  20, "p3"));
```

II. Text output 3:

p1:1 x 3 = 3
p2:70 x 5 = 350
p3:0 x 0 = 0
 = 353

III. Screenshot 3:



```
p1:1 x 3 = 3
p2:70 x 5 = 350
p3:0 x 0 = 0
 = 353
```

The code passed all the tests.