Day 1

2. Solve this:

5.1 Can you arrange Fruit,Apple,Orange,Mango in inheritance hierarchy ?

Use tight encapsulation.

5.2 Properties (instance variables) : color : String , weight : double , name:String, fresh : boolean

5.3 Add suitable constructors.

5.4 Override toString correctly to return state of all fruits (return only : name ,color , weight)

5.5 Add a taste() method : public String taste()

For Fruit : Can you identify taste of any general fruit ?

So will you add a taste() with this definition : returns "no specific taste" OR can u apply abstraction?

Apple : should return

Mango : should return "sweet"

Orange : should return "sour"

5.6 Add specific functionality , in the sub classes

In Mango : public void pulp() {Display name n color of the fruit + a mesg creating pulp!}

In Orange : public void juice() {Display name n weight of the fruit + a mesg extracting juice!}

In Apple : public void jam() {Display name of the fruit + a mesg making jam!}

5.7 Add all of above classes under the package "com.app.fruits"

5.8 Create java application FruitBasket , with main method , as a tester , in com.app.tester package.

5.9 Prompt user for the basket size n create suitable data structure

5.10 Supply options

1. Add Mango

2. Add Orange

3. Add Apple

NOTE : You will be adding a fresh fruit in the basket , in all of above options.

4. Display names of all fruits in the basket.

5. Display name,color,weight , taste of all fresh fruits , in the basket.

6. Mark a fruit in a basket , as stale(=not fresh)

i/p : index

o/p : error message (in case of invalid index) or mark it stale

7. Mark all sour fruits stale

Hint : Use equals() method of the String class.

8. Invoke fruit specific functionality (pulp / juice / jam)

i/p : index

Invoke correct functionality (pulp / juice / jam)

10. Exit

Code:

Fruit Class:

```
package com.app.fruits;

public abstract class Fruit {

	private String color;
	private double weight;
	private String name;
	private boolean fresh=true;


	public boolean isFresh() {
		return fresh;
	}

	public void setFresh(boolean fresh) {
		this.fresh = fresh;
	}
```

```
    public Fruit(String color, double weight, String name) {
        super();
        this.color = color;
        this.weight = weight;
        this.name = name;
    }

     abstract public String taste();


    public String getColor() {
        return color;
    }

    public void setColor(String color) {
        this.color = color;
    }

    public double getWeight() {
        return weight;
    }

    public void setWeight(double weight) {
        this.weight = weight;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    @Override
    public String toString() {
        return "Fruit [color=" + color + ", weight=" +
weight + ", name=" + name + ", fresh=" + fresh + "]";
```

```
    }

}
```

Apple Class:

```
package com.app.fruits;

public class Apple extends Fruit
{
    public Apple(String color, double weight, String name)
    {
        super(color,weight,name);

    }
    @Override
    public String taste()
    {
        return "sweet n sour";
    }
    public void jam()
    {
        System.out.println("name: "+getName()+ ", color:
"+getColor()+ "\nmaking jam....!" );
    }
}
```

Orange Class:

```
package com.app.fruits;

public class Orange extends Fruit {

	public Orange(String color, double weight, String name)
	{
		super(color,weight,name);

	}
	@Override
	public String taste()
	{
		return "sour";
	}

	public void juice()
	{
		System.out.println("name: "+getName()+ ", color:
"+getColor()+ "\nExtracting juice....!" );
	}
}
```

Mango Class:

```
package com.app.fruits;

public class Mango extends Fruit
{
	public Mango(String color, double weight, String name)
	{
		super(color,weight,name);

	}
	@Override
	public String taste()
	{
		return "sweet";
	}

	public void pulp()
	{
		System.out.println("name: "+getName()+ ", color:
"+getColor()+ "\nCreating Pulp....!" );
	}
}
```

FruiteBasket:

```
package com.app.tester;
import java.util.Scanner;

import com.app.fruits.*;
public class FruitBasket {

	public static void main(String[] args)
	{
		Scanner sc = new Scanner(System.in);
		System.out.println("Mention Basket size: ");
		Fruit[] f = new Fruit[sc.nextInt()];

		System.out.println("1. Add Mango\r\n"
				+ "2. Add Orange\r\n"
				+ "3. Add Apple\r\n"
				+ "4. Display names of fruits\r\n"
				+ "5. Display details and taste fresh
fruits in the basket\r\n"
				+ "6. Mark a fruit in a basket , as
stale(=not fresh\r\n"
				+ "7. Mark all sour fruits stale\r\n"
				+ "8. get fruit specific functionality
(pulp / juice / jam)");
		int ch=0;
		int count=0;
		do
		{
			System.out.println("Please enter your choice:
");
			ch=sc.nextInt();
			switch(ch)
```

```
                {
                case 1:
                     if(count<f.length)
                     {
                          System.out.println("Enter color
weight and name: ");
                          f[count++]=new
Mango(sc.next(),sc.nextDouble(),sc.next());
                     }
                     else
                     {
                          System.out.println("Basket is
Full...!");
                     }
                     break;
                case 2:
                     if(count<f.length)
                     {
                          System.out.println("Enter color
weight and name: ");
                          f[count++]=new
Orange(sc.next(),sc.nextDouble(),sc.next());
                     }
                     else
                     {
                          System.out.println("Basket is
Full...!");
                     }
                     break;
                case 3:
                     if(count<f.length)
                     {
                          System.out.println("Enter color
weight and name: ");
                          f[count++]=new
Apple(sc.next(),sc.nextDouble(),sc.next());
                     }
                     else
                     {
```

```
                    System.out.println("Basket is
Full...!");
               }
               break;
          case 4:
               for(Fruit e:f)
               {
                    if(e!=null)
                    {
                         System.out.println(e.getName());
                    }
               }
               break;
          case 5:
               for(Fruit e:f)
               {
                    if(e!=null)
                    {
                         System.out.println(e);
                    }
               }
               break;

          case 6:
               System.out.println("Enter fruit(index)
number to mark as a stale: ");

               int b=sc.nextInt();
               b-=1;
               if(f[b]!=null)
               {
                    f[b].setFresh(false);
               }
               else
                    System.out.println("invalid index");
               break;
          case 7:
               for(Fruit e:f)
               {
```

```
                    if(e!=null &&
e.taste().equals("sour"))
                    {
                         e.setFresh(false);
                    }
               }
               System.out.println("all Sour fruits now
stale...!");
               break;

          case 8:
               System.out.println("Enter fruit(index)
number to call specific functionality: ");
               b=sc.nextInt();
               b-=1;
               if(f[b]!=null)
               {
                    if(f[b] instanceof Apple)
                    {
                         ((Apple)f[b]).jam();
                    }
                    else if(f[b] instanceof Orange)
                    {
                         ((Orange)f[b]).juice();
                    }
                    else if(f[b] instanceof Mango)
                    {
                         ((Mango)f[b]).pulp();
                    }

               }
               else
               {
                    System.out.println("not to much fruit
available...!");
               }
               break;

               default:
```

```
                        System.out.println("Invalid
choice.....!");
                        break;
                }
            }while(ch<9);


    }

}
```

Output:

```
Mention Basket size:
5
1. Add Mango
2. Add Orange
3. Add Apple
4. Display names of fruits
5. Display details and taste fresh fruits in the basket
6. Mark a fruit in a basket , as stale(=not fresh
7. Mark all sour fruits stale
8. get fruit specific functionality (pulp / juice / jam)
Please enter your choice:
1
Enter color weight and name:
Yellow 4 Mango
Please enter your choice:
2
Enter color weight and name:
Green 7 Orange
Please enter your choice:
3
Enter color weight and name:
Red 9 Apple
```

```
Please enter your choice:
2
Enter color weight and name:
Green 3 Orange
Please enter your choice:
4
Mango
Orange
Apple
Orange
Please enter your choice:
5
Fruit [color=Yellow, weight=4.0, name=Mango, fresh=true]
Fruit [color=Green, weight=7.0, name=Orange, fresh=true]
Fruit [color=Red, weight=9.0, name=Apple, fresh=true]
Fruit [color=Green, weight=3.0, name=Orange, fresh=true]
Please enter your choice:
6
Enter fruit(index) number to mark as a stale:
1
Please enter your choice:
5
Fruit [color=Yellow, weight=4.0, name=Mango, fresh=false]
Fruit [color=Green, weight=7.0, name=Orange, fresh=true]
Fruit [color=Red, weight=9.0, name=Apple, fresh=true]
Fruit [color=Green, weight=3.0, name=Orange, fresh=true]
Please enter your choice:
7
all Sour fruits now stale...!
Please enter your choice:
5
Fruit [color=Yellow, weight=4.0, name=Mango, fresh=false]
Fruit [color=Green, weight=7.0, name=Orange, fresh=false]
Fruit [color=Red, weight=9.0, name=Apple, fresh=true]
Fruit [color=Green, weight=3.0, name=Orange, fresh=false]
Please enter your choice:
8
Enter fruit(index) number to call specific functionality:
2
```

```
name: Orange, color: Green
Extracting juice....!
Please enter your choice:
9
Invalid choice.....!
```