ASSIGNMENT 4 : LAB 3

QUESTION 1

Write a class **Triangle**, which has two member variables **base** of type int, and **height** of type int.

Write a constructor which initialises the base and the height of a Triangle instance.

Write a method **getArea**() that returns the area of the Triangle as a double.

Write a method **show**(), to print the dimensions and area of the Triangle instance.

Write a method **compare**(Triangle t1, Triangle t2), which determines compares the area of two given Triangle objects (hint: recall the Float class compare() method used in Lab #2).

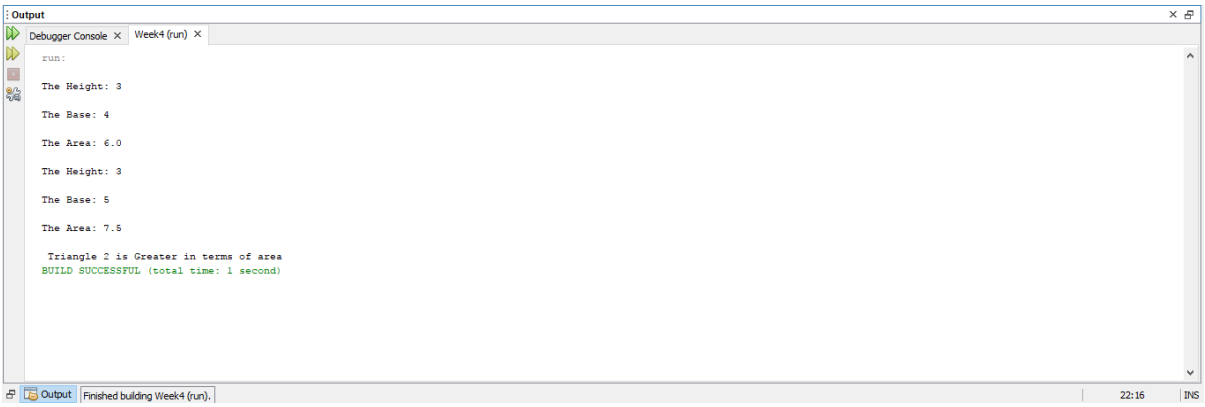
SOURCE CODE

```
/*
 * @author harsh
 */
public class Triangle {
   /*
   When a variable is declared with final keyword,
   it's value can't be modified,essentially, a constant.
   We must initialize a final variable.
   */
   private final int base;
   private final int height;

   public static void main(String[] args){
   Triangle tri= new Triangle(4,3);
   tri.show();
   Triangle tri1= new Triangle(5,3);
```

```
   tri1.show();
   //switch case
      switch (Triangle.compare(tri,tri1)) {
         case 1:
            System.out.println("\n Triangle 1 is Greater in terms of area.");
            break;
         case -1:
            System.out.println("\n Triangle 2 is Greater in terms of area");
            break;
         case 0:
            System.out.println("\n Triangles are equal");
            break;
         default:
            break;
      }
   }
public Triangle(int base, int height){
this.base=base;
this.height=height;
}
public double getArea(){
return (this.base * this.height * 0.5);
}
public void show(){
// height,base,area
System.out.println("\nThe Height: "+ this.height);
System.out.println("\nThe Base: "+ this.base);
```

```
System.out.println("\nThe Area: "+ getArea());

}

//comparing the two triangles

public static int compare(Triangle t1, Triangle t2){

if(t1.getArea()>t2.getArea()) //t1>t2

return 1;

else if(t1.getArea()==t2.getArea())//t1=t2

   return 0;

else

return -1; //t1<t2

}

}
```

OUTPUT :



OBJECT ORIENTED PROGRAMMING LAB

QUESTION 2

Write an IFCSManager class to maintain an array of Equipment objects, sorted according to Equipment id. The IFCSManager will

- add new Equipment instances
- remove an Equipment instance specified by its id
- given an id, report if the Equipment instance resides in the Lab
- display the list of Equipment instances in the Lab.

SOURCE CODE :

File Name : IFCSManager.java

```
/**
 *
 * @author harsh
 */
import java.util.*;
public class IFCSManager
{
   static int i=0,m=0;
   static String n=" ";
   public static void main(String[] args)
   {
      int choice;



      Scanner sc = new Scanner(System.in);
      System.out.print("Enter total number of Equipment objects");
      int ob=sc.nextInt();
      Equipment[] ar =new Equipment[ob];
```

```
    for(m=0;m<ob;m++)
    {
       ar[m]=new Equipment();
    }
  do {
       System.out.println("Perform the following methods:");
       System.out.println("case1: add");
       System.out.println("case2: remove");
       System.out.println("case3: report");
       System.out.println("case4: display");
       System.out.println("case5: quit");
       System.out.println("Enter your Choice:" );
       choice = sc.nextInt();
    switch (choice)
    {
       case 1: add(ar);
       break;
       case 2: remove(ar,n);
       break;
       case 3: report(ar,n);
       break;
       case 4: display(ar);
       break;
          case 5:
       System.out.println("Thank You");
       break;
       case 6:
```

```
            default :System.out.println("Default");
        }
       }
    while (choice != 5);


   }
     public static void add(Equipment[] br){
            br[i].getId();
        if(br[i].id==null)
        {
            System.out.println("ID cannot be null");
            System.out.println("Enter again");
            br[i].getId();
        }



        br[i].getDesc();
        if(br[i].desc==null)
        {
            System.out.println("ID  cannot be null");
            System.out.println("Enter again");
            br[i].getDesc();
        }


        System.out.println("Equipment added to lab");
        i++;
```

```
    }

  public static void report(Equipment[] br,String n)
  {
        int k=0,j=0,count=0;
        Scanner sc = new Scanner(System.in);
        System.out.println("Enter id ");
        n=sc.next();

        if(n==null)
        {
           System.out.println("ID cannot be null");
        }
         else
        {
        for(k=0;k<=i;k++)
        {
           if(n.equals(br[k].id))
           {
            count++;
            break;
           }
         }
        if(count==0)
        {
           System.out.println("Equipment Not found");
```

```
        }
        else
       {
          System.out.println("Equipment present in lab");
       }
       }

    }

    public static void remove(Equipment[] br,String n){
    int k=0,j=0,count=0;
    Scanner sc = new Scanner(System.in);
    System.out.println("Enter the ID to be removed");
    n=sc.next();
    while(true)
    {
     if(n==null)
    {
        System.out.println("ID cannot be null");
        System.out.println("Please Enter again");
        n=sc.next();

    }
    else
    {
    for(k=0;k<=i;k++)
```

```
    {
      if(n.equals(br[k].id))
      {
        for(j=k;j<(i-1);j++)
        {
          br[j].id=br[j+1].id;
          br[j].desc=br[j+1].desc;
        }
        count++;
        break;
      }
    }
    if(count==0)
    {
      System.out.println("ID Not found");
    }
    else
    {
       i--;
       System.out.println("Equipment removed from lab");
    }
    break;
    }
    }
  }
```

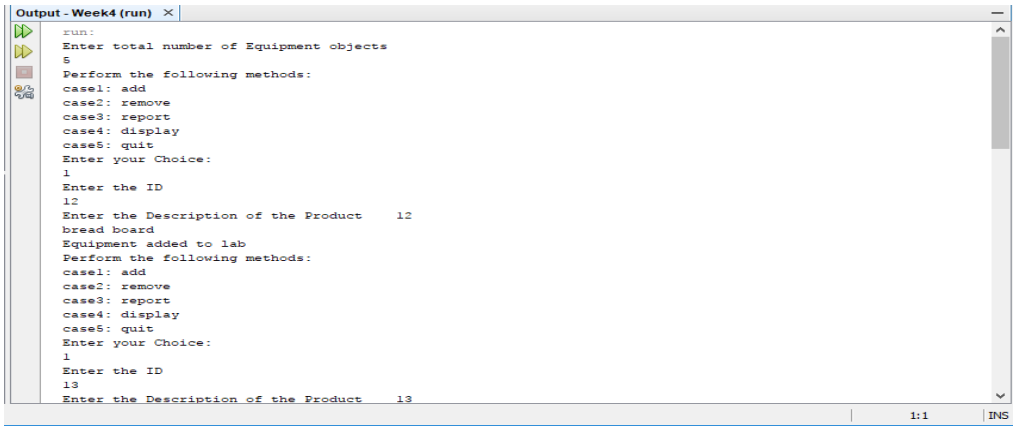
```
  public static void display(Equipment[] br){
     sort(br,i);
     for(int k=0;k< i;k++)
     {
        System.out.println(br[k].id);
        System.out.println(br[k].desc);


     }


  }
  public static void sort(Equipment[] br,int f)
  {
     String temp1=" ",temp2=" ";
     for(int k=0;k<f-1;k++)
     {
        for(int j=0;j<f-k-1;j++)
        {
           if(br[j+1].id.compareTo(br[j].id)>0)
           {
              temp1=br[j+1].id;
              temp2=br[j+1].desc;
              br[j+1].id=br[j].id;
               br[j+1].desc=br[j].desc;
              br[j].id=temp1;
              br[j].desc=temp2;
           }
        }
```

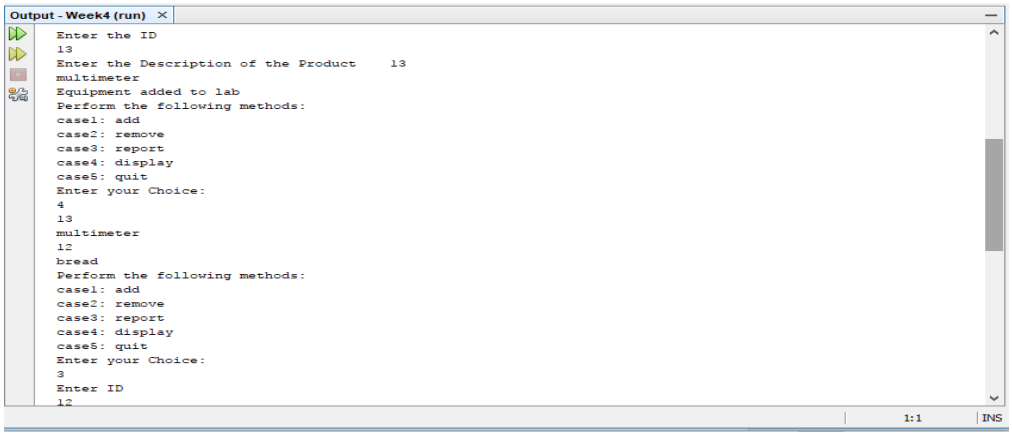
```
            }
          }
    }
```

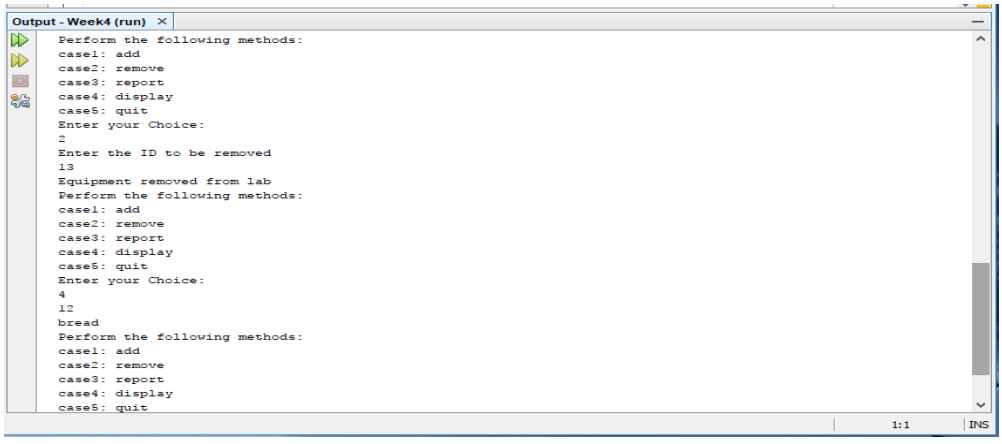
File :- Equipment.java

```
/*
@author harsh
 */
import java.util.Scanner;
//extending the class
public class Equipment extends IFCSManager {
     String id,desc;
     Scanner sc = new Scanner(System.in);
    Equipment()
    {
       id=" ";
       desc=" ";
    }
     Equipment(String a,String b)
    {
       id=a;
       desc=b;
    }
    public void getId()
    {
      System.out.println("Enter the ID");
       id=sc.next();
    }
     public void getDesc()
    {
      System.out.println("Enter the Description of the Product \t" +id+ " ");
       desc=sc.next();
    }}
```

OUTPUT :-


Output - Week4 (run)
run:
Enter total number of Equipment objects
5
Perform the following methods:
case1: add
case2: remove
case3: report
case4: display
case5: quit
Enter your Choice:
1
Enter the ID
12
Enter the Description of the Product 12
bread board
Equipment added to lab
Perform the following methods:
case1: add
case2: remove
case3: report
case4: display
case5: quit
Enter your Choice:
1
Enter the ID
13
Enter the Description of the Product 13
1:1
INS


Output - Week4 (run)
Enter the ID
13
Enter the Description of the Product 13
multimeter
Equipment added to lab
Perform the following methods:
case1: add
case2: remove
case3: report
case4: display
case5: quit
Enter your Choice:
4
13
multimeter
12
bread
Perform the following methods:
case1: add
case2: remove
case3: report
case4: display
case5: quit
Enter your Choice:
3
Enter ID
12
1:1
INS


Output - Week4 (run)
Perform the following methods:
case1: add
case2: remove
case3: report
case4: display
case5: quit
Enter your Choice:
2
Enter the ID to be removed
13
Equipment removed from lab
Perform the following methods:
case1: add
case2: remove
case3: report
case4: display
case5: quit
Enter your Choice:
4
12
bread
Perform the following methods:
case1: add
case2: remove
case3: report
case4: display
case5: quit
1:1
INS