```
package ap.compsci.unit.pkg10;

/**
 *
 * @author lucca
 */
public class APCompsciUnit10 {

    /**
     * @param args the command line arguments
     */
    public static void main(String[] args) {
        Worker[] lantrineIndustries = new Worker[10];
        lantrineIndustries[0] = new Janitor("Dennis");
        lantrineIndustries[1] = new Janitor("Jake");
        lantrineIndustries[2] = new Designer("Jacob");
        lantrineIndustries[3] = new Designer("Edward");
        lantrineIndustries[4] = new Designer("Aman");
        lantrineIndustries[5] = new Coder("Elana");
        lantrineIndustries[6] = new Coder("Aaron");
        lantrineIndustries[7] = new Coder("Chandler");
        lantrineIndustries[8] = new SoundDesigner("Izzy");
        lantrineIndustries[9] = new SoundDesigner("Marc");

        firmOutput(lantrineIndustries);
        profit(lantrineIndustries);
    }

    private static int count = 0;
    private static double profits = 0;
    public static void firmOutput(Worker[] lantrineIndustries){
        if(lantrineIndustries.length > count){
            lantrineIndustries[count].workDay();
            count++;
            firmOutput(lantrineIndustries);
        }
        else
        {
            count = 0;
        }
    }

    //make it so this takes in an integer that counts types of workers (whether it be coder or
whatever) and write a seperate method for that that overwrites the .equals method.
```

```
public static void firmOutput(Worker[] lantrineIndustries,String typeOfWorker){

    Worker[] temp = new Worker[1];

    if(typeOfWorker == "Janitor")
    {
        temp[1] = new Janitor("temp");
    }
    else if(typeOfWorker == "Designer")
    {
        temp[1] = new Designer("temp");
    }
    else if(typeOfWorker == "Coder")
    {
        temp[1] = new Coder("temp");
    }
    else
    {
        temp[1] = new SoundDesigner("temp");
    }

    for(int i = 0; i<lantrineIndustries.length-1; i++)
    {
        if(lantrineIndustries[i].equals(temp[1]))
        {
            lantrineIndustries[i].workDay();
        }
    }


}

public boolean equals(Object other)
{
  return this.getClass().equals(other.getClass());
}

public static void profit(Worker[] lantrineIndustries){

    if(lantrineIndustries.length > count){
        if(lantrineIndustries[count] instanceof Janitor)
        {
            profits = profits + ((Janitor)lantrineIndustries[count]).getProfit();
        }
```

```
            else if(lantrineIndustries[count] instanceof Designer)
            {
               profits = profits + ((Designer)lantrineIndustries[count]).getProfit();
            }
            else if(lantrineIndustries[count] instanceof Coder)
            {
               profits = profits + ((Coder)lantrineIndustries[count]).getProfit();
            }
            else
            {
               profits = profits + ((Designer)lantrineIndustries[count]).getProfit();
            }

            count++;
            profit(lantrineIndustries);
         }
         else if(profits < 0.0 || profits > 100000)
         {
            count = 0;
            System.out.println("The firm made $" + profits + " today. This number should be double
checked before it is added to the checkbook as it is unusual for a firm to make this amount of
money in a day.");
         }
         else
         {
            count = 0;
            System.out.println("The firm made $" + profits + " today.");
         }
      }

      public static void profit(Worker[] lantrineIndustries, String typeOfWorker){

         Worker[] temp = new Worker[1];

         if(typeOfWorker == "Janitor")
         {
            temp[1] = new Janitor("temp");
         }
         else if(typeOfWorker == "Designer")
         {
            temp[1] = new Designer("temp");
         }
         else if(typeOfWorker == "Coder")
         {
```

```
            temp[1] = new Coder("temp");
        }
        //else{}

        for(int i = 0; i<lantrineIndustries.length-1; i++)
        {
            if(lantrineIndustries[i].equals(temp[1]))
            {
                if(lantrineIndustries[count] instanceof Janitor)
                {
                    profits = profits + ((Janitor)lantrineIndustries[count]).getProfit();
                }
                else if(lantrineIndustries[count] instanceof Designer)
                {
                    profits = profits + ((Designer)lantrineIndustries[count]).getProfit();
                }
                else if(lantrineIndustries[count] instanceof Coder)
                {
                    profits = profits + ((Coder)lantrineIndustries[count]).getProfit();
                }
                else
                {
                    profits = profits + ((Designer)lantrineIndustries[count]).getProfit();
                }

                System.out.println("The firm made $" + profits + " today.");
            }
        }

        if(profits < 0.0 || profits > 100000)
        {
            System.out.println("The firm made $" + profits + " today. This number should be double checked before it is added to the checkbook as it is unusual for a firm to make this amount of money in a day.");
        }
    }

}

package ap.compsci.unit.pkg10;

/**
 *
 * @author lucca
```

```
 */
public class Coder extends Worker{
    private double dailyWage = 346.9;
    public Coder(String name){
        super(7,17,45.0,name);
    }
    public double getSalary(){return dailyWage;}

    public double getProfit(){
        return getRevenue() + getSalary();
    }

    public String toString(){
        return super.toString() + "They require a total of $" + getSalary() + " for being in the office
for " + getOfficeHours() + " hours. " + "They provided a total profit to the company of $" +
getProfit() + " today doing tasks such as drawing diagrams, writing code but mostly fixing bugs.";
    }

    public void workDay(){
        super.workDay();
        System.out.println(toString());
        System.out.println();
    }
}

package ap.compsci.unit.pkg10;

/**
 *
 * @author lucca
 */
public class Designer extends Worker{
    private double dailyWage = 285.7;
    public Designer(String name){
        super(8,17,40.0,name);
    }
    public Designer(int inputStartTime, int inputEndTime, double inputValueOfWork, String
name){
        super(inputStartTime,inputEndTime,inputValueOfWork,name);
    }

    public double getSalary(){return dailyWage;}

    public double getProfit(){
```

```
        return getRevenue() + getSalary();
    }

    public String toString(){
        return super.toString() + "They require a total of $" + getSalary() + " for being in the office
for " + getOfficeHours() + " hours. " + "They provided a total profit to the company of $" +
getProfit() + " today doing tasks such as drawing, borrowing assets and designing game
worlds.";
    }

    public void workDay(){
        super.workDay();
        System.out.println(toString());
        System.out.println();
    }
}

package ap.compsci.unit.pkg10;

/**
 *
 * @author lucca
 */
public class Janitor extends Worker{
    private double dailyWage = 73.46;
    public Janitor(String name){
        super(7,17,9.0,name);

    }
    public double getSalary(){return dailyWage;}

    public double getProfit(){
        return getRevenue() + getSalary();
    }

    public String toString(){
        return super.toString() + "They require a total of $" + getSalary() + " for being in the office
for " + getOfficeHours() + " hours. " + "They provided a total profit to the company of $" +
getProfit() + " today doing tasks such as cleaning, inventory keeping, and providing moral
support to the coders and designers.";
    }

    public void workDay(){
        super.workDay();
```

```
        System.out.println(toString());
        System.out.println();
    }
}

package ap.compsci.unit.pkg10;

/**
 *
 * @author lucca
 */
public class SoundDesigner extends Designer{

    public SoundDesigner(String name){
        super(8,17,35.0,name);
    }
    public String toString(){
        return super.toString() + " What they did specifically, was sound designing the game.";
    }

}

package ap.compsci.unit.pkg10;

//@author lucca

public class Worker {
    private int time = 0;
    private int startTime;
    private int endTime;
    private String name;
    private boolean present;
    private int hoursWorked;
    private int hoursSlacked;
    private double valueOfWork;

    public Worker(int inputStartTime, int inputEndTime, double inputValueOfWork, String
inputName){
        startTime = inputStartTime;
        endTime = inputEndTime;
        valueOfWork = inputValueOfWork;
        name = inputName;
    }
```

```
    public void workDay(){
        while(time<24)
        {
            if(time >= startTime && time < endTime){
                present = true;
            }
            else{
                present = false;
            }
            if(time == 12){
                present = false;
            }

            if(present == true){
                int whatWillWorkerDo = (int)(Math.random() * 5);

                if(whatWillWorkerDo == 0){
                    hoursSlacked++;
                }
                else{
                    hoursWorked++;
                }
            }
            time++;
        }
    }

    public int getHoursWorked(){return hoursWorked;}
    public int getHoursSlacked(){return hoursSlacked;}
    public double getRevenue(){
        return valueOfWork * hoursWorked;
    }
    public int getOfficeHours(){
        return endTime - startTime;
    }
    public String getName(){return name;}

    public String toString(){
        return getName() + " worked for " + getHoursWorked() + " hours and slacked off for " +
getHoursSlacked() + " hours today. ";
    }

}
```