Faculty of Engineering
Ain Shams University

REPORT LAB (2)

NAME	ID	EMAIL
Mennat – Allah Ashraf Fouad Fetouh	**17p3051**	mennatallaashraf@gmail.com

Submitted to:

Dr. Islam El-Maddah
Eng. Omar Taalat
Eng. Adham Nour

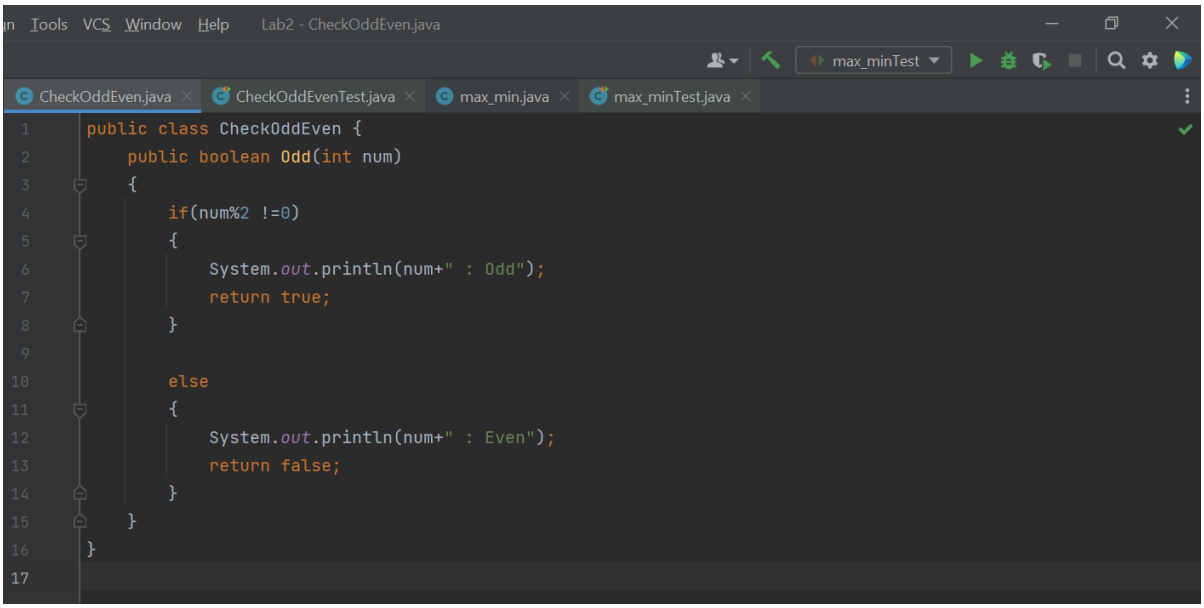
Problem 1:

Check even or odd number.

Solution of Problem 1:

1- The main code:

CheckOddEven.java



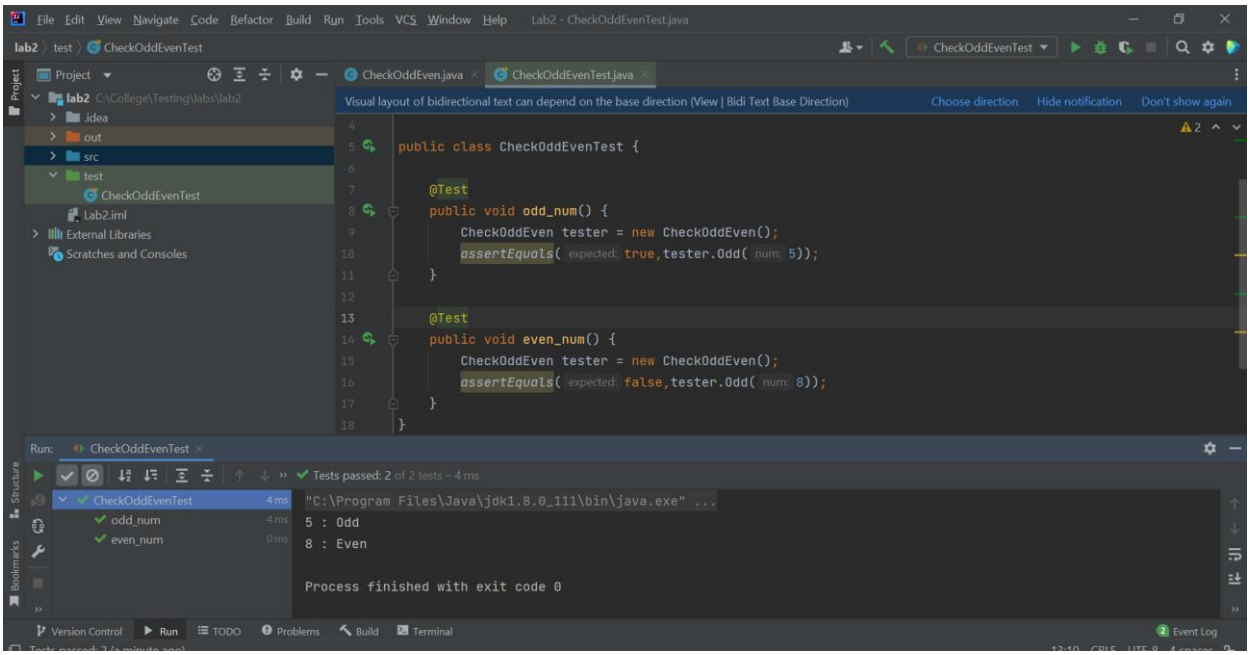

```
public class CheckOddEven {
    public boolean Odd(int num)
    {
        if(num%2 !=0)
        {
            System.out.println(num+" : Odd");
            return true;
        }

        else
        {
            System.out.println(num+" : Even");
            return false;
        }
    }
}
```

2- Test Cases:

CheckOddEvenTest.java

```
import org.junit.Test;
import static org.junit.Assert.*;

public class CheckOddEvenTest {

    @Test
    public void odd_num() {
        CheckOddEven tester = new CheckOddEven();
        assertEquals(true,tester.Odd(5));
    }

    @Test
    public void even_num() {
        CheckOddEven tester = new CheckOddEven();
        assertEquals(false,tester.Odd(8));
    }
}
```

Test Cases "Run" window:



Problem 2:

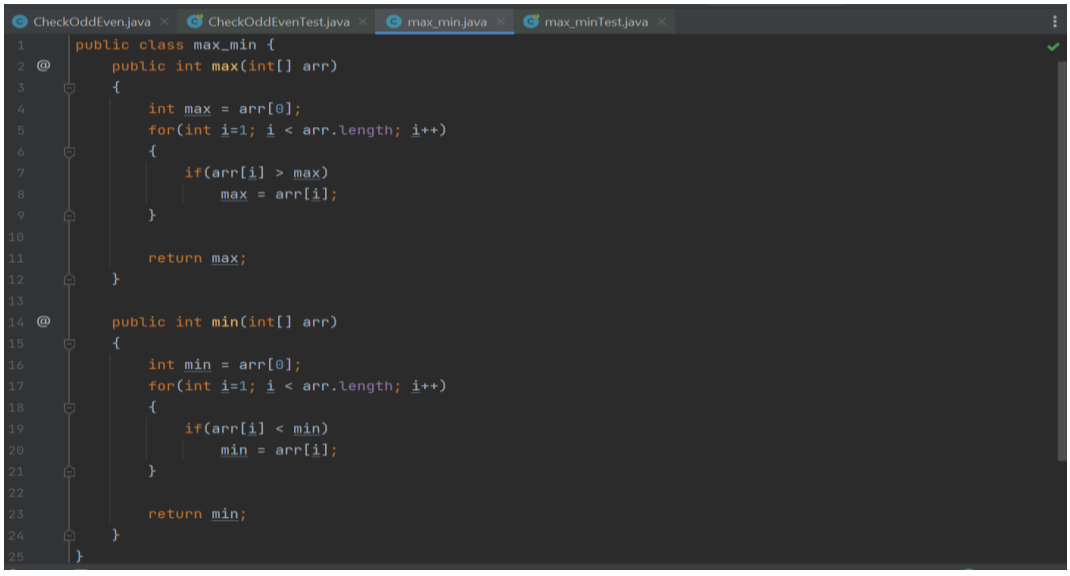
Finding maximum and minimum number in an array.

Solution of Problem 2:

1- The main code:

max_min.java

```
public class max_min {
    public int max(int[] arr)
    {
        int max = arr[0];
        for(int i=1; i < arr.length; i++)
        {
            if(arr[i] > max)
                max = arr[i];
        }

        return max;
    }

    public int min(int[] arr)
    {
        int min = arr[0];
        for(int i=1; i < arr.length; i++)
        {
            if(arr[i] < min)
                min = arr[i];
        }

        return min;
    }
}
```

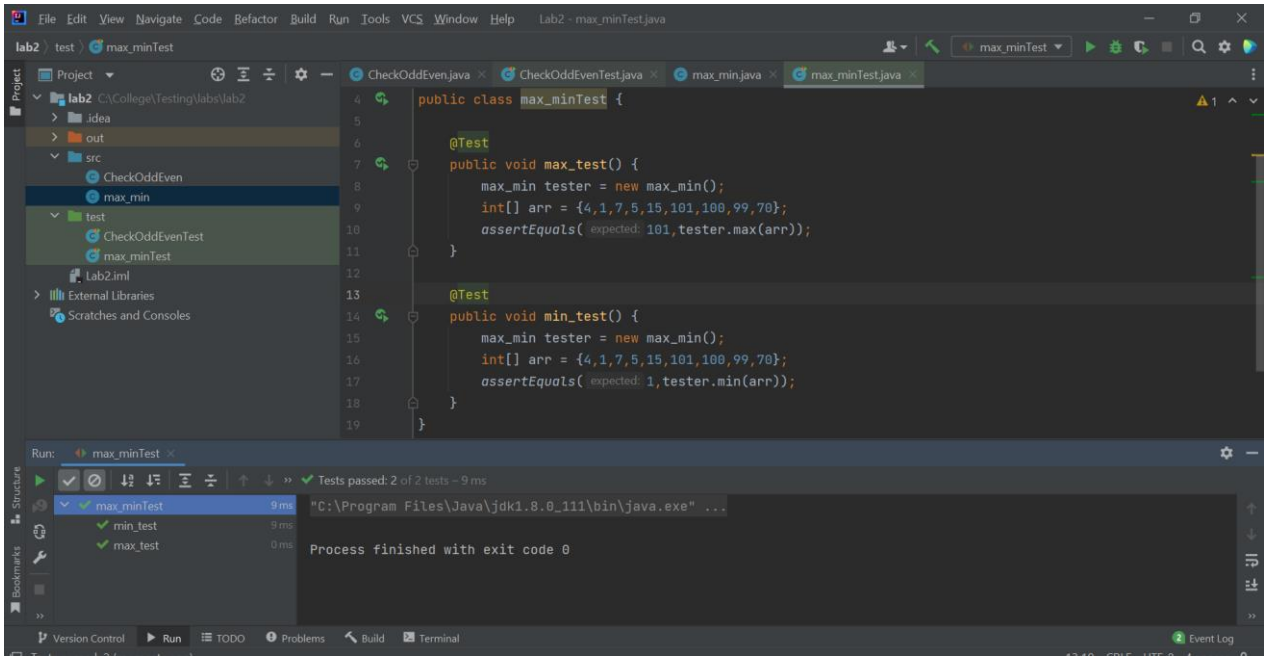


2- Test Cases:

max_minTest.java

```java
import org.junit.Test;
import static org.junit.Assert.*;

public class max_minTest {

    @Test
    public void max_test() {
        max_min tester = new max_min();
        int[] arr = {4,1,7,5,15,101,100,99,70};
        assertEquals(101,tester.max(arr));
    }

    @Test
    public void min_test() {
        max_min tester = new max_min();
        int[] arr = {4,1,7,5,15,101,100,99,70};
        assertEquals(1,tester.min(arr));
    }
}
```

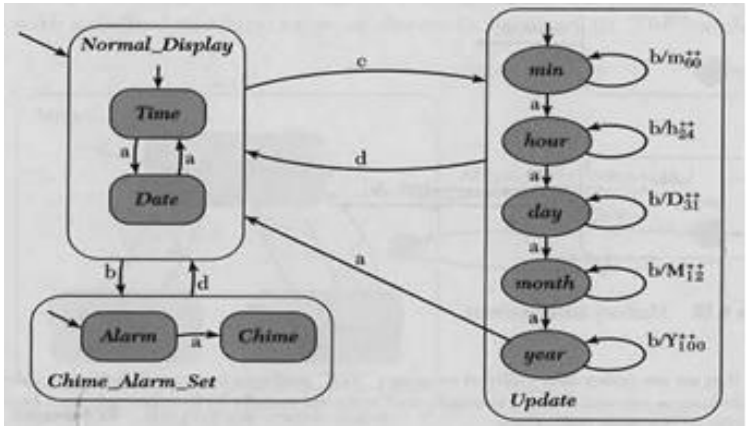
Test Cases "Run" window:



Problem 3:

Sheet3 Q3:

Consider the following state chart for a hand watch that has four buttons a, b, c, d. The watch has three modes normal display to display time or date, update or adjust mode, alarm mode to switch the alarm on or off, M12++ means increase months by one (if current month is 12 it will be 1).



For this problem the following algorithm can be used to model the state chart

Input: is either a, b, c, or d output is a variable showing the current state + the values of current time and current date

Output variables are m=0, h=0, D=1,M=1, Y=2000.

Output displayed information are Time, Date .

set state= NORMAL;

set state1=TIME;

set m=0,h=0, D=1,M=1, Y=2000.

For each input do

switch(state)

Case NORMAL: {

If (input ==c) state=UPDATE

If (input ==b) state=ALARM

If (input ==a)

If (state1==TIME)

state1=DATE

else

state1=TIME

}

Case UPDATE: ..

Case ALARM: ..

}

DisplayDate() {print Y+"-"+M+"-"+D}

DisplayTIME() {print h+":"+m}

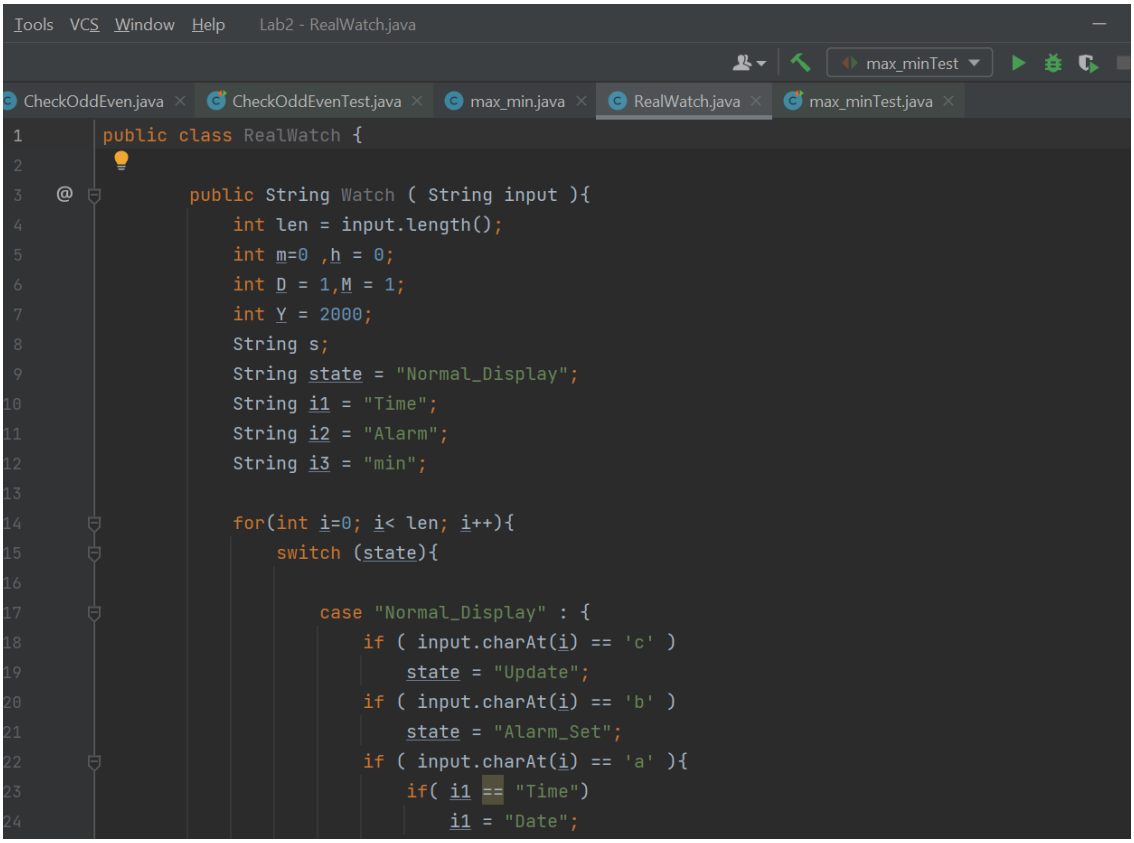
1) Completing the algorithm and code it using C/java, so it can be used as a real watch.

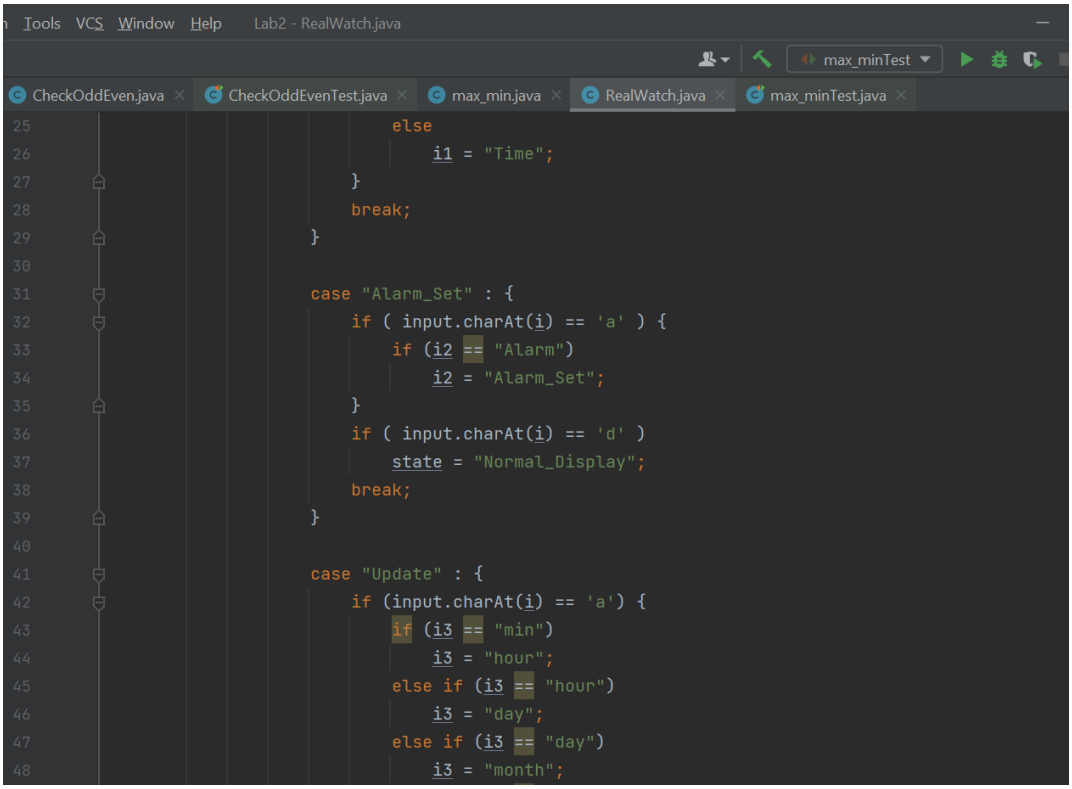
RealWatch.java

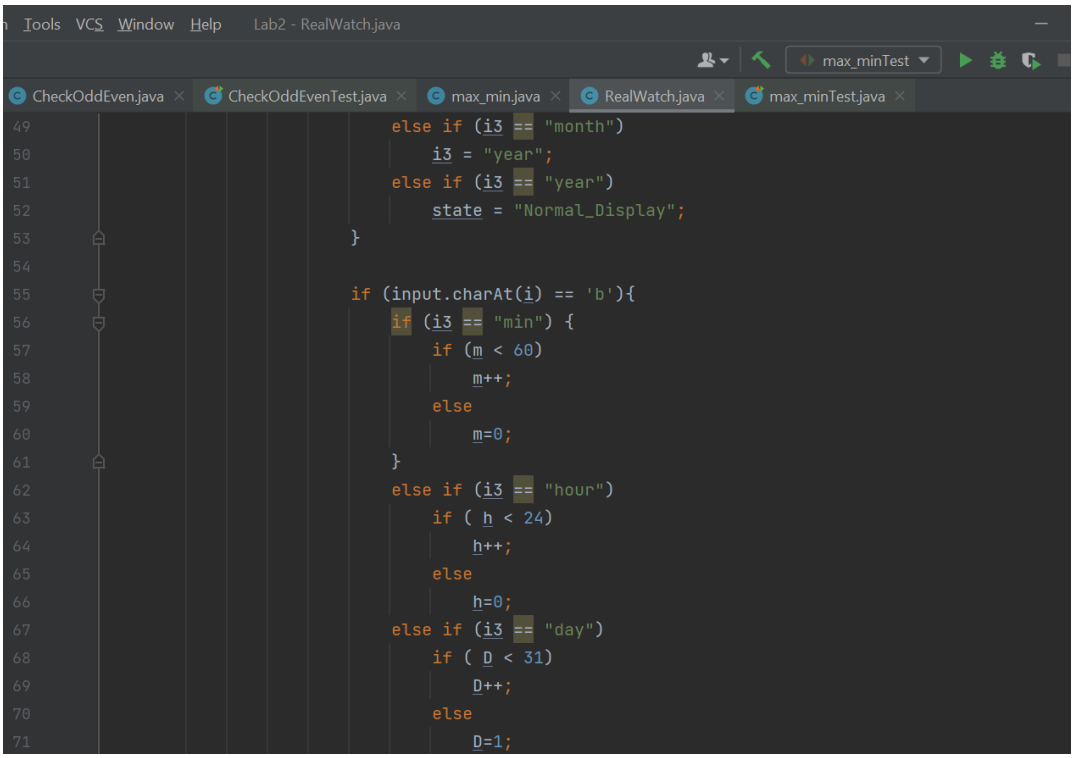
```
public class RealWatch {

        public String Watch ( String input ){
            int len = input.length();
            int m=0 ,h = 0;
            int D = 1,M = 1;
            int Y = 2000;
            String s;
            String state = "Normal_Display";
            String i1 = "Time";
            String i2 = "Alarm";
            String i3 = "min";

            for(int i=0; i< len; i++){
                switch (state){

                    case "Normal_Display" : {
                        if ( input.charAt(i) == 'c' )
                            state = "Update";
                        if ( input.charAt(i) == 'b' )
                            state = "Alarm_Set";
                        if ( input.charAt(i) == 'a' ){
                            if( i1 == "Time")
                                i1 = "Date";
                            else
                                i1 = "Time";
                        }
                        break;
```

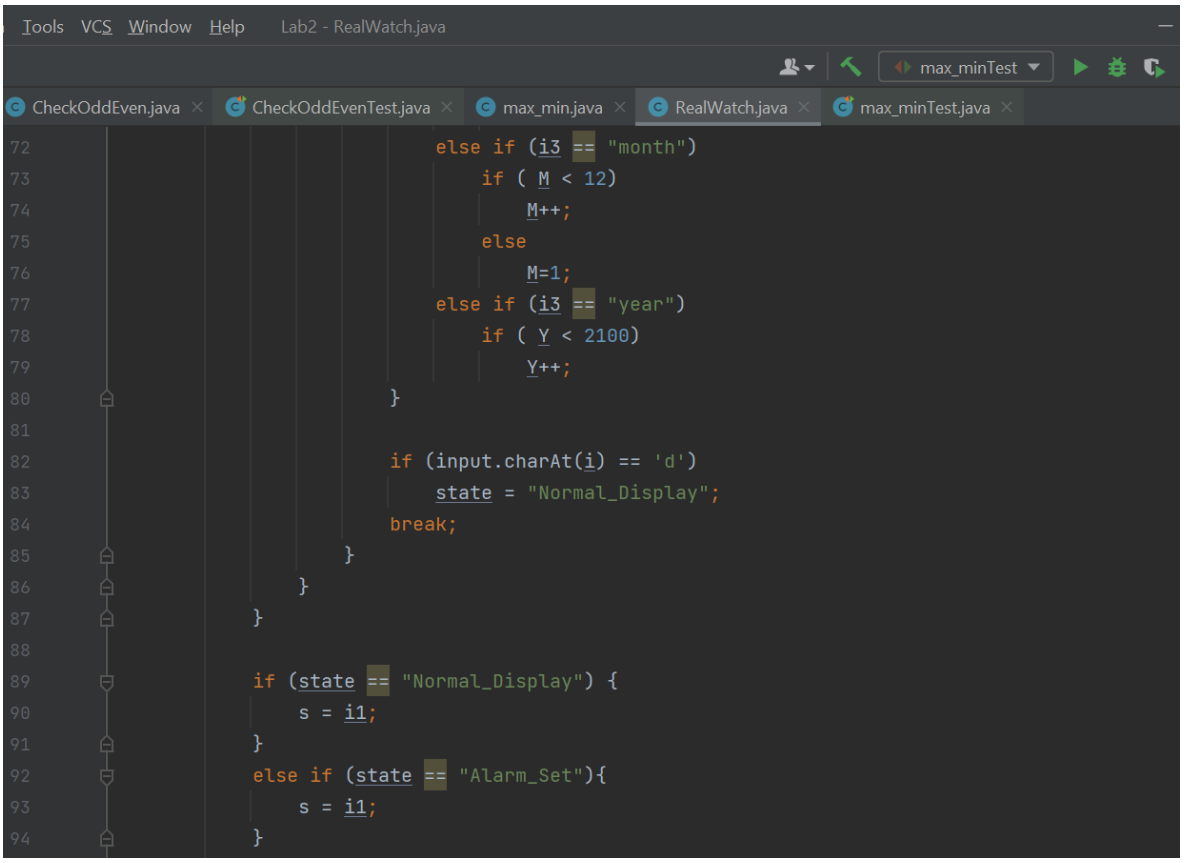
```
}

case "Alarm_Set" : {
    if ( input.charAt(i) == 'a' ) {
        if (i2 == "Alarm")
            i2 = "Alarm_Set";
    }
    if ( input.charAt(i) == 'd' )
        state = "Normal_Display";
    break;
}

case "Update" : {
    if (input.charAt(i) == 'a') {
        if (i3 == "min")
            i3 = "hour";
        else if (i3 == "hour")
            i3 = "day";
        else if (i3 == "day")
            i3 = "month";
        else if (i3 == "month")
            i3 = "year";
        else if (i3 == "year")
            state = "Normal_Display";
    }

    if (input.charAt(i) == 'b'){
        if (i3 == "min") {
            if (m < 60)
                m++;
            else
                m=0;
        }
        else if (i3 == "hour")
            if ( h < 24)
                h++;
            else
                h=0;
        else if (i3 == "day")
            if ( D < 31)
                D++;
            else
                D=1;
        else if (i3 == "month")
            if ( M < 12)
                M++;
            else
                M=1;
        else if (i3 == "year")
            if ( Y < 2100)
                Y++;
    }

    if (input.charAt(i) == 'd')
```

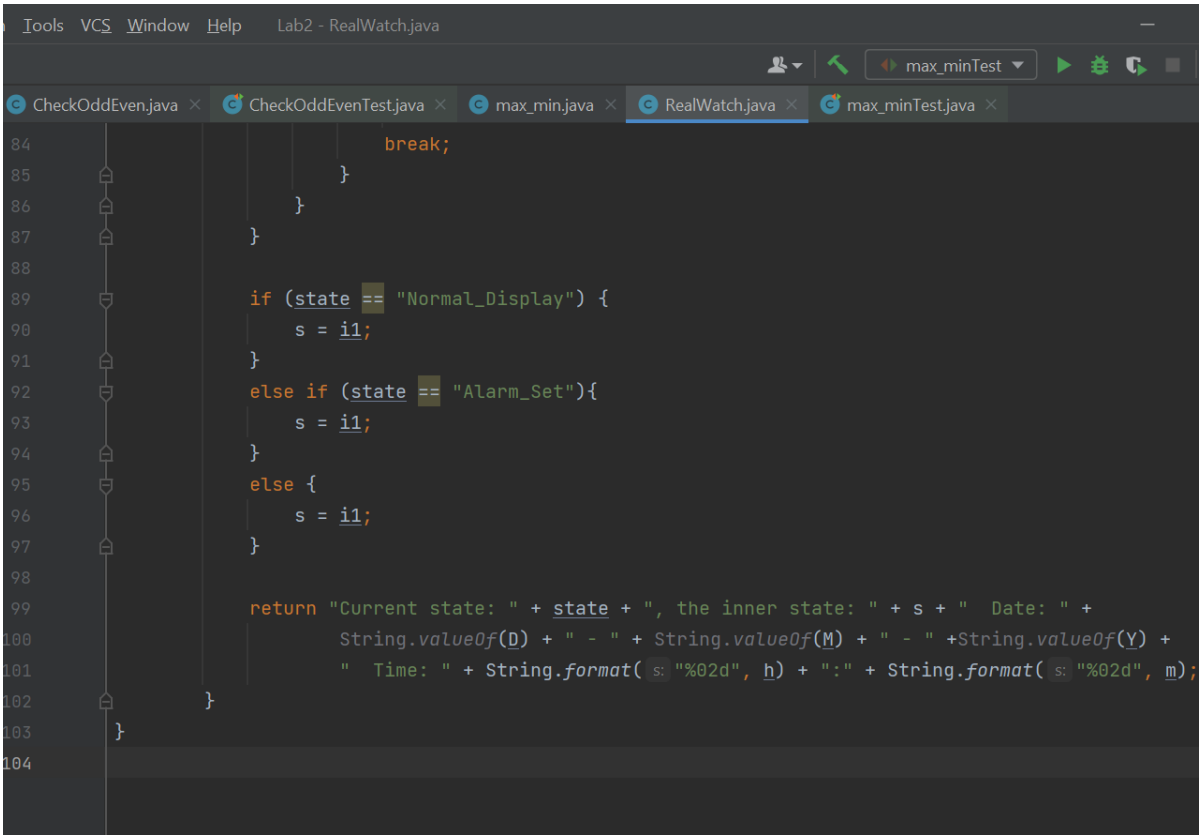
```
                    state = "Normal_Display";
                    break;
                }
            }
        }

        if (state == "Normal_Display") {
            s = i1;
        }
        else if (state == "Alarm_Set"){
            s = i1;
        }
        else {
            s = i1;
        }

        return "Current state: " + state + ", the inner state: " + s + "  Date: "
+ String.valueOf(D) + " - " + String.valueOf(M) + " - " +String.valueOf(Y) + "  Time:
" + String.format("%02d", h) + ":" + String.format("%02d", m);
    }
}
```



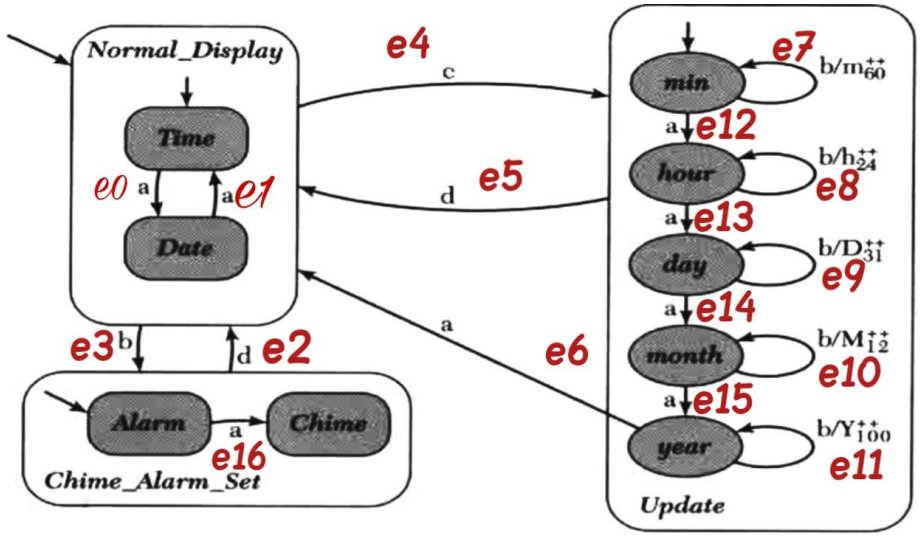

Tools VCS Window Help Lab2 - RealWatch.java
max_minTest
CheckOddEven.java CheckOddEvenTest.java max_min.java RealWatch.java max_minTest.java
else
i1 = "Time";
}
break;
}
case "Alarm_Set" : {
if (input.charAt(i) == 'a') {
if (i2 == "Alarm")
i2 = "Alarm_Set";
}
if (input.charAt(i) == 'd')
state = "Normal_Display";
break;
}
case "Update" : {
if (input.charAt(i) == 'a') {
if (i3 == "min")
i3 = "hour";
else if (i3 == "hour")
i3 = "day";
else if (i3 == "day")
i3 = "month";


Tools VCS Window Help Lab2 - RealWatch.java
max_minTest
CheckOddEven.java CheckOddEvenTest.java max_min.java RealWatch.java max_minTest.java
else if (i3 == "month")
i3 = "year";
else if (i3 == "year")
state = "Normal_Display";
}
if (input.charAt(i) == 'b'){
if (i3 == "min") {
if (m < 60)
m++;
else
m=0;
}
else if (i3 == "hour")
if (h < 24)
h++;
else
h=0;
else if (i3 == "day")
if (D < 31)
D++;
else
D=1;


Tools VCS Window Help Lab2 - RealWatch.java
max_minTest
CheckOddEven.java CheckOddEvenTest.java max_min.java RealWatch.java max_minTest.java
else if (i3 == "month")
if (M < 12)
M++;
else
M=1;
else if (i3 == "year")
if (Y < 2100)
Y++;
}
if (input.charAt(i) == 'd')
state = "Normal_Display";
break;
}
}
}
if (state == "Normal_Display") {
s = i1;
}
else if (state == "Alarm_Set"){
s = i1;
}


Tools VCS Window Help Lab2 - RealWatch.java
max_minTest
CheckOddEven.java CheckOddEvenTest.java max_min.java RealWatch.java max_minTest.java
break;
}
}
}
if (state == "Normal_Display") {
s = i1;
}
else if (state == "Alarm_Set"){
s = i1;
}
else {
s = i1;
}
return "Current state: " + state + ", the inner state: " + s + " Date: " +
String.valueOf(D) + " - " + String.valueOf(M) + " - " +String.valueOf(Y) +
" Time: " + String.format(s: "%02d", h) + ":" + String.format(s: "%02d", m);
}
}

2) find test suite for edge coverage, ADUP.



"e" refers to edges numbers.

Edge coverage:

TC1= <e0,e1>

TC2= <e2,e3>

TC3= <e4,e5>

TC4= <e7,e12,e8,e13,e9,e14,e10,e15,e11,e6>

ADUP :

variables are: m= min, h=hour, D= day, M=month, Y=year.

variable	***D***	***U***
m	*e4*	*e7*
m	*e7*	*e7*
h	*e12*	*e8*
h	*e8*	*e8*
D	*e13*	*e9*
D	*e9*	*e9*
M	*e14*	*e10*
M	*e10*	*e10*
Y	*e15*	*e11*
Y	*e11*	*e11*

GitHub Repo:

GitHub Repo Link is:

https://github.com/Mennah-Ashraf/Testing---LAB2.git