Week 7 Tutorial

Instructions:

The aim of these sessions is to help you prepare for this week's practical. This is a paper-based tutorial. You should record all your answers on a separate sheet (you may use a text editor if you prefer). **You must keep away from BlueJ at all times!!!**

You should work together in a pair. Take turns to lead the discussion for each question and ensure that everyone is happy with the answers and the reasons behind them. If you cannot agree on the answers, or need some help or advice, then please ask the lecturer.

EVERYONE HAS TO CONTRIBUTE THROUGHOUT THE SESSION!!! IF NOT, YOU WILL BE ASKED TO WORK INDIVIDUALLY.

There will be a discussion session at the start of our next class that involves providing alternative solutions and checking these against the ones you produced. Again ask for any clarification of the answers.

These exercises are based on Chapter 3 of "*Objects First with Java*". The code for the NumberDisplay and ClockDisplay classes is given overleaf in the Appendix.

Tasks

1. Look at the **getDisplayValue()** method in the NumberDisplay class. Make sure you understand how it works. Does this work correctly in all cases? What is the logic behind it?

2. What would happen if you replaced the **&&** operator in the test with **||**, so that it read

```
if((replacementValue >= 0) || (replacementValue < limit))
```

3. Look at the **setValue()** method in NumberDisplay. Assume a NumberDisplay object exists (let's call it **nd**) with a limit of 24 and **setValue**() is invoked with a value of 10, i.e. **nd.setValue(10)**; Explain precisely what happens to the parameter 10, paying particular attention to the role of **replacementValue** in this process.

4. Draw **object diagrams** to represent the following fragments of Java code. Follow them through step-by-step. Pay particular attention to the values of the fields and make sure you understand what happens when internal and external methods are called:

```
// a) Create a new clock display
ClockDisplay cd = new ClockDisplay();
// b) set the time
cd.setTime(23, 58);
// c) tick!
cd.timeTick();
// d) tock!
cd.timeTick();
```

5. Assume that the **setValue()** method was modified to that below:

```
public void setValue(int value)
 {
   if ((value >=0) && (value < limit))
       value = value;
 }
```

Would this still work? If not, why not? Again explain very carefully what is going on when this method is called.

6. Rewrite the increment method without the modulo operator, using an if statement.

APPENDIX: Code for NumberDisplay and ClockDisplay classes:

```java
public class NumberDisplay
{
    private int limit;
    private int value;

    public NumberDisplay(int rollOverLimit)  {
        limit = rollOverLimit;
        value = 0;
    }

    // Return the current value.
    public int getValue() {
        return value;
    }

    public String getDisplayValue()   {
        if(value < 10)
            return "0" + value;
        else
            return "" + value;
    }

    public void setValue(int replacementValue)
    {
        if((replacementValue >= 0) && (replacementValue < limit))
            value = replacementValue;
    }

    public void increment()
    {
        value = (value + 1) % limit;
    }
}


public class ClockDisplay  {
    private NumberDisplay hours;
    private NumberDisplay minutes;
    private String displayString;

    public ClockDisplay()   {
        hours = new NumberDisplay(24);
        minutes = new NumberDisplay(60);
        updateDisplay();
    }

    public ClockDisplay(int hour, int minute)   {
        hours = new NumberDisplay(24);
        minutes = new NumberDisplay(60);
        setTime(hour, minute);
    }

    public void timeTick()  {
        minutes.increment();
        if(minutes.getValue() == 0) {
            hours.increment();
        }
        updateDisplay();
    }

    public void setTime(int hour, int minute)  {
        hours.setValue(hour);
        minutes.setValue(minute);
        updateDisplay();
    }

    public String getTime()   {
        return displayString;
    }

    private void updateDisplay() {
        displayString = hours.getDisplayValue() + ":" + minutes.getDisplayValue();
    }
}
```