In this exercise we study how to utilize abstraction with a class in Java.

In this course the official environment is Netbeans 8.1 (or later) for compiling the programs. It is widely used, and it has fairly easy to use graphical user interface. You are also free to use any other environment you prefer.

Using Microsoft Netbeans for Simple Java Programs

Metropolia has Netbeans installed to the main server system which is operated using VMware View Client. To start Visual Studio:

1. Click the WMware icon on the desktop
2. then press Connect (to the connection server desktop.metropolia.fi)
3. finally select e.g. the Android desktop (from the list)
4. Now you have connection to the virtual Windows environment running on the main server. Then click the 'Windows Logo' –button
5. Select All Programs -> Programming -> Netbeans. Now the Netbeans logo should be appeared on your screen.

All the input and output of our exercises is from the user keyboard and to the console screen. So you can use standard `System.out.print` and `Scanner in = new Scanner(System.in); value = in.readDouble()` statements for the input/output operations.

Coding Practices

In order to have our programs more readable we use the following coding rules:

1. Intendation. We use here the K&R intendation style (so-called because it was used in Kernighan and Ritchie's book *The C Programming Language*) because it is commonly used in C++/Java. It keeps the first opening brace on the same line as the control statement, indents the statements within the braces (by 4 spaces), and puts the closing brace on the same indentation level as the control statement (on a line of its own). An example:

```
public static void main(String[] args) {
    ...
    while (x == y) {
        something();
        somethingelse();
        if (some_error)
            do_correct();
        else
            continue_as_usual();
    }
    finalthing();
    ...
}
```

2. Variable names. Use descriptive names for the variable. Your own defined classnames must start with a capital letter, e.g.

```
class Complex {
    private int re;
    private int im;
```

```
    }
```

3. Use empty lines to separate program groups (to separate functions from other functions, variable declarations from the code, etc.), e.g.

```
/*
 * Calculate CCITT (HDLC, X25) CRC
 */
unsigned crc(byte[] blk, int len) {
    const int poly = 0x8408;   // x^16+x^12+x^5+1

    int   result;
    byte  ch;
    int   i, p;

    result = 0xffff;
    for (p = 0; p < len; p++) {
        ch = blk[p];
        for (i = 0; i < 8; i++) {
            if ((result^ch) & 0x001) {
                result = result >>> 1;
                result = result ^ poly;
            } else
                result = result >>> 1;
            ch = ch >>> 1;
        }
    }

    return (result);
}
```

Excercise 1 (Using a class 1p)

We need to write an application, which reads two times (times are represented by two int numbers: hours and minutes). Then the program finds out which time is later. After that it calculates the time difference between these times. Finally the program displays the smaller (earlier) time and the time difference (duration) in the format

starting time was 11:22
duration was 1:04

The main function that does these things looks as follows:

```
public class Timetester {
    public static void main(String[] args) {
         Time time1 = new Time(),
              time2 = new Time(),
              duration;

    time1.read("Enter time 1 ? ");
    time2.read("Enter time 2 ? ");
    if (time1.lessThan(time2)) {
        duration = time2.subtract(time1);
        System.out.print("Starting time was " +
                         time1.toString());
```

```
        } else {
            duration = time1.subtract(time2);
            System.out.print("Starting time was " +
                             time2.toString());
        }

     System.out.println("\nDuration was " + duration.toString());
    }
}
```

Now you need to define and implement class Time so that the program starts working. As can be seen from the main function, class Time has the following member functions:

1. `read` that is used to read time (minutes and hours) from the keyboard.
2. `lessThan` that is used to compare two times.
3. `subtract` that is used to calculate time difference between two times.
4. `toString` that is used to return string representation of the time in the format hh:mm.[1]

[1] If you want to display time values in two character fields with leading '0', use the `Formatter` function.