In this exercise we implement a simple class.

Exercise A (15p) Implement a class

We need to write an application, which reads two times (times are represented by two integers: 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:

```
int main() {
     Time time1, time2, duration;

     time1.read("Enter time 1");
     time2.read("Enter time 2");
     if (time1.lessThan(time2)) {
          duration = time2.subtract(time1);
          cout << "Starting time was ";
          time1.display();
     } else {
          duration = time1.subtract(time2);
          cout << "Starting time was ";
          time2.display();
     }
     cout << "Duration was ";
     duration.display();

     return 0;
}
```

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. `display` that is used to display time in the format hh:mm.[1]

In this exercise we implement overloaded operators to a simple class.

[1] If you want to display time values in two character fields with leading '0', use the following formatting commands:

```
cout << setiosflags(ios::right); cout << setfill('0') << setw(2) << hours;
cout << ":" << setfill('0') << setw(2) << minutes;
```

Note you also need to include <iomanip>

Exercise B(15 p) Implement operators

Improve class you wrote in exercise 4 by adding overloaded operators. The operators to add are:

1. Output operator (<<) that outputs the time in two character fields with leading zeros and separates the fields with a colon.
2. Comparison operator less than (<) that compares two times
3. Addition operator (+) that adds two times
4. Subtract operator (-) that subtracts two times.
5. Pre and post increment operators (++). Both operators increment the time by one minute

Your class should work with the test program below. Note that your class must have a default constructor that initializes time to 0:00.
Addition must make times to roll over to "next day" but doesn't have to keep track of days. For example, adding 14:30 and 13:45 should result in 4:15 or adding 18:30 and 5:37 should yield 0:07.