SW08 **Assignment 1**

The Lap Timer

The task

The goal of this assignment is to build a lap timer application. The lap timer is used to take time in races where the race consists of a number of consecutive laps. The lap timer can record the time of a lap just completed, and it can calculate and display the number of laps completed, the total time, average lap times, and so on.

You will be given a BlueJ project that implements part of the lap timer, and your job is to complete it.

The work – in pairs

This is a pair assignment. You have to work in pairs on solving it (including discussion of the solution and implementation). It is your responsibility to ensure that both partners are equally involved in solving the assignment. After submitting, each partner must be able to fully explain the solution.

The project

Load the project *lap-timer* from the subject web site. Open the project and have a look. You will see three classes: LapTimer, UserInterface and TimerEngine.

LapTimer

This is the main class. It creates a timing engine (TimerEngine) and a user interface and starts off the application. That's really all it does. You do not have to modify this class.

UserInterface

This class provides the user interface of the calculator: the window on screen, the buttons, etc. You do not need to modify this class.

TimerEngine

This is where all the real work is done. The timer engine is responsible for implementing the logic of the timer. It gets informed of button clicks and does the calculations.

This class is not properly implemented! It is your job to write this class.

When you implement the TimerEngine class, you will need to define instance fields and implement methods. The method declarations (the signatures) are already there, just the body is missing (they don't do anything). You may also like to add methods for private use within TimerEngine.

Do not remove or modify the method signatures that are there! These methods get called from the other classes - if you change them, your project will not compile.

Implementation steps

It is a good idea to approach the implementation of this project in a few separate steps. For each step, set yourself a goal that implements part of the solution, and then work towards this step until it is completed. Only if it works, move on to the next step.

Here are a few recommended steps:

Step 1: Get the project to compile. You will notice, if you just try to compile when you get the project outline, that you will get compilation errors. Work towards removing those errors, even if the results are not yet correct. This will help to test partial a implementation of the timer.

Step 2: Make your timer work with a single click on the *Start* button, followed by a single click on the *Stop* button. The *Number of laps* display should show 1, and the *Last lap time* display should show the time between the clicks.

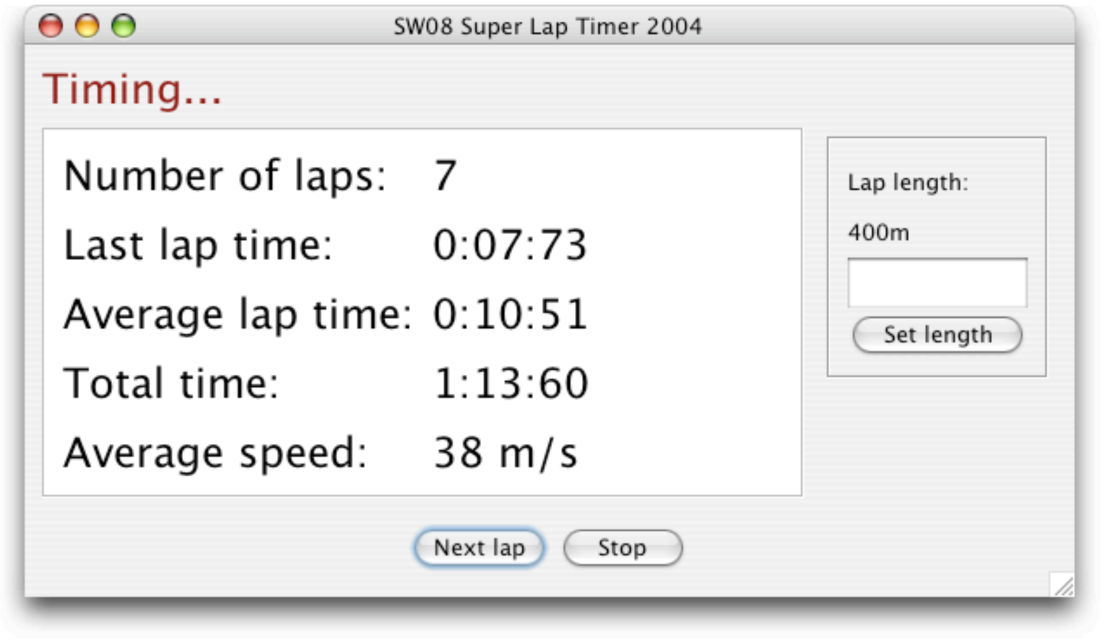
Step 3: Work on recording multiple laps, calculating sums and averages.

Step 4: Make the lap length variable, so that it can be changed.

At every step, think carefully what fields you need, and add then to your class.

The goal

The goal is to produce a fully working lap timer.


SW08 Super Lap Timer 2004

Timing...

Number of laps: 7
Last lap time: 0:07:73
Average lap time: 0:10:51
Total time: 1:13:60
Average speed: 38 m/s

Lap length:
400m

Challenge task

Add the speed of the last lap to the lap timer display.

To do this, you need to modify the UserInterface class to add new data to the display. Then you need to define and implement the required methods in the timer engine.

Submission

Due date: **8 Oct 2004, 12:00**

The assignment must be handed in to the Maersk office on or before the due date and time! Late submissions will not be accepted. If you cannot submit on that day, you have to submit your assignment **before** that time.