EDAA40, Lab 1: Learning Clojure

(version 1.2)

The purpose of this lab is to get acquainted with the programming language Clojure. You'll be working with (parts of) a tutorial called **Clojure from the ground up** by Kyle Kingsbury.

Part 1: Doing the tutorial

The tutorial is available online at the following address:

`https://aphyr.com/tags/Clojure-from-the-ground-up`

We will work with the first three chapters, near the bottom of the page.

Most of the lab will be spent reading and discussing with your lab partner. Once you have a working *repl* (read-eval-print-loop, explained in the first chapter of the tutorial), you don't actually need to type in all the examples, as long as you've read and understood them. You are encouraged to discuss with your lab partner, and you'll find that the repl is often useful for getting ideas across, and trying things out.

- **Welcome**. Read all of this chapter, and follow the instructions for installing Leiningen. You'll have to start a terminal window, and copy/paste some commands from the browser into the terminal.
- **Basic types**. Read up to and including **Vectors**, but not the later sections about sets and maps.
- **Functions**. Read up to and including **Defining functions** (but not **How does type work?**).

Part 2: Editing source files

So far, you have been working with the repl. It is also possible to write clojure code in files and compile them. When you installed Leiningen, you created a scratch project, and navigated to the corresponding directory in the terminal window.

- Exit from the repl with Control-D.
- Edit the file `src/scratch/core.clj` (where `scratch` is the name of your project). If you don't know what editor to use, you could try `gedit`: At the terminal prompt, type the following:

  ```
  gedit src/scratch/core.clj &
  ```

- Add a function definition at the end of this file. The function should compute a given element of the Fibonacci series, $\text{fib}(n)$, defined as follows:

$$\begin{cases} \text{fib}(n) & = 1 & \text{if } n < 2 \\ \text{fib}(n) & = \text{fib}(n-1) + \text{fib}(n-2) & \text{if } n \geq 2 \end{cases}$$

You will need a conditional expression, for example `if`. Type `(doc if)` in the repl to see its documentation.

- Save the file, and type in the terminal: `lein repl` This will give you a repl where you get access to the Fibonacci function you defined. Run `(use 'scratch.core :reload)` in the repl to load (or reload) and import the namespace `scratch.core` where your functions are defined. Test your function. For instance, `(fib 10)` should return 89.

- What is `(fib 50)`? If this seems to take a long time, don't wait for it to complete; it'll take ages. Stop the program with Control-C. Modify your function to be more efficient by translating the following Java code into Clojure:

```
int fibHelper(int n, int i, int previous, int current) {
        if (n == i) {
                return current;
        } else {
                return fibHelper(
                        n,
                        i + 1,
                        current,
                        previous + current);
        }
}

int fib(int n) {
        return fibHelper(n, 0, 0, 1);
}
```

- What is `(fib 100)`? Clojure might complain about an integer overflow. In that case, modify your function to use BigInts.

- Ask the lab assistant to review your code.