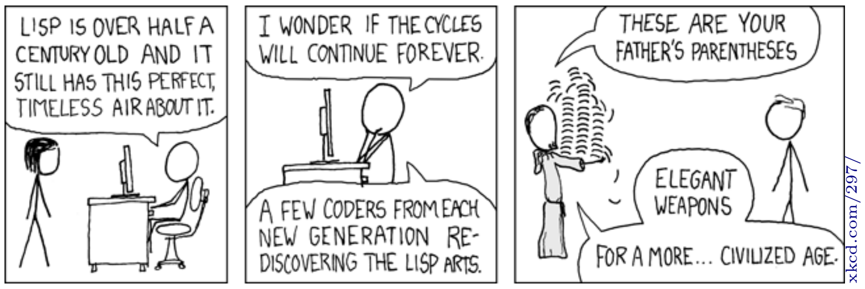
"If you want everything to be familiar, you will never learn anything new."
Rich Hickey, the creator of CLOJURE

Learn the cheekiest programming language!

clojure.org


LISP IS OVER HALF A CENTURY OLD AND IT STILL HAS THIS PERFECT, TIMELESS AIR ABOUT IT.
I WONDER IF THE CYCLES WILL CONTINUE FOREVER.
A FEW CODERS FROM EACH NEW GENERATION RE-DISCOVERING THE LISP ARTS.
THESE ARE YOUR FATHER'S PARENTHESES
ELEGANT WEAPONS
FOR A MORE... CIVILIZED AGE.

xkcd.com/297/

Poetry of Programming – Puzzle-based Introduction to Functional Programming

Level: 200

Credits: 3

Instructor: Attila Egri-Nagy

Office: A3-4

Email: egri-nagy@aiu.ac.jp

Course Description

Experiencing the thrill of problem solving by crafting computer programs is a privilege of very few people: professional software engineers, students of computer science and a handful of hobbyists. Historically, there were reasons for this, since using a computer required an almost complete understanding of its inner workings. This is not true any more, even programming has become user friendly. Indeed, many programming languages nowadays provide small interactive problems to solve and to contemplate on, modelled after zen-buddhist *koans*, as a way of entertaining and insightful learning. For instance, a problem might be stated as a task to fill in the blanks with code,

Given two integers, write a function which returns the greatest common divisor.

```
(= (____ 10 5) 5)
(= (____ 5 7) 1)
(= (____ 1023 858) 33)
```

for which there might are several different solutions. Here is one of them.

```
(fn f [a b]
  (if (zero? b)
    a
    (recur b (mod a b))))
```

This is of course totally meaningless for a reader not familiar with programming languages from the LISP family, in this case CLOJURE. However, even for the first glimpse, it should be clear from the example that the syntax is minimalistic, analogous to a natural language with simple grammar. There are other features of CLOJURE that help the beginner and make it possible to concentrate on the problem itself. It comes with an interactive environment and built-in general purpose immutable and efficient data structures, more than sufficient for problem solving and small size projects. Therefore, tooling requirements are minimal and the need for building heavy code infrastructure is eliminated.

We all learn reading and writing but only a few us turn into professional writers. It is nowadays a frequent thought in education and even at government level that writing computer programs should be a skill shared by many. Similarly to literacy skills, this does not imply that everyone should become a software developer. Actually, in real software development projects, the aforementioned thrilling moments of problem solving are relatively rare, dwarfed by the more prosaic tasks of the profession. Following the literacy metaphor of coding: big software systems are long and dry technical documents, while the short programs developed during the course correspond to little poems. In a sense, this course aims to develop the skill of writing functional program haikus.

Learning Outcomes

The main goal of the course is *to give a direct access to the deepest ideas of functional programming through problem solving.* On the successful completion of this course, it is expected that the students will be able to:

1. write snippets of CLOJURE code for solving logic puzzles and programming problems;
2. read and appreciate code written by others;
3. to use fundamental data structures: lists, vectors and hash-maps;
4. understand the importance of immutable data structures and consequently the principles of the now pervasive version control systems;
5. have a thorough understanding of recursion and self-reference;
6. be able to work with higher-order functions (functions working on functions);
7. improve general problem solving skills;
8. see connections between programming, advanced mathematics and real-world problems.

The course is not planned to provide all the skills necessary for developing complex software applications. However, it will give a fast-track entry point to the post-object-oriented multi-threaded software development world. In case of an IT-oriented career choice, technical knowledge is easier to obtain later, once the deep ideas are comprehended.

Tentative Schedule

Week	Topics
1	**Historical overview:** LISP and CLOJURE. **Introducing the REPL:** interacting with the computer through the command line. Mathematical functions and function composition. Function calls. CLOJURE as a calculator.
2	**List, the most fundamental data collection:** constructing lists, accessing elements, quoting. The `map` function. **Predicates:** `filter` and `remove`.
3	**Associations:** giving meaning to symbols by `def`. **Associative data structures:** vector, hash-map, hash-set.
4	**Defining functions:** `defn`, anonymous functions. **Local bindings:** `let`. **Conditionals.**
5	**Iteration & Recursion:** `loop` `recur`.
6	**Higher-order functions:** `reduce` `iterate` `partial`.
7	**The sequence and collection abstractions.**
8	**MIDTERM TEST**
9	**Pattern matching in text:** regular expressions.
10	**Destructuring:** pattern matching for function arguments.
11	**Lazy sequences:** infinite data structures, integer sequences.
12	**Recursion revisited:** self-reference without names (Y-combinator).
13	**The meta-circular evaluator, macros.**
14	CLOJURE **and art:** OVERTONE – collaborative programmable music; QUIL – interactive drawings and animations.
15	**FINAL EXAM**

Textbooks

A tutorial text for the course will be available in electronic format. (Preliminary draft: `http://www.egri-nagy.hu/pdf/PoP.pdf`)

Optional, beginner-friendly textbooks containing more material:

– Daniel Higginbotham: **Clojure for the Brave and True – Learn the Ultimate Language and Become a Better Programmer**, No Starch Press, 2015, ISBN: 978-

1-59327-591-4, http://www.braveclojure.com/clojure-for-the-brave-and-true/, freely available online

– Carin Meier: **Living Clojure – An Introduction and Training Plan for Developers**, O'Reilly Media, 2015, ISBN: 978-1-4919-0904-1

Software & Resources

Clojure can be used from a browser on all platforms without installing any software package, or installed on any laptop or desktop computer. All software tools are freely available, fully documented and open-source.

– https://www.clojure.org/ – the official home of the Clojure language

– http://www.tryclj.com/ – an interactive REPL for Clojure

– https://www.clojuredocs.org/ – community maintained example-based documentation

– https://www.4clojure.com/ – large collection Clojure koans and a community of problem solvers

Assessment Components

Lab work 40%; continuous evaluation of problem solving work

Midterm & Final exams 30% each; quizzes, code reading exercises and selected problems

Delivery Format

Guided problem solving and code writing in computer lab. Examples will be presented as live coding. **Class capacity:** 25.

Related courses

Strongly recommended:

MAT150 College Algebra, or an equivalent Mathematics course, or mathematically strong high school background. The fundamental concept of functions is required for understanding functional programming.

Recommended:

MAT240 Mathematics Behind The Technological Society. Useful for understanding the computer's inner workings and for learning about more complex algorithms.

CCS125 Programming Principles. Introducing the industry mainstream object-oriented programming paradigm and also the tooling for real-world software development.