Most people would say that interesting puzzles have nothing to do with mathematics as they know it. The problem is that school gives a rather distorted view of the subject. Indeed, the mindless, cruel drill of standard exercise problems has little common with creative problem solving.

Solving a Sudoku puzzle is doing Mathematics. Similarly, MAT245 does contain some maths, but only the artful, enjoyable one. Hence the name, **Poetry** of Programming.

7. TL;DR, just tell me, Why should I?

Computers are everywhere. Can you control them? or do you simply follow their instructions?

> *"Program or be programmed."*
> Douglas Rushkoff

> *"If you want everything to be familiar, you will never learn anything new."*
> Rich Hickey

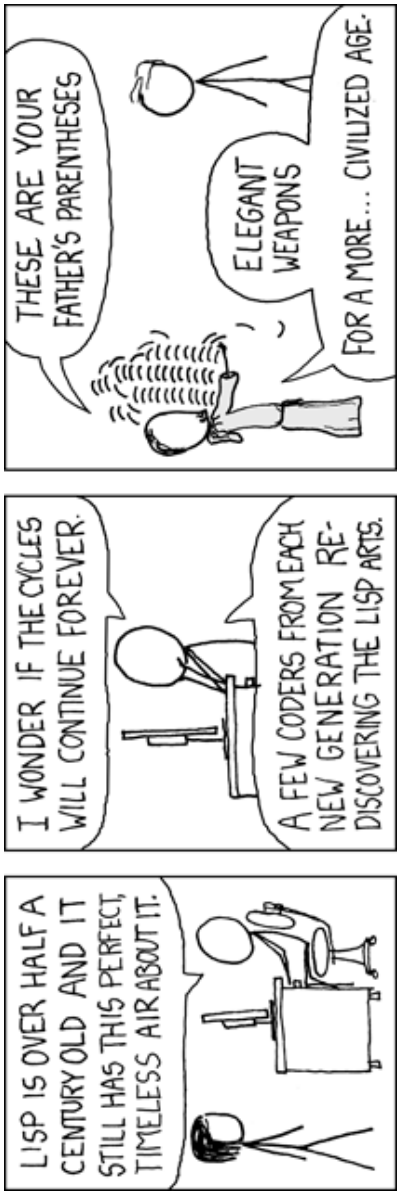
More information:

`https://egri-nagy.github.io/popbook/`

Attila Egri-Nagy `egri-nagy@aiu.ac.jp`

Please feel free to ask questions!

Illustration taken from the course. If you can make fruit salad, then you can code.



made with LaTeX

MAT 245 Poetry of Programming
Puzzle Based Introduction to Functional Programming

2019 Spring @AIU

Questions and Answers

The official logo of the Clojure programming language.
`clojure.org`

1. Why study programming?

Why not? The university is the right place and time to learn new things. After that, specialization takes over. Everyone will focus on his or her career. Regarding the breadth of our knowledge, sadly this often implies closing down, a narrowing worldview.

Programming is like learning a foreign language. One gets to know a different culture, history, ways of thinking.

Programming languages are also used to communicate an understanding of the world, expressing our ideas.

And yes, it is generally considered to be a useful skill.

2. I've never written a program. Can I still do this?

Absolutely! The course is designed for total beginners.

3. I know how to code in a different language. Does it still make sense to take the course?

Yes, learning a new programming language makes you a better programmer. A little warning though, you have to keep an open mind. People knowing some other language usually progress a bit slower, since the chosen language is quite different from languages taught in standard computing courses.

4. Why Clojure? Why this peculiar language?

Ultimately, programming languages don't matter much. What can be programmed in a language can be done in any other language. However, for specific problems some languages might fit better.

Clojure is great for beginners, since it can achieve a lot by using only a few language elements. It sort of has a cheeky character.

Its origin goes back to Lisp (1958), the second programming language ever invented.

John Mccarty (1927-2011), the inventor of Lisp, who also coined the term artificial intelligence, following a computer chess game. (source: Stanford University)

Clojure is designed to be dynamic, which in one sense means that writing programs is like having a conversation with the computer.

> *"I hope you find Clojure's combination of facilities elegant, powerful, practical and fun to use."*
> Rich Hickey, author of Clojure

Rich Hickey (left), the author of Clojure, playing Go. (source: The 1999 New Jersey Yang 7p Go Workshop)

Initially, choosing Clojure was a bet, but now after three semesters we can say that it was a good one. The course also got positive feedback from the industry when it was presented at tech conferences.

Clojure conj 2017

The poetry of programming

`https://youtu.be/XRjPnuPv6xo`

The philosophy of (functional) programming

`https://youtu.be/-yGHsXSgYdg`

5. How does Clojure look like?

This is the implementation of the Caesar-shift substitution cipher, the state of the art cryptosystem in ancient Rome. It simply scrambles text by systematically substituting letters with letters from a shifted alphabet.

```
(ns etudes.caesar-shift
  "Caesar shift and its brute-force attack.")

(def letters "abcdefghijklmnopqrstuvwxyz ")

(defn shift-map
  "A hash-map that maps letters to letters
  cyclically shifted by n."
  [n]
  (zipmap letters
          (take (count letters)
                (drop n (cycle letters)))))

(defn encrypter
  "A function the encrypts plaintext
  by an n-shift cipher."
  [n]
  (fn [text]
    (apply str
           (map (shift-map n) text))))

(defn decrypter
  "A decrypter is just an encrypter
  that shifts back."
  [n]
  (encrypter (mod (- n) (count letters))))

(defn brute-force-attack
  "Calculates all shifts
  thus breaking the cipher."
  [ciphertext]
  (map
   (fn [n] ((decrypter n) ciphertext))
   (range 1 (count letters))))
```

By the end of the course you will be able to read and fully understand this code.

6. How mathematical is the course?

How mathematical is a Sudoku puzzle?