DA2005: Programming techniques

Course introduction

2022-01-17

Who am I?

Evan Cavallo, postdoc in computational mathematics
(https://staff.math.su.se/evan.cavallo)

Have worked at SU for 1 year. Before that, doctorate student in USA

Research programming languages for formalizing mathematical proofs

Ask me practical questions about the course by email
(evan.cavallo@math.su.se) or on the course site

Today

- Goals of the course, programming, and Python
- Practical information: teaching, labs, and literature
- Getting started with programming in Python

Goals of the course, programming, and Python

DA2005: Programming techniques

The class is about **programming**:

- Basic concepts of data science.
- Programming in a modern programming language (Python).
- Data structures and classes.
- Problem solving by dividing up.
- Program structuring.
- ...

Programming is the process of creating a set of instructions that tell a computer how to perform a task. (Khan Academy - What is Programming?)

Algorithm

A programmer implements *algorithms* in the form of programs

Algorithm = sequence of instructions for solving some type of problem

Implement = make an algorithm into a *program* understandable by a computer

Examples of algorithms:

- Mathematical calculations (long division ("liggande stolen"), Erathostenes sieve for generating prime numbers, encryption…)
- Recipes for cooking
- Sheet music
- Sorting (insertion sort, selection sort…)
- …

Python

A very popular programming language (try Googling “most popular programming languages”)

Invented in 1991 by the Dutch programmer Guido van Rossum (https://gvanrossum.github.io/)

Used to write programs for in principle all kinds of applications

Used by many of the biggest IT companies, for example Instagram, Google, YouTube, Spotify, Amazon…

https://www.python.org/
https://en.wikipedia.org/wiki/Python_(programming_language)
https://en.wikipedia.org/wiki/History_of_Python

Why learn to program?

Useful: computers and software are everywhere (phones, computers, cars)

> *"Whether you want to uncover the secrets of the universe, or you just want to pursue a career in the 21st century, basic computer programming is an essential skill to learn." - Stephen Hawking*

Develop problem solving skill and logical thinking

> *"Everybody in this country should learn how to program a computer… because it teaches you how to think" - Steve Jobs*

Well-paying jobs want you (average pay > 40tkr a year for programming in Sweden, even higher in for example USA)

Fun and creative!

How does one learn to program?

Practice, practice, practice…

Like learning any other skill (say, playing an instrument)

Good mathematical knowledge helps, but isn't necessary

This course focuses on developing *practical* skills in programming through labs and a project. We *expect* that you will work actively over the course of the course (half speed = 20h/week)

Obs: The learning curve is fairly steep if you have never programmed before

Practical information

Teachers

Course leader: Evan Cavallo

Teaching assistants:

- Emilia Dunfelt
- Axel Ljungström
- Joel Persson
- Alexander Petri
- Marina Herrera Sarrias

Examiner: Lars Arvestad

Course site: https://kurser.math.su.se/course/view.php?id=1112

Obs: Double-check that you are registered with an email address that you read regularly!

Examination

Three separate Ladok-moment:

- LABO (3hp): 6 labs (G/U)
- INDU (3hp): project (A-F)
- THEO (1.5hp): tests (A-F)

Final grade is calculated according to the table on the course site

Alla units are **individual** (not in a pair or group)

Important: read Rules for labs, project, and exam

Material

Course literature:

- Course notes (in progress)
- Kompendium (in Swedish)

Recorded lectures:

- Institute's YouTube channel

Complementary literature:

- Introduction to Computation and Programming using Python

  by John V Guttag

Please let us know of any errors or typos in the kompendium!

Labs

6 total, put up over time (lab 1 is already on the course site)

Peer grading on PeerGrade: http://www.peergrade.io/ (activation code: J8NG64)

Deadline for labs:

- Hand-in: Sundays 20:00
- Peer grading: Tuesdays 20:00

All deadlines are **strict**. If you miss one, you must redo the lab during the next iteration of the course! (see Rules for labs, project, and exam)

Correction of labs

When the labs have been peer-graded, the teaching assistants will go through them and give detailed feedback by email

We have 4 types of feedback on the labs:

1. Approved (nothing more needed)
2. Minor changes needed (send in a corrected solution by email)
3. Incomplete or buggy code (correct and discuss code in a TA session)
4. Not handed in on time, missed peer-grading, or unserious attempt[1] (hand in during next iteration of the course)

Obs: All submissions are compared automatically and suspected cheating is reported to the university's disciplinary board!

[1] For example if most of the lab is missing.

Tests and project

THEO: individual written exam (14/3)

INDU: larger individual programming project

1. Hand-in deadline: 20/3.
2. Peer-grading deadline: 22/3.

More information to come later in the course

The test will be given remotely.

Obs: separate Ladok-moment from LABO (so even if you miss a lab, you can get Ladok points for THEO och INDU)

Course schedule (schema)

Lectures:

- Monday and Thursday: 12:45–14:30
- Read course notes as preparation
- We'll have discussion and practical group work from the course notes

Lab sessions (Discord: https://discord.gg/7NE54z6H8t):

- Monday and Thursday: 14:45–16:30
- **Individual** help with labs with teaching assistants
- Possibility to review labs
- Format: *see Discord channel*