Programming Lanaugages (0)
Roadmap

Kenjiro Taura

Objectives of programming languages

- easy to learn
- easy to get programs right
- execute fast
- safe (avoid disaster)

The course objectives

- get how different programming languages approach these goals differently
- topics
  - types
  - code reusability (generics, subtyping, inheritance, etc.)
  - memory management/safety
  - performance
  - building compilers
- the main course work: you choose a language from below and do course work in it
  - Go
  - Julia
  - OCaml
  - Rust

The course format

- after a few weeks, we group students
- each group will be four students, each working on a different language
- we discuss approaches to the above objectives taken by different languages, within and across groups
- you are expected to engage in these discussions and other activities (not just to listen to talks and get things done)

Evaluation

- small coding-centric assignments (a few times)
- reflective essay (every week, until the end of the next day)
- participations (esp. in discussions)
- a final report (building a simple C compiler by default; other options are possible)
- no exams

assignments / reports subject for evaluation/grading are indicated as assignments in UTOL

Reflective essay

- every week, you write a short reflective essay that expresses such things as
  - what *you* have learned (conceptualize/internalize experiences)
  - what came through *your* mind while listening to the talk and working on assignments
  - how *you* worked on the exercise (where you struggled, how you got help, how useful was AI, etc.)

Today

- answer a survey on your programming language experiences and the preferred (natural) language
- play with the Jupyter environment
  - choose a language you work on (for today)
  - write a few programs in it
- practice submission (submit pl00_intro in Jupyter and UTOL (Assignment 1))
- work on assignment pl01_basics
- and share your answers!

AI Tutor (Hey Tutor!)

- A few functions you can call from the Jupyter environment to ask any question, ask exercise problems, or feedback to your code
- see `pl00_intro/pl00_tutor.sos` for details
- I encourage you to use AI to help you learn
- You should try to solve the assignment problems by yourself (at least for a while)
  - OK to ask AI to help you when you are stucked (e.g., by sending the code you wrote but did not work)
  - Not OK to ask AI to tell you the answer immediately
  - OK, as a last resort, to ask AI to tell you the answer after you struggle for a long time

Other recommended mindset about the use of AI

- Don't count on AI for the final judgement
  - e.g., judge whether the code is correct or not by actually running/testing it or proving/disproving correctness
- Don't count on AI for the authorative information
  - e.g., sit down and read the language spec/tutorial/manual when you have time
- Reading AI-generated text takes as long as other sources
- Ultimately it is about how much *your brain* absorbs, not how much exercises you solved