CS 314 Lecture 1

January 22, 2019

Logistics

Syllabus

(see the syllabus)

Web sites

Sakai will be the primary source of course info:

https://sakai.rutgers.edu/portal/site/d84c83da-3c3d-4535-bb5c-7f1b7a2753e2

We'll also use Piazza for questions:

https://piazza.com/class/jr7c87u7v73el

Textbooks

- Programming Language Pragmatics
- But likely others...

Prerequisites

- Prerequisite: CS 211
    - We assume you know C and Java

What you'll learn

- How to program in some different programming languages (Python, Haskell, Prolog?)
- How languages are specified
- A bit about types
- What lambda calculus is and why it's universally loved
- What functional programming is about
- Some logic programming and where it might be useful

Expectations

- Roughly 5 assignments (possibly paper & pencil)
- 3 projects

- 1 midterm
- 1 final exam
- All exams are comprehensive

Expectations

- What we expect from you
    - Attend lectures and recitations
    - Read the assigned readings before lecture
    - Read and think about the programming and homework assignments
    - Ask questions
    - Start programming assignments early
    - Don't copy or cheat

Late assignments

- Late assignments will not be accepted
- Programming assignments to be handed in on Sakai
- Deadline will be enforced by Sakai
- Can hand in assignments multiple times so if you are working at the last minute, hand in a version early

Collaboration

Collaboration is encouraged. You learn by discussing with others. But must not cheat...

Department's academic integrity policy:
https://www.cs.rutgers.edu/academic-integrity/

If you are having trouble with the course for any reason, come talk to us.

Corollary: once cheating has occurred, there's nothing we can do to help you avoid the consequences

Topics

- Introduction
- Python
- Dynamic typing
- Lambda calculus
- Functional programming
- Haskell
- Racket?
- Logic programming
- Prolog

Programming assignments

- 5 programming assignments
  - Don't wait until the last minute
  - Learn how to use tools
  - Don't program/debug "by accident" or "by blind search"
- Will be done using the Instructional Lab
  - https://www.cs.rutgers.edu/resources/instructional-lab
  - If you don't already have an account, get one asap
  - https://www.cs.rutgers.edu/resources/getting-started-with-technical-resources-at-the-department-of-computer-science
  - You will be programming in a Linux environment

Grading

Grading:

- 20%: homeworks
- 30%: projects
- 20%: midterm exam
- 30%: final exam
- All exams are cumulative

No make-up exams except for university sanctioned reasons (with professor's approval).

Programming Languages

Programming languages

"A computational process is indeed much like a sorceror's idea of a spirit. It cannot be seen or touched. It is not composed of matter at all. However, it is very real. It can perform intellectual work. It can answer questions. It can affect the world by disbursing money at a bank or by controlling a robot arm in a factory. The programs we use to conjure processes are like a sorceror's spells. They are carefully composed from symbolic expressions in arcane and esoteric programming languages that prescribe the tasks we want our processes to perform."

– Structure and Interpretation of Computer Programs (SICP)

Why not assembly?

- 16 lines of C $\rightarrow$ 200 lines of assembly
- hard to write
- hard to read

Why multiple languages?

Different languages encourage thinking about problems in different ways.

Different languages are expressive in different ways.

Paradigms

- Imperative
- Functional
- Logic

Imperative

A program is a sequence of actions that modify state.

Matches the von Neumann architecture / Turing machines.

Functional

Composition of functions operating on a set of data.

Based on lambda calculus.

Logic

Logical specification of a problem.

Programs declare the form of the solution, not how to find it.

Object oriented

Objects hold state and have methods that can mutate state.

Objects communicate by passing messages or calling methods.

Somewhat orthogonal to other paradigms.

Other paradigms

- Event-driven: asynchronous events trigger actions (GUIs)
- Parallel: breaks task into pieces (at some granularity), executes in parallel with synchronization mechanisms

Syntax and semantics

- apple
- banana
- aodorcuoacedgaduea

Syntax and semantics

- I eat an apple.
- Colorless green ideas sleep furiously.

Syntax and semantics

Variable names:

- abc123
- 123abc
- 24
- while