Course Description and Goals:
This purpose of this course is to consider in detail the main constructs of modern programming languages and several programming language paradigms, including imperative, functional and object-oriented. We'll also discuss some pragmatics of programming as time allows.

Why study principles of programming languages?
- To enhance your ability to learn new languages,
- To allow you to choose an appropriate tool for a given task,
- To gain an appreciation for the challenges involved in implementing a language,
- To expand your ability to express your ideas using a given language,
- To see some very different styles of programming (e.g., Haskell),
- Because it's fun!

Textbooks
Miran Lipovaca, *Learn You a Haskell for Great Good*, Required - online
Flanagan & Matsumoto, *The Ruby Programming Language*, Recommended
Robert Sebesta, *concepts of Programming Languages*, Optional

Student Evaluation
Assignments: 45%
Class participation: 5%
Exams and quizzes: 45%
Language comparison: 5%

Late and Submission Policy
Late assignments will be penalized 10% per day. For small assignments (less than 10 points), the late penalty is 1 point per day. No assignments will be accepted more than 2 days late. In-class exercises should be turned in before the start of the next class period.

Programs submitted for grading must run. DO NOT SUBMIT A BUGGY PROGRAM THAT DOESN'T RUN! It is much better to submit a working program that only meets part of the requirements. You are strongly encouraged to ask for help via piazza! This is good experience, and will avoid receiving a 0 on homework assignments when you've put
in lots of hours. If piazza posts don't provide sufficient help, I will also look at your program. Don't wait til the last minute, though.

Course Schedule (Tentative)

Available on the web

Collaboration Policy for Programming Projects in EECS Courses

The following policy exists for all CS courses in the EECS department. This policy is a minimum standard; your instructor may decide to augment this policy.

1. If the project is an individual effort project, you are not allowed to give code you have developed to another student or use code provided by another student. If the project is a group project, you are only allowed to share code with your group members.

2. You are encouraged to discuss programming projects with other students in the class, as long as the following rules are followed:
   a) You view another student's code only for the purpose of offering/receiving debugging assistance. Students can only give advice on what problems to look for; they cannot debug your code for you. All changes to your code must be made by you.
   b) Your discussion is subject to the empty hands policy, which means you leave the discussion without any record [electronic, mechanical or otherwise] of the discussion.

3. Any material from any outside source such as books, projects, and in particular, from the Web, should be properly referenced and should only be used if specifically allowed for the assignment.

4. To prevent unintended sharing, any code stored in a hosted repository (e.g., on github) must be private. For group projects, your team members may, of course, be collaborators.

5. If you are aware of students violating this policy, you are encouraged to inform the professor of the course. Violating this policy will be treated as an academic misconduct for all students involved. See the Student Handbook for details on academic dishonesty.