Today

• Course Overview
• Survey and Exercise

Assignments

• HW-1, R-1
Course Overview

Course webpage

- [http://www.cs.gonzaga.edu/~bowers/courses/cpsc326](http://www.cs.gonzaga.edu/~bowers/courses/cpsc326)
- Be sure to check frequently (notes, readings, assignments, etc.)

Please be sure to **carefully** read the syllabus

- **Office hours:** Tu/Th from 11:00am–12:25pm, Wed 1:00pm–2:00pm or by appointment
- **Grading:** 30% hw, 10% project, 15% quizzes, 45% 3 mid-semester exams
- **Homework late policy:** In class on due date, NO LATE WORK ACCEPTED
- **Missing class:** more than 4 absences may result in a V (i.e., F) grade

Required Textbook:

Exam and Homework Policies

Exam, Quiz, and Homework basics

- Exams and Quizzes are closed book, closed note, your own work
- You are responsible for understanding how to do the assignments
- You will be tested on material from assignments (reading & programming)
- Okay to work on and discuss assignments together
- **Must turn in homework individually**

Weekly Reading/Written Assignments

- Grading scheme: “✓ +” (excellent), “✓” (pass), “✓ −” (issues)
- Make sure I can read your answers (or else type them)
- Organize your answers by question
- You **must** have access to a copy of the textbook!
- Why reading assignments?
  - Motivate you to read the book, help you think about material
  - Mainly focused on general PL topics

Weekly Programming Assignments

- Graded out of 15 points (5 on documentation, 10 on correctness)
- **Must include cover sheet with source code**

Extra credit opportunities

- There will be some additional extra credit assignments
- More details later
Python ...

One of the first thing’s we’ll do is implement an interpreter

- We’ll be using Python (2.7) for this
- If you haven’t used Python or are rusty ...
  - The Hitchiker’s Guide to Python – docs.python-guide.org
  - Crash into Python – stephensugden.com/crash_into_python

We’ll be starting with Python programming next week

- I’ll go over some aspects needed
- But, you need to know the basic syntax
- E.g., functions, conditions, loops, lists, dictionaries
Quizzes

We’ll have frequent quizzes

- Cannot make up a quiz
- Given at the beginning of class (so don’t be late!)

Why?

- Give you feel for kind of questions I ask on exams
- Give you feedback on your understanding of material
- Give me feedback on your understanding

Be prepared for quizzes

- You’ll likely need to study for quizzes to do well
- Often over material from previous lecture
- Won’t necessarily be given after an assignment is due
- I will tell you when a quiz is coming
Project

Learn a new language and use it to develop an application

- you’ll pick a language you don’t know
- use tutorials, books, manuals, etc.
- build a small application that uses what you’ve learned
- present a “tutorial” and your application

Goal is to:

- gain experience learning a language on your own and
- gain experience doing an “un-guided” side project

More on the project later in the semester ...
Course Topics

This course covers various aspects of programming languages

Focus is on:

- Programming language paradigms (or “families”)
- Programming language design (constructs)
- Programming language implementation
- Hands-on use of different languages

Languages and topics

About 2/3 of course (approximately 10–11 weeks)

- Functional programming in Haskell
- Logic programming in Prolog

About 1/3 of course (approximately 4 weeks)

- Language design and implementation
- ... syntax, semantics, compilation/interpretation
Exercise ... What is a Programming Language?

One type of definition of a PL (from wikipedia)

“A programming language is a formal language designed to communicate instructions to a machine, particularly a computer.”

“A programming language is a notation for writing programs, which are specifications of a computation or algorithm.”

Another definition: **Turing Complete**

- A programming language can express all computations (algorithms)
- Turing complete if can simulate any Turing Machine (i.e., a universal TM)
- Examples of languages that are not Turing Complete:
  - Markup languages: HTML, XML, JSON, ...
  - Many “domain-specific” languages: SQL, regular expressions
- Not necessarily tied to specific constructs
  - imperative languages with conditional branching (if-goto, while loops) and arbitrary mem access (# of variables)
  - whereas Haskell and Prolog use recursion (no goto, no loops)
Exercise ... Why study Programming Language concepts?

1. Easier to learn new languages
   - you'll likely need to learn many in your career

2. Ability to choose appropriate languages
   - only knowing one or two languages limits choices

3. Better use of languages
   - most modern languages are complex
   - hard to know all features & details

4. Better able to express ideas
   - languages place limits on control & data structures, abstractions
   - awareness of features can make some problems easier
   - simulate features in languages (e.g., closures or design patterns)

5. Understand implementation issues
   - visualize better how PLs work (can help, e.g., with optimization)
   - understand trade-offs between languages

6. Advance software engineering
   - most popular not always the best (e.g., ALGOL 60 vs. Fortran)
   - those choosing languages not always most knowledgeable