Today

- Course Overview
- Survey and Exercise

Assignments

- HW-1, R-1, PROJ-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, Wed, Th from 1:00–2:30pm or by appointment
- **Grading**: 30% hw, 10% 2 projects, 10% quizzes, 25% 2 exams, 25% final
- **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

- 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

Programming Assignment Grading

1. Hand in on due date
2. First pass (P1) – 1 point per question attempted
3. I will hand back and post the key
4. Grade your own work against the key:
   - **Mark**: Correct, Minor mistake, Major mistake, Wrong approach/Confused
   - **Identify**: Circle/mark where mistake happened
   - **Explain**: Explain what happened
   - **Fix**: Fix the mistake
5. Hand in corrections
6. Second pass (P2) – 4 points per question answered
   - if correct, all 4 points (assuming you marked as correct)
   - if incorrect, 1 point for marking, identifying, explaining, and fixing

Reading Assignment Grading

- Check plus (excellent), Check (pass), Check Minus (major/minor issues)
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

- Often over material from previous lecture
- Not necessarily after an assignment is due
- Will tell you when a quiz is coming
Homework

Reading Assignments

- Motivate you to read the book
- Reflect on material
- Mainly focused on general PL topics
- Make sure I can read your answers (or else type them)
- Organize your answers by question

Programming Assignments

- Only hand in your program source code (unless otherwise instructed)

Extra credit opportunities

- There will be some additional extra credit assignments
- Of course, these are optional
- More details later
Projects

Two learn-a-new-language side projects

• one in first half of semester
• one in second half of semester

In each project you will learn a new language and use it to develop an application

• pick a language you don’t know
• use tutorials, books, manuals, etc.
• build a small application that uses what you’ve learned
• do a little every week
• create a presentation (e.g., powerpoint video)

Goal is to gain experience learning on your own and doing side-projects
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 two 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