Principles of Programming Languages

Instructor                     Prof. Stephen Freund
Office                         TPL 302, 597-4260
Email                          freund@cs.williams.edu

Class Web Page                 http://www.cs.williams.edu/~freund/cs334/index.html

Texts

We will be using the following text book:

- *(Required)* *Concepts in Programming Languages*, John C. Mitchell.

Additional readings will be posted on the web site.

Course Objectives

A programming language is a programmer’s principle interface to the computer. As such, the choice of an appropriate language can make a large difference in a programmer’s productivity. A major goal of this course is to present a comprehensive introduction to the principle features and overall design of both traditional and modern programming languages. You will examine language features both in isolation and in the context of more complete language descriptions. The material will enable you to:

1. Quickly learn programming languages, and how to apply them to effectively solve programming problems.

2. Rigorously specify, analyze, and reason about the behavior of a software system using a formally defined model of the system’s behavior.

3. Realize a precisely specified model by correctly implementing it as a program, set of program components, or a programming language.

We will examine features of a large variety of languages, though we will not study many of languages themselves extensively. Like other CS courses, we will discuss alternate ways of solving problems, looking at the pros and cons. Because programming languages are so tied up (and motivated by) programming problems, we will not only investigate language features, but also the software engineering problems that spawned them.

At the end of this course you will have a more thorough understanding of why certain programming language features provide better support for the production of reliable programs, while others are fraught with ambiguity or other problems. Since programming languages mediate between the programmer and the raw machine, we will also gain a deeper understanding of how programming languages are compiled, what actually happens when a program is executed on a computer, and how the programming language design affects these issues. As an example, by the end of the course, you should be able to understand why Java has replaced C++ language of choice for many projects and to recognize where language design is likely to head in the future.

An important feature of this course is the discussion of programming language paradigms (in particular, languages supporting new ways of thinking about implementing algorithms). We will investigate both the new features themselves and the software engineering problems which spawned these developments.

This course will involve extensive reading on your part, both in the text and in outside sources. The segments of the course that introduce new programming language paradigms will also feature some
programming in languages representative of the functional and object-oriented paradigms (Lisp, ML, Scala, and possibly others).

Lectures

Lectures are mandatory. I expect you to attend and participate. The website has a tentative schedule of lecture topics.

Homework

Problems involving analysis of programming language features will be assigned weekly and typically on Tuesday. Your homework should:

• be turned at the beginning of class on the due date,
• be clearly written or typed,
• list any students with whom you discussed the problems (see Honor Code below), and
• be stapled, with your answers clearly marked and in the same order as the questions.

You should submit source code print outs for questions involving programming separately from the written questions.

You will be asked to fix and resubmit homework not following these conventions. I will not accept homework by email.

Each student may use a maximum of three late days during the course of the semester. A single late day enables you to hand in the homework up to 24 hours after the original due date. Once those late days are used up, late homework will be penalized 20% per day. Late homeworks should be turned into Lauren Vining in TCL 303.

Programming

The homeworks will often include small programming assignments to reinforce conceptual ideas from lecture and expose you to different programming paradigms. For some of these you will be required to work with a partner. I will provide more details on the first homework including a programming component.

We will use the Computer Science Department’s Unix computers for the programming problems. If you are not familiar with the Unix computing environment, talk to me or the TA as soon as possible so we can bring you up to speed on what you need to know. You may also see Lida Doret (TCL 205) to get your Unix password if you have forgotten it.

Exams

There will be midterm and final exams covering topics from lecture, readings, and homework. The midterm will be self-scheduled and (tentatively) taken between 10/24 and 10/26. The final will be a 24-hour Take Home Exam.

Grading

Grades will be determined roughly as follows:

<table>
<thead>
<tr>
<th>Component</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Midterm</td>
<td>25%</td>
</tr>
<tr>
<td>Final Exam</td>
<td>35%</td>
</tr>
<tr>
<td>Homework and programs</td>
<td>35%</td>
</tr>
<tr>
<td>Other (class participation, attendance, quizzes, etc.)</td>
<td>5%</td>
</tr>
</tbody>
</table>
Honor Code

Discussing written problems with other students in the class is permitted. However, copying of solutions is not. The work you hand in should be your own. A good rule to follow is to work through the problems with others, taking only rough notes. You should then write your solutions independently, referring to your notes as little as possible. The idea is to understand each solution well enough that you can reconstruct it by yourself. In addition, you should write on each assignment the list of people with whom you collaborated.

The same applies to programming — feel free to discuss general ideas with others, but the design and code you submit should have been written entirely by you (or your group if it is a group assignment).

I will occasionally give more detailed collaboration guidelines on some assignments. Uncredited collaborations will be considered a violation of the honor code and will be handled appropriately.

Reference Material: You may refer to any course materials and the web sites listed under the Resources heading when completing your homework. You may not use other sources to look for solutions to your homework questions, and you may never copy prose or code directly from another source.

Please also read the complete The CS Honor Code and Computer Usage Policy at

https://csci.williams.edu/the-cs-honor-code-and-computer-usage-policy/

If in doubt as to what is appropriate, do not hesitate to ask me.

After the semester: Please do not post your solutions to the CS 334 problem sets or programming problems in any public forum, including public GitHub repositories.