15-122: Principles of Imperative Computation

Lecture 1: Frank Pfenning & Ananda Gunawardena
Lecture 2: Thomas Cortina & Rob Simmons

http://www.cs.cmu.edu/~fp/courses/15122-f12
http://c0.typesafety.net
Overview

• Goals of this course
• Interactions
  • Lectures, Recitations, Office Hours
• Assessment
  • Quizzes, Homeworks, Exams
• A mysterious function!
Goals: computational thinking

• Specification vs. implementation; correctness
• Logical vs. operational reasoning
• Abstraction and interfaces
• Loop and structure invariants
• Reasoning about resource bounds
Goals: programming skills

• Transformation of algorithmic ideas into correct imperative code

• Specify, write, test, debug, (re)factor code in the small

• Some familiarity with Unix tools and C
Goals: programming language

• $C_0$: a small safe subset* of C
  • int, bool, char, string, arrays, pointers, structs
• Essential algorithmic and programming ideas
• Relatively close to machine (imperative)
• Sound reasoning with contracts
• Transition to C near end of course
Goals: algorithmic ideas

• Asymptotic complexity
  • time/space/amortized
  • worst case/average case
  • important classes: \( O(1) \), \( O(\log n) \), \( O(n \log n) \), \( O(n^k) \), \( O(2^n) \)

• Divide-and-conquer

• Self-adjusting data structures

• Emphasis on imperative prog’s, ephemeral data structures
Goals: concrete algorithms

- Basic arithmetic
- Binary search, sorting
- Stacks and queues, priority queues
- Binary trees, dictionaries, maps, sets, tries
- Hashing, hash tables
- Graph traversal, minimum spanning trees
- Binary decision diagrams
The Big Picture

• Pre- or Co-requisites:
  • either 15-151 Math Foundations for CS
  • or 21-127 Concepts of Mathematics

• Counterpart:
  • 15-150 Principles of Functional Programming

• Is Pre-requisite for:
  • 15-213 Introduction to Computer Systems
  • 15-210 Fundamental Algorithms and Data Structures
  • 15-214 Principles of Software Systems
Overview

• Goals of this course

• Interactions
  • Lectures, Recitations, etc.

• Assessment
  • Quizzes, Homeworks, Exams

• A mysterious function!
Lectures

• Tuesday and Thursday

• Please be here, please be active
  • Ask and answer questions, pay attention
  • Lecture notes after lecture

• Please be here, please be active
  • No surfing, email, games, ...
Recitations

• (Hello, TAs!)
• Wednesday and Friday, starting tomorrow
• Reinforce lecture material
• Problem solving
Unix/Tools Tutorial

- Tuesday-Thursday (this week) and Tuesday-Thursday (next week)
- 4:30pm & 5:30pm
- GHC 5201 & GHC 5205
- You need to attend for Assignment 0!
- Due next Thursday
Online Communication

• Blackboard for grades, quizzes, email announcements

• Piazza for questions & communication with course staff. Get help, help each other!

• Autolab: Programming homework

• Cluster Linux machines for assignments
Other Resources

• Course home page
  • http://www.cs.cmu.edu/~fp/courses/15122-f12
  • Schedule, lecture notes, calendar, contact info,...

• Office hours, tutoring

• C0 home page
  • http://c0.typesafety.net
  • Tutorial, reference, examples, binaries,...
Overview

• Goals of this course
• Interactions
  • Lectures, Recitations, Office Hours
• Assessment
  • Quizzes, Homeworks, Exams
• A mysterious function!
Assessment

- 10% - Quizzes on Blackboard
  - Check the course schedule. Due at midnight(!)
- 20% - Midterms (two of them, 10% each)
- 25% - Final
- 45% - Assignments (eight of them)
  - Combination of written and programming
  - Written due at lecture (no late days)
  - Online due at midnight, total of three late days, one per assignment
Academic Integrity

• Quizzes, exams, homework must be your own

• OK: discussion of course material, practice problems, study sessions

• Not OK: copying or discussing answers, looking at or copying code (even parts)

• Not OK: talking through the assignment as you code with a classmate
Overview

• Goals of this course
• Interactions
  • Lectures, Recitations, Office Hours
• Assessment
  • Quizzes, Homeworks, Exams
• A mysterious function!
Bug Report!

```c
int f(int x, int y) {
    while (x != y){
        if (x > y){
            x = x - y;
        } else {
            y = y - x;
        }
    }
    return x;
}
```