15-122
Principles of Imperative Computation

Spring 2011

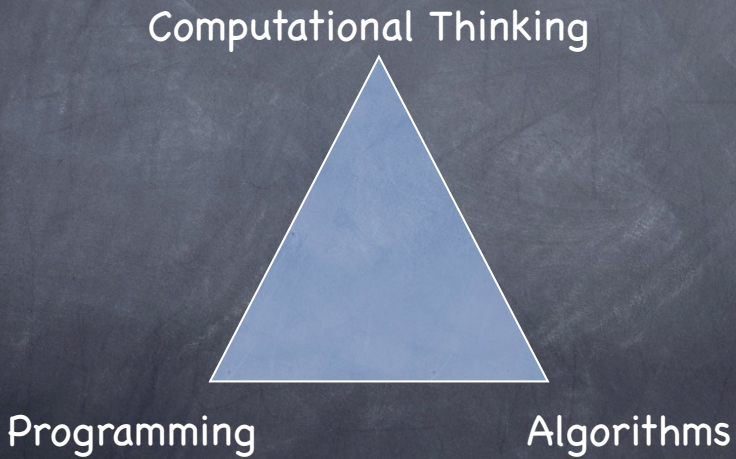
Frank Pfenning
Tom Cortina, Ananda Gunawardena, William Lovas,
Karl Naden, Jacob Potter
Jason Chow, Jason Koenig, Sri Raghavan,
Thomas Tuttle, Bill Zorn

http://www.cs.cmu.edu/~fp/courses/15122-s11

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 data structure invariants
- Reasoning about resource bounds

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

Programming Language

- C0: 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

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
- Randomness
- Dynamic programming
- Emphasis on imperative prog's, ephemeral data struct's

Concrete Algorithms

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

Role in Curriculum

- 15-150 Principles of Functional Programming
- 15-213 Introduction to Computer Systems
- 15-210 Fundamental Alg's & Data Struct's
- 15-214 Principles of Software Systems

Overview

- Goals of this course
- Interactions
  - Lectures, Recitations, Office Hours
- Assessment
  - Quizzes, Homeworks, Exams
- A Mysterious Function

Lectures

- Please be here, please be active
  - Ask and answer questions, pay attention
  - Lecture notes after lecture
- Laptops for note-taking only
  - No surfing, email, games, ...
  - Too distracting for everyone else

Recitations

- Reinforce lecture material
- Problem solving
- How-to programming and tool support
- Get to know your instructor

Office Hours

- We like to see you!
- Any questions and issues with course
- See web page for current hours and location
- Cluster help available Mon-Thu 6:30-9:30!

On-line Communication

- Blackboard for grades, quizzes, email announcements
- Bboard cyrus.academic.cs.15-122
- Email to me, TA, or CA
- Cluster Linux machines for assignments

Overview

- Goals of this course
- Interactions
  - Lectures, Recitations, Office Hours
- Assessment
  - Quizzes, Homeworks, Exams
- A Mysterious Function

Quizzes

- Test basic understanding
- On-line on Blackboard, auto-graded
- Due at midnight(!) (see schedule on-line)
- 8 quizzes, drop lowest score
- Total of 7 * 15 = 100+5 pts

Midterms

- Test functional understanding of material
- During lecture period (80 mins)
- Closed book, closed laptop, 1 sheet of notes
- Total of 2 * 100 = 200 pts

Final

- Testing cumulative mastery of material
- Three hours during final exam period
- Closed book, closed laptop, 1 sheet of notes
- Total of 250 points

Assignments

- Weekly assignment (see on-line schedule)
- Apply material in problem solving context
- Combination of written and programming
- Hand-in start of lecture (written) & online (prog.)
- Total of 3 late days on prog, none on written
  - Max of 1 late day per assignment
- Total of 7 * 50 + 1 * 100 = 450 pts

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 each others code (even parts)
- University policy will be applied rigorously!

Overview

- Goals of this course
- Interactions
  - Lectures, Recitations, Office Hours
- Assessment
  - Quizzes, Homeworks, Exams
- A Mysterious Function

Bug Report!

```
int f (int x, int y) {
  int r = 1;
  while (y > 1) {
    if (y % 2 == 1) r = x * r;
    x = x * x;
    y = y / 2;
  }
  return r * x;
}
```