CS/ENGRD 2110 Object-Oriented Programming and Data Structures

Fall 2012
Prof. Doug James

Lecture 1: Overview
http://courses.cs.cornell.edu/cs2110

1

Course Staff

- Instructor
  - Doug James (djames@cs.cornell.edu)
- Teaching Assistants
  - Konstaninos Mamouras (mamouras@cs.cornell.edu)
  - Mark Verheggen (mark@cs.cornell.edu)
  - more TAs are TBD
- Consultants
  - TBD

2

Course Staff

- Teaching Assistants
  - Lead sections ("recitations", "discussions") starting next week
  - Act as your main contact point
- Consultants
  - In Upson 360, hours online
  - "Front line" for answering questions
  - consulting hours start next week
- More info?
  - See website

3

What is wrong with this Program?

```
public class Mystery {
   public static void main(String[] args) {
      int[] a = {7, 121, 12, 13, 9, 324, 1};
      boolean d;
      do {
         d = false;
         for (int b = 1; b < a.length; b++) {
            if (a[b-1] > a[b]) {
               int c = a[b];
               a[b] = a[b-1];
               a[b-1] = c;
               d = true;
            }
         }
      } while (d);
      for (int e : a) {
         System.out.println(e);
      }
   }
}
```

➔ Output: "1, 7, 9, 12, 13, 121, 324"

4

Most Software Sucks

(see "Article on Software" on webpage)

5

Moore's Law

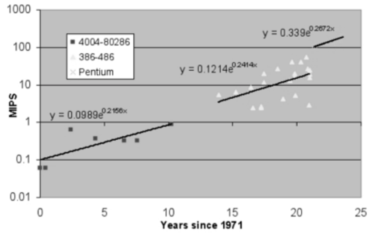


Figure 5: Processor performance in millions of instructions per second (MIPS) for Intel processors, 1971-1995.

From *Lives and death of Moore's Law*, Ilkka Tuomi, 2002

6

Grandmother’s Law

- Brain takes about 0.1 second to recognize your grandmother
  - About 1 second to add two integers (e.g. 3+4=7)
  - About 10 seconds to think/write statement of code
- Your brain is not getting any faster!

7

Why the world need CS 2110!

- Real systems are large and complex.

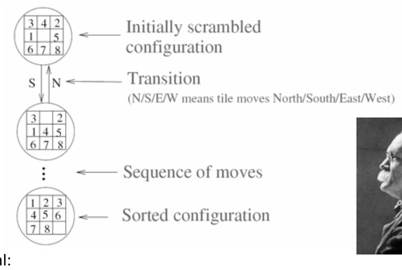
Year	*Operating System*	*Millions of lines of code**
1993	Windows NT 3.1	6
1994	Windows NT 3.5	10
1996	Windows NT 4.0	16
2000	Windows 2000	29
2001	Windows XP	40
2005	Windows Vista	50

*source: Wikipedia

- Computer Science ➔ Managing Complexity
  - Analyze highly complex situations
  - Decompose problem into independent components
  - Assure correctness of components
  - Reuse prior work that is proven correct
  - Spread work over multiple people

8

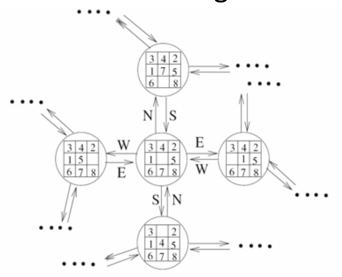
“Sam Loyd’s 8 Puzzle”


Initially scrambled configuration
S
N
Transition
(N/S/E/W means tile moves North/South/East/West)
Sequence of moves
Sorted configuration

- Goal:
  - Given an initial configuration of tiles, find a sequence of moves that will lead to the sorted configuration.
- A particular configuration is called a state of the puzzle.

9

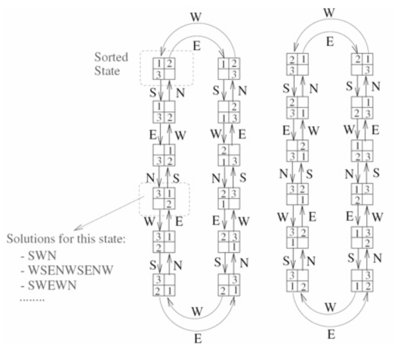
State Transition Diagram of 8-Puzzle



- State Transition Diagram: picture of adjacent states.
- A state Y is adjacent to state X if Y can be reached from X in one move.

10

State Transition Diagram for a 2x2 Puzzle



11

Graphs

- State Transition Diagram in previous slide is an example of a graph: a mathematical abstraction
  - vertices (or nodes): the puzzle states
  - edges (or arcs): connections between pairs of vertices
  - vertices and edges may be labeled with some information (name, direction, weight, cost, …)
  - Graphs: vocabulary/abstraction for problems
  - Airline routes, roadmaps, …
  - Polygon meshes
  - Social networks
  - etc.

12

Path Problems in Graphs

- Is there a path from node A to node B?
  - Solve the 8-puzzle
- What is the shortest path from A to B?
  - 8-puzzle (efficiently)
  - Driving directions
- Network flow
  - Friendship structure of facebook
- Eigenvectors
  - Pagerank in Google

13

Simulating the 8-puzzle

- What operations should puzzle objects support?
- How do we represent states?
- How do we specify an initial state?
- What algorithm do we use to solve a given initial configuration?
- How should we present information to the user? (GUI design)
- How to structure the program so it can be understood, maintained, upgraded?

14

Course Objectives

- Concepts in modern programming languages
  - recursive algorithms and data structures
  - data abstraction, subtyping, generic programming
  - frameworks and event-driven programming, graphical user interfaces

  → Organizing large programs
- Building blocks: data structures and their algorithms
  - arrays, lists, stacks, queues, trees, hashtables, graphs
- Algorithm analysis and designing for efficiency
  - asymptotic complexity, induction

Using Java, but not a course on Java!

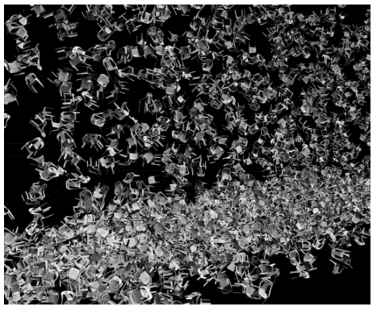
15

Why you need CS 2110?

- Fun and intellectually interesting: cool math ideas meet engineering (and make a difference)
- Crucial to any engineering or science career
  - Good programmers are 10x more productive
  - Leverage knowledge in other fields, create new possibilities
  - Where will you be in 10 years?
- Great job prospects with high salaries…
- Computational Thinking: You'll learn to think in a more logical, structured way
- Computational thinking pervades almost every subject of inquiry in today's world

16

Graphics, Simulation & Games
(arrays, lists, stacks, queues, trees, hashtables, graphs, and algorithms galore!)



[James & Pai, SIGGRAPH 2004]

17

Are you ready for CS2110?

- CS2110 assumes you know Java
  - You took CS1110 at Cornell
  - You have "completed" CS1130
  - Or took a high school course and got a 4 or 5 on the CS AP exam
- CS2110 assumes you actually remember Java
  - Go over online material of CS1130
  - classes, objects, fields, methods, constructors, static and instance variables, control structures, arrays, strings, exposure to inheritance
- Don't take CS1110 just because you are worried that your high school Java experience won't do
- *We recommend against trying to skip directly into CS3110.*

18

And how about CS2112?

- CS2112 is equivalent to CS2110 in terms of
  - requirements
  - prerequisites
  - course material that is covered
- CS2112 is different in the following respects
  - gives more depth in some areas
  - is more project driven
  - more challenging assignments
  - includes a lab in addition to sections (→ 4 credits)
-

the first 3 weeks. Same course time.

Lectures

- Time and place
  - Tuesday/Thursday 10:10-11am, Olin 155
  - Attendance is mandatory
  - In-class quizzes
- ENGRD 2110 or CS 2110?
  - Same course! We call it CS 2110
  - Non-engineers sign up for CS 2110
  - Engineers sign up for ENGRD 2110
- Reading and examples will be posted online together with lecture notes

Sections

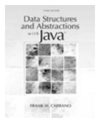
- Each section will be led by a TA
  - Reinforces lecture material, help on homework
  - Sometimes additional material
- Everybody needs to register for a section
  - Section numbers are different for CS and ENGRD
  - Like lecture, attendance is mandatory
  - No permission needed to switch sections
  - We recommend that you do NOT switch often
- You may attend more than one section
- No sections this week – they start next week!

Consulting and Office Hours

- Office Hours
  - Prof. James (after class, Upson 5146)
  - Teaching Assistants
  - See webpage for times and locations
- Consulting Hours
  - Google calendar on webpage
  - In Upson 360
  - "Front line" for answering questions
  - Consulting hours start next week

Resources

- Course web site
  - http://courses.cs.cornell.edu/cs2110
- Piazza
  - Good place to ask questions
  - Announcements
- Textbook
  - Frank M. Carrano, Data Structures and Abstractions with Java, 3rd ed., Prentice Hall
  - Additional material on the Prentice Hall website
- Recorded Videonote Lectures Fall 2010 (Birman)
  - Warning: Different instructor, different content



Academic Excellence Workshops

- Two-hour labs in which students work together in cooperative setting
- One credit S/U course based on attendance
- Time and location TBA
- See the website for more info http://www.engineering.cornell.edu/student-services/learning/academic-excellence-workshops

/

Obtaining Java

- See “Resources” on website
- We recommend Java 6
- Need Java Development Kit (JDK), not just Java Runtime Environment (JRE)

25

Eclipse

- IDE: Interactive Development Environment
  - We highly recommend use of Eclipse
  - Helps you write/compile your code
  - Helps with debugging
  - Eclipse tutorial in section
- See “Resources” on website

26

Coursework

-
  - Five assignments (43%)
  - Two prelims (15% each)
  - Final exam (20%)
  - Course evaluation (1%)
  - Survey (1%)
  - Quizzes in class (5%)
- Submit assignment late:
  - 5 points deduction per 24h late.
  - Everybody has 5 “free” late days.
  - Maximum 3 days late on each assignment
- For assignments and quizzes, lowest grade gets replaced by second-lowest grade.

27

Assignments

- Assignments may be done by teams of two students (except for A1)
  - You may choose to do them by yourself
  - A1 will be posted on Thursday
- Finding a partner
  - Choose your own or contact your TA.
  - Piazza may be helpful.
  - Monogamy encouraged. However, you may change partners between assignments (but not within).
- Please read partner info and Code of Academic Integrity on website

28

Survey

- Soon available on CMS as a “quiz”
- Learn about course participants
  - Understand better who you are
  - Refine CS2110 content
- Participating accounts for 1% of overall grade
  - Obviously not graded
  - There are no wrong answers
- Deadline: next week Friday, Feb 3.

29

Academic Integrity

- See Academic Integrity Policy on website
- We use artificial intelligence tools to check each homework assignment
  - The software is very accurate!
  - It tests your code and also notices similarities between code written by different people
- Sure, you can fool this software
  - … but it’s easier to just do the assignments
  - Penalty ranges from negative points for the assignment to failing the course.

30

Welcome!

We hope you have fun, and enjoy programming as much as we do.

31