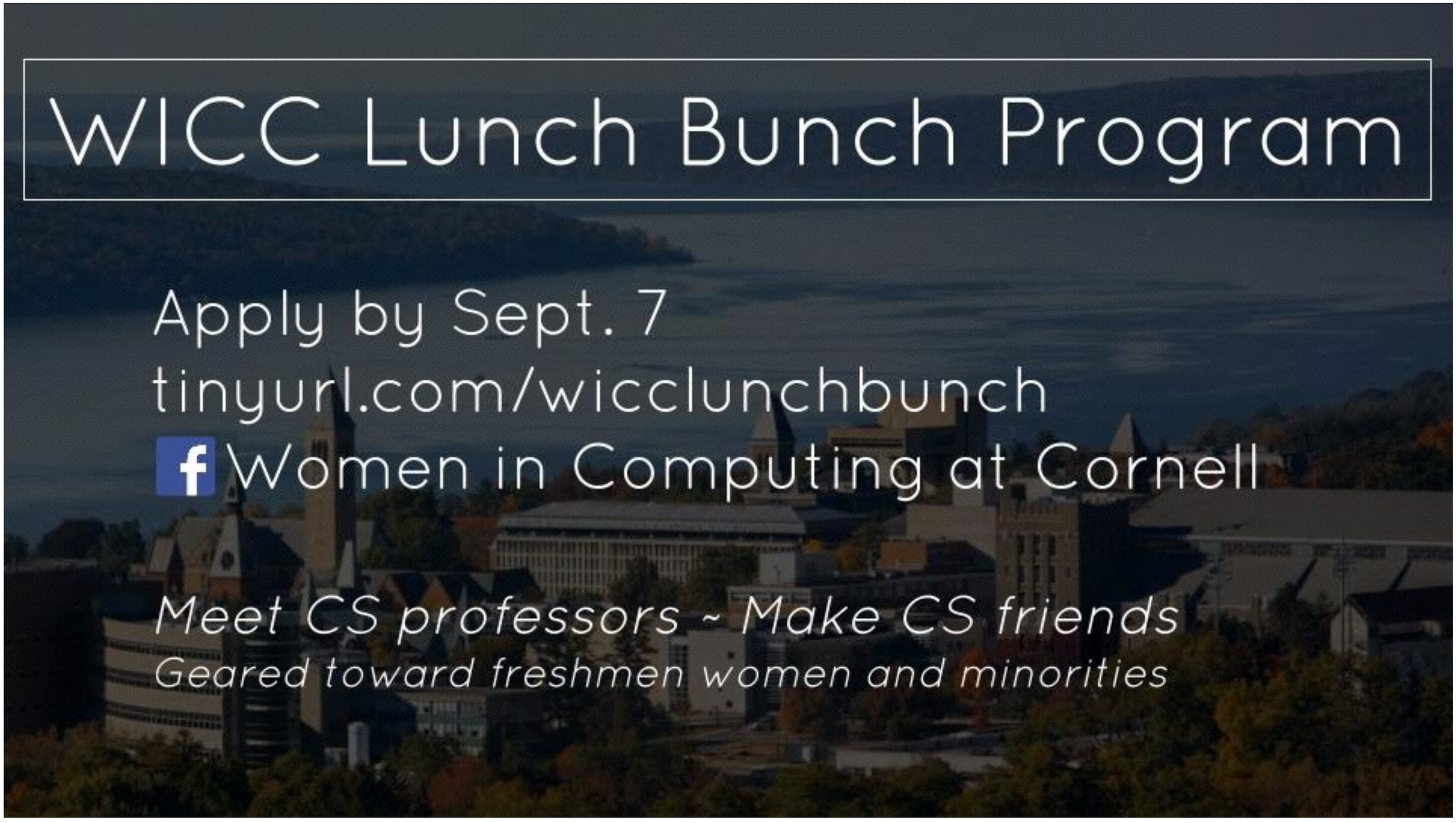
CS 2112 / ENGRD 2112
Object-Oriented Design and Data Structures — Honors

Fall 2016

Dexter Kozen

Cornell University

Lecture 1: Introduction


WICC Lunch Bunch Program
Apply by Sept. 7
tinyurl.com/wicclunchbunch
Women in Computing at Cornell
Meet CS professors ~ Make CS friends
Geared toward freshmen women and minorities

Course staff

- Instructor: Dexter Kozen
- Grad TAs: Hubert Lin, Stephen McDowell, Maheer Iqbal
- Undergrad staff:

Zander Bolgar

Jake Chen

Agi Csaki

Jane Du

Jacob Glueck

Harry Goldstein

Matt Habel

Sumner Hearth

Alex Renda

Lucas Silver

Jesse Yuan

What it's about

An introduction to computer science and software engineering

- **Programming language features**
  - –data abstraction, subtyping, generic programming
  - –concurrency and threads
  - –*Not a course about Java!*
- **Object-oriented design — organizing large programs**
  - –specifications
  - –design patterns
  - –frameworks and event-driven programming
- **Data structures and algorithms**
  - –recursive algorithms and data structures
  - –algorithm analysis and designing for efficiency
  - –asymptotic complexity, induction
  - –arrays, lists, stacks, queues, trees, graphs, hash tables

Web site

- Your source for information:

  `http://courses.cs.cornell.edu/cs2112`

  – Lecture notes: you are expected to read
    - mostly *not* slides!
    - may not include *everything* covered in lecture
    - may include extra material not covered in lecture
    - often updated after the lecture
  – Assignments
    - may be updated after initial release
  – Pointers to resources

Communicating with staff

- Best: use Piazza to post questions
  - `http://piazza.com/class#fall2016/cs2112`
  - –Answering other questions (well) is good karma
  - –Watch out for violating academic integrity!
- Course announcements posted to Piazza (or emailed to all students if urgent)
- My office hours: Gates 436, times TBA
- Consultants–hours online, location TBA
  - –"Front line" for answering questions – consulting hours start today

CMS

- Assignments will be submitted to CMS
- Grades, solutions will be posted on CMS
  `https://cms-b.csuglab.cornell.edu`
- Regrade requests should be posted to CMS soon after receiving grade (remind us if necessary…)

Meetings

- **Lectures**: TTh 10:10-11, Gates G01
- **Discussion sections** (attend **1** per week)
  - –T 12:20-1:10 (Gates G01)
  - –W 1:25-2:15 (Hollister 110)
- **Labs** (attend **1** per week)
  - –Monday 7:30-8:20 (Phillips 219)
  - –Wednesday 7:30-8:20 (Hollister 110)
- 4 hours per week — attendance required

Room caps

- If the lecture, lab, or recitation is full…
  - –don't panic, you can still take the course
  - –let us know, including your constraints
  - –we *will* work out a solution

Assignments

- 7 assignments
  - –mostly programming but some written problems
  - –total to 40% of total score
- First 3 assignments done solo
- Final project (last 4 assignments) with a partner
- Assignment late penalties:
  - –1 day late: –10%
  - –2 days late: –20%
  - –3 days late: –30%
  - –weekends count as 1 day

Exams and more

- One evening prelim: Sept 29, 7:30pm, Gates G01
  - –20% of total score
- Final exam: sometime during Dec 7–15, location and time TBA
  - –35% of total score
- 5% of score:
  - –participation (in-class, Piazza, course evals)
  - –possible in-class quizzes

Labs

- Programming exercises, solve problems, learn about tools
- Bring a laptop if you have one (or share if you don't—you will work in pairs anyway)
- First lab: Wednesday in Hollister 110 (set up Eclipse and do some programming) — attend if you can even if you are normally in the Monday lab

Textbook

- Data Structures and Abstractions with Java, 4th ed., Frank M. Carrano and Timothy Henry, Pearson Education, 2014.
  - –Available at Campus Store
  - –On reserve in library
  - –Recommended, not required
  - –Also the 2110 textbook
  - –Not heavily used—can share with a friend or two
  - –Earlier editions are probably ok

Academic integrity

- You must never misrepresent someone else's work as your own or let others misrepresent your work as theirs
  - Copying code or answers is never okay
  - Letting others copy you is also a violation
  - You must be able to explain your answers fully
    - Discussions with others are definitely fine if they could have happened in a lightless room
- We will use *highly* effective tools for detecting plagiarism
- Report any discussions about assignments and any use of external code
- Our goal: spend time on course content

Social integrity

Everyone is to be treated with respect, regardless of background, experience, religion, ethnicity, citizenship, gender, or sexual orientation

If you are made to feel unwelcome or disrespected, please contact Dexter

If you become aware of anyone else being made to feel unwelcome or disrespected, please encourage them to contact Dexter

CS 2112 or ENGRD 2112?

Doesn’t matter

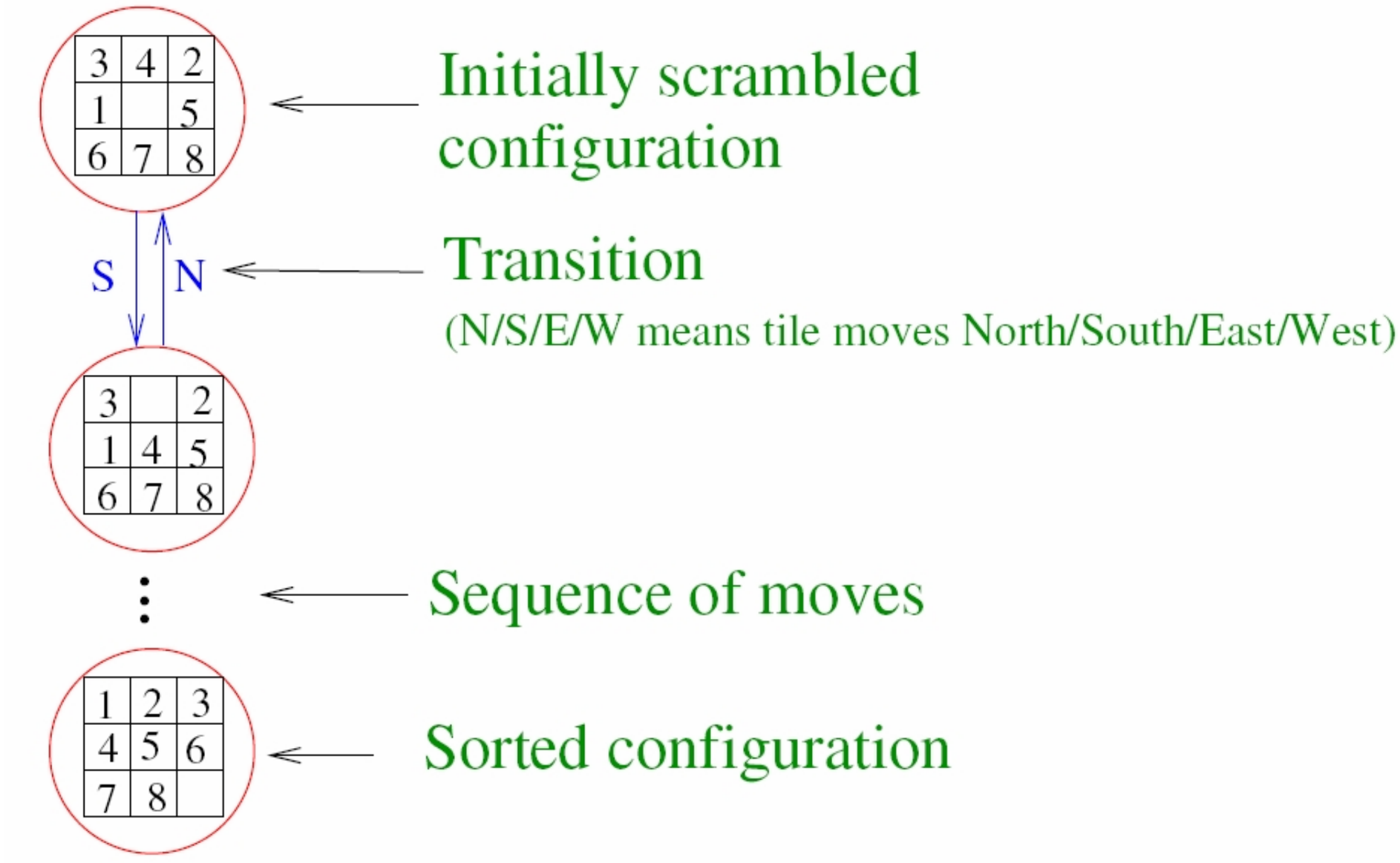
CS/ENGRD 2110 or CS/ENGRD 2112?

- 2112 is an 'honors' version of 2110.
  - –aimed at CS majors: smaller
  - –harder and more interesting assignments including a final project spanning 4 assignments
  - –more material
    - e.g., more algorithms and their analysis (theory)
    - e.g., more about design and design patterns (practice)
  - –more credits (4 vs 3)

2110 vs. 2112

- Warning: you will be challenged here

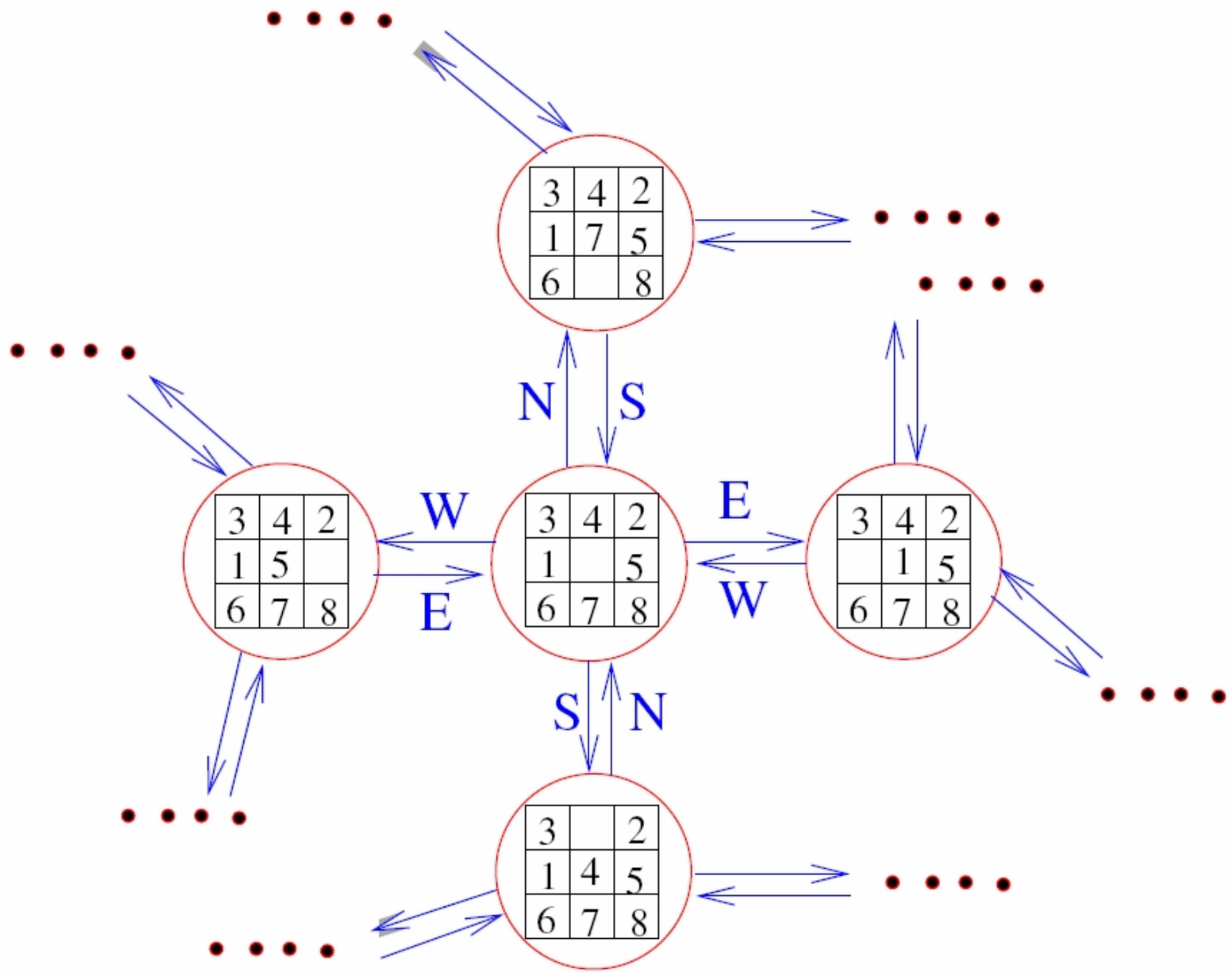
Sam Loyd's 8 Puzzle



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.

State Transition Diagram of 8-Puzzle

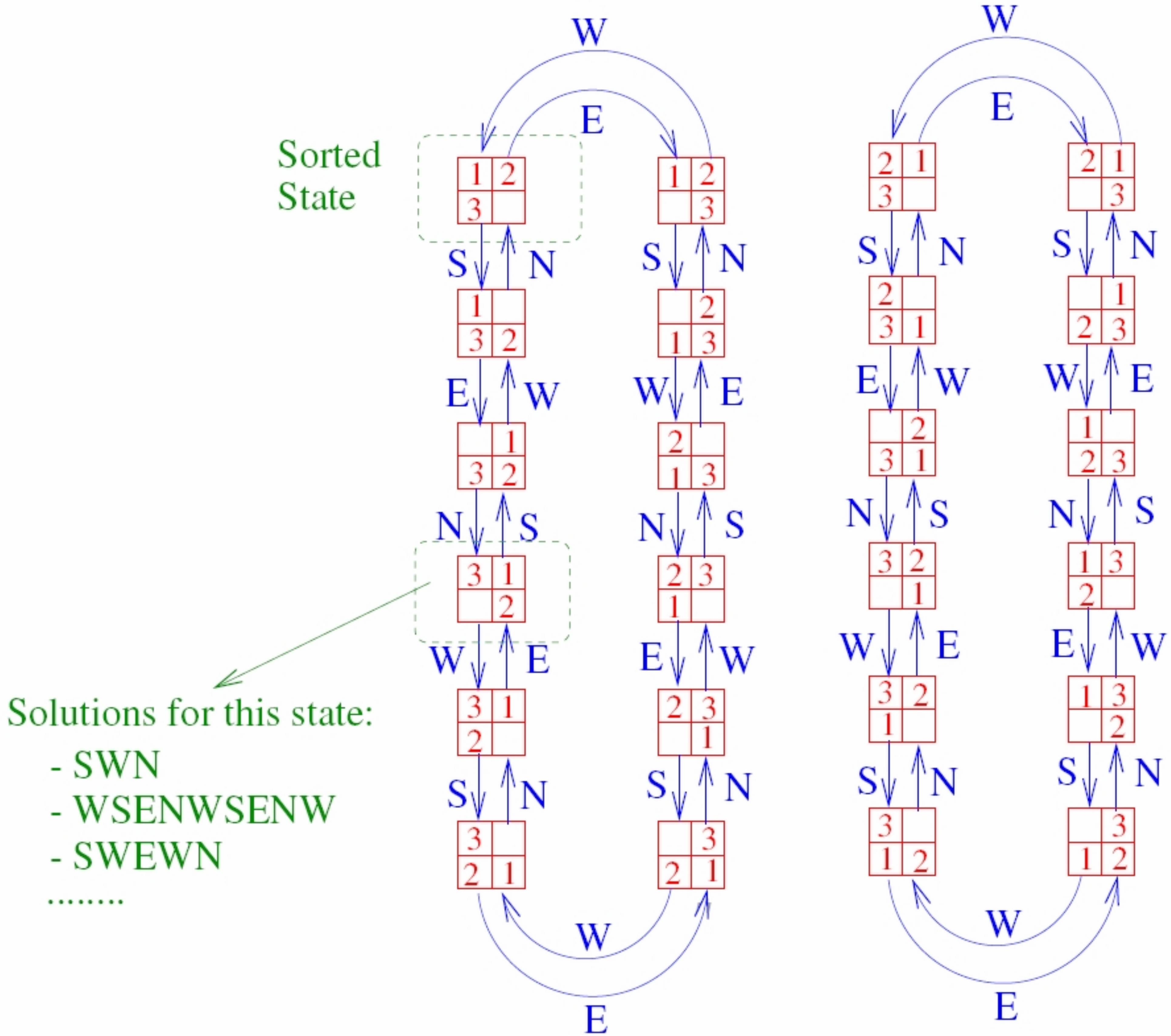


State Transition Diagram: picture of adjacent states.

A state Y is ***adjacent*** to state X if Y can be reached from X in one move.

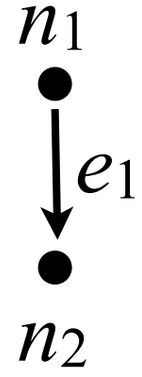
20

State Transition Diagram for a 2x2 Puzzle



Graphs

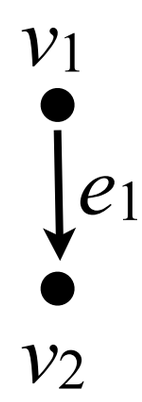
- State transition diagram in previous slide is an an example of a graph: a mathematical abstraction
  - –nodes (or vertices) : the puzzle states
  - –edges (or arcs) : the transitions, possibly labeled



- Graphs are all around us : airline routes, roadmaps, org charts, pipelines, …

Graph algorithms

- Large toolbox of efficient **algorithms** for graphs help us solve problems:
  - –searching for best nodes/shortest paths
  - –finding maximum flow through graph
  - –minimum spanning trees
  - – . . .



- And known **hardness results** (e.g., finding Hamiltonian cycles) tell you what you **can't** solve.

Software design choices

- What operations should puzzle objects have?
- How do we represent states? The initial state? Transitions?
- How do we present information to the user and support interaction?
- How do we break the coding up into parts that can be coded independently?
- How to structure code so it can be maintained, upgraded?

Why you need CS 2112

- Data structures and algorithms to solve problems efficiently and effectively
- Design techniques to produce code that works quickly and keeps working
- Computer science:
  - algorithms, data structures, programming languages, design principles, knowledge of what is possible and feasible
- Good programmers have more fun!
  - 10x more productive
  - better able to adapt, grow, see opportunities, change the world

Next steps

- Get your Piazza account set up ASAP
- Keep an eye on the 2112 website
- Download the first programming assignment, released today, due in a week
- Make sure you have Eclipse downloaded and working — see consultants for help
- Come to lab on Wednesday for help getting started
- Have fun!