Welcome to CSE21!

CSE21 Fall 2016, Day 1

September 23, 2016

http://cseweb.ucsd.edu/classes/fa16/cse21-ab/
About this course

Formulate & solve problems

Describe data

Analyze algorithms

Using math
About this course

Why is math part of the CS curriculum?

Proofs: key to convincing arguments, but also key part of software engineering

Vocabulary: basic language of Computer Science

Quantitative Analysis: are our solutions / programs / algorithms good enough? How many computational resources (time, memory, power) does our solution use?
About you

Have you used iClickers before?

A. Yes
B. No

To change your remote frequency:

1. Press and hold power button until flashing
2. Enter two-letter code BB
3. Checkmark / green light indicates success

BB
Did you take CSE 20 at UC San Diego?

A. Yes.
B. No, I took Math 15A instead.
C. No, I took an equivalent course elsewhere.
D. No, for some other reason.
Introductions
What do we assume you know?

Rosen Chapters 1, 2, some of 5, some of 9.

More precisely: You can describe algorithms and their correctness using precise mathematical terminology and techniques. For example:

• Sets, relations (equivalence relations, orders)
• Logical equivalence, conditionals, hypotheses, conditionals, contrapositives
• Universal and existential quantifiers
• Proof by contradiction (indirect proof)
• Proof by induction
• Algorithm invariants
Logistics, part 1


Participation: Lecture (iClicker questions)
Discussion (quizzes)

Exams:  
First Exam: Weds, Oct 19
Second Exam: Weds, Nov 16
Final Exam: A00: Mon, Dec 5 (8-11 AM)
            B00: Wed, Dec 7 (8-11 AM)
Logistics, part 1

Student accounts are **automatically billed** for the book. You can choose to **opt out before Oct 8, 11:59pm**, otherwise you will be charged. **Free access until then.**

**Textbook:** Rosen 7th Edition, available through TritonEd

**Participation:** Lecture (iClicker questions)
Discussion (quizzes)

**Exams:**
- **First Exam:** Weds, Oct 19
- **Second Exam:** Weds, Nov 16
- **Final Exam:**
  - **A00:** Mon, Dec 5 (8-11 AM)
  - **B00:** Wed, Dec 7 (8-11 AM)
Logistics, part 2

Class Website:
Homework assignments, syllabus, calendar, announcements, study guides, office hour info, lecture slides (available the day after lecture.)

Gradescope:
Homework submission and exam return.

TritonEd:
Participation scores.

Piazza:
Announcements and Q&A. Contact instructors here! No HW questions online!

Office hours:
Instructor and some TAs and tutors. Weekdays and weekends. Discuss HW questions here in person!

Podcasts:
Listen to recordings of the lectures.
Logistics, part 3

Exams (60%), HW (35%), Participation (5%)

Details on class website

* **Exams**: Can use note sheet, drop lowest midterm score if do better on final
* **HW**: 8 HWs, drop lowest HW score
* **Participation**: Max of lecture and discussion
  * **Lecture**: credit if you answer 80% of clicker question in that day's class
  * **Discussion**: credit if you attend and take a quiz

HW and exam answers evaluated not only on the correctness of your answers, but on your ability to effectively communicate your ideas and convince the reader of your conclusions through proofs and logical reasoning.
Academic Integrity Scenarios

You’re working on a homework question and run across a definition you don’t understand. You Google the term and the first hit includes a full solution to the homework question. You avoid reading the solution and close the browser. You keep working on the solution and hand in the assignment, without mentioning the Google search since you didn’t use the result. Is this acceptable?

A. Yes    B. No
Academic Integrity Scenarios

You form a study group with two friends and start working on the next homework. Since there are 6 questions you each pick two questions, think about them, and write out your solutions in a shared Google doc. You glance over each other's work before turning in the assignment. Is this acceptable?

A. Yes  B. No
See Piazza for a list of things you should do right away.
Goal: Learn Concepts

Concepts which computer science relies upon:

- Algorithms
- Asymptotic notation
- Recurrence relations
- Graphs
- Enumeration and data representation
- Probability
An example of CS vocabulary: Trees

**Data structure:** Binary search trees

Stay tuned: Chapter 11 in Rosen, Week 6
Algorithm: parsing
An example of CS vocabulary: Trees

**Model:** possible paths of computation
An example of CS vocabulary: Trees

**Model:** Phylogenetic (evolutionary) tree
**State space**: possible configurations of a game
An example of CS vocabulary: Trees

Conclusion: Many different applications but same underlying idea.

- How do we define a tree?
- What properties are guaranteed by this definition?
- What algorithms can exploit these properties?
Goal: Solve Problems

- Come up with *new* algorithms
- Think of the homework questions as puzzles that you need to unravel: the solution or even the approach won't be clear right away.
- You can work on homework in groups of 1-3 students.
* Assume elements of the set to be sorted have some underlying order
Which of the following collections of elements is listed in sorted order?

A. 42, 10, 30, 25
B. 10, 25, 30, 40
C. 40, 30, 25, 10
D. All of the above
E. None of the above
Why sort?

A TA facing a stack of exams needs to input all 400 scores into a spreadsheet where the students are listed in alphabetical order.

OR

You want to find all the duplicate values in a long list.
A TA facing a stack of exams needs to input all 400 scores into a spreadsheet where the students are listed in alphabetical order.

OR

You want to find all the duplicate values in a long list.

It's easier to access data when it is sorted because you know exactly where to find it.
1. **Find a group** of about 20 people nearby. Write your first names on separate papers.

2. **Sort** the names of the people in your group alphabetically by first name.

3. **Discuss as a group** the strategy you used to sort the papers, and how you might describe it to someone else.

4. **Write** a clear English description of the strategy your group used (each person should do this.)

5. **Select** one person in the group to turn in their English description.
Discussion of Sorting Algorithms

- Is the strategy clear?

- Will the strategy always work?

- Does the strategy scale well to bigger groups?
General questions to ask about algorithms

1) **What** problem are we solving?
2) **How** do we solve the problem?
3) **Why** do these steps solve the problem?
4) **When** do we get an answer?
General questions to ask about algorithms

1) **What** problem are we solving?  **PROBLEM SPECIFICATION**

2) **How** do we solve the problem?  **ALGORITHM DESCRIPTION**

3) **Why** do these steps solve the problem?  **CORRECTNESSS**

4) **When** do we get an answer?  **RUNNING TIME PERFORMANCE**
Given a list

\[ a_1, a_2, \ldots, a_n \]

rearrange the values so that

\[ a_1 \leq a_2 \leq \ldots \leq a_n \]

Values can be any type (with underlying total order). For simplicity, use integers.
Your approaches: HOW
Selection Sort (Min Sort)

"Find the first name alphabetically, move it to the front. Then look for the next one, move it, etc."
procedure selection sort($a_1, a_2, \ldots, a_n$: real numbers with $n \geq 2$)

for $i := 1$ to $n-1$
    $m := i$
    for $j := i+1$ to $n$
        if ($a_j < a_m$) then $m := j$
        interchange $a_i$ and $a_m$

{ $a_1, \ldots, a_n$ is in increasing order}
Bubble Sort

"Compare the first two cards, and if the first is bigger, keep comparing it to the next card in the stack until we find one larger than it. Repeat until the stack is sorted."
procedure bubble sort(a₁, a₂, ..., aₙ: real numbers with n ≥2 )
for i := 1 to n-1
    for j:= 1 to n-i
        if ( aⱼ > aⱼ₊₁ ) then interchange aⱼ and aⱼ₊₁

{ a₁, ..., aₙ is in increasing order}
"We passed the cards from right to left, each individual inserting their own card in the correct position as they relayed the pile."
procedure insertion sort(a₁, a₂, ..., aₙ: real numbers with n >=2 )
for j := 2 to n
    i := 1
    while aⱼ > aᵢ
        i := i+1
    m := aⱼ
    for k := 0 to j-i-1
        aⱼ-k := aⱼ-k-1
        aᵢ := m

{ a₁, ..., aₙ is in increasing order}
"Call out from A to Z, collecting cards by first letter. If there are more than one with the same first letter, repeat with the second letter, and so on."
Bucket Sort – Pseudo pseudo code

• Create empty buckets that have an ordering.
• Put each of the elements of the list into the correct bucket.
• Sort within each bucket.
• Concatenate the buckets in order.
"We split into two groups and organized each of the groups, then got back together and figured out how to interleave the groups in order."
Merge Sort – Pseudo pseudo code

• If the list has just one element, return.
• Otherwise,
  • Divide list into two pieces:
    \[ L_1 = a_1 \ldots a_{n/2} \text{ and } L_2 = a_{n/2+1} \ldots a_n \]
  • \( M_1 = \text{Merge sort ( } L_1 \text{ )} \)
  • \( M_2 = \text{Merge sort ( } L_2 \text{ )} \)
  • Merge the two (sorted) lists \( M_1 \) and \( M_2 \)
Others?

Bogo sort

Quick sort

Binary search tree traversal

Why so many algorithms?
Why so many algorithms?

Practice for homework / exam / job interviews.

Some algorithms are better than others. Wait, better?
Reminder

See Piazza for a list of things you should do right away.
Summer Internship Symposium
http://cseweb.ucsd.edu/~minnes/cse197/symposium.html

Wednesday Sep 28th
CSE 1202
10:30am-4pm