Welcome to CSE21!

<table>
<thead>
<tr>
<th>Lecture</th>
<th>Instructor</th>
<th>Days</th>
<th>Time</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lecture A</td>
<td>Tiefenbruck</td>
<td>Tu, Th</td>
<td>11am-12:20pm</td>
<td>Center 119</td>
</tr>
<tr>
<td>Lecture B</td>
<td>Tiefenbruck</td>
<td>Tu, Th</td>
<td>9:30am-10:50am</td>
<td>Center 119</td>
</tr>
<tr>
<td>Lecture C</td>
<td>Minnes</td>
<td>Tu, Th</td>
<td>3:30pm-4:50pm</td>
<td>WLH 2005</td>
</tr>
</tbody>
</table>

http://cseweb.ucsd.edu/classes/fa15/cse21-abc/
About this course

Formulate & solve problems

Describe data

Analyze algorithms

Using math
What math?

Proofs: key to convincing arguments

Vocabulary: basic language of Computer Science

Approximations: are our solutions / programs / algorithms good enough?
Have you used iClickers before?

A. Yes
B. No

CENTR119: BB    WLH2005: AB

To change your remote frequency
1. Press and hold power button until flashing
2. Enter two-letter code
3. Checkmark / green light indicates success
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 at my community college / another university
D. No, for some other reason.

CENTR119: BB  WLH2005: AB

To change your remote frequency
1. Press and hold power button until flashing
2. Enter two-letter code
3. Checkmark / green light indicates success
About you

What other CSE class are you taking this quarter?

A. None.
B. CSE 12.
C. CSE 11.
D. CSE 8B.
E. Some other CSE class.

CENTR119: BB  WLH2005: AB

To change your remote frequency
1. Press and hold power button until flashing
2. Enter two-letter code
3. Checkmark / green light indicates success
Introductions
What do we assume you know?

Short answer: HW 1.

 Longer answer: Rosen Chapters 1, 2, some of 5, some of 9.

 Longest answer: 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

**Textbook:** Rosen 7th Edition

**Participation:** Class times (iClicker questions) & discussion (quizzes)

[https://www1.iclicker.com/register-clicker/](https://www1.iclicker.com/register-clicker/)  
[http://sections.ucsd.edu](http://sections.ucsd.edu)

**Exams:**
- Tuesday October 20
- Tuesday November 17
- Saturday December 5

**Gradescope:** Homework submission and exam return

**Piazza:** announcements and Q&A. Contact instructors here!

**Office hours:** instructors and tutors. Discuss HW questions here!

**No makeup exams**
Logistics, part 2

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

* Details on class website: http://cseweb.ucsd.edu/classes/fa15/cse21-abc/
* Drop lowest HW score
* Drop lowest midterm score if do better on final
* Can use note sheet for exams
* Participation earned via either class participation or discussion quizzes
* Drop lowest discussion quiz score
* Credit for participation if answer 80% of clicker question in that day's class
* Drop two lowest class participation scores
* HW and exams 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.
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’re not sure if you are interpreting a homework problem correctly. You write a post on Piazza showing what you did to answer it, and asking if this is the correct way of interpreting the question. Is this acceptable?

A. Yes     B. No
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
1. **Learn 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 5
Algorithm: parsing
Model: possible paths of computation

Automaton

Example Inputs
An example of CS vocabulary: Trees

**Model:** Phylogenetic (evolutionary) tree
State space: possible configurations of a game
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?
2. 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.
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.

It's easier to access data when it is sorted because you know exactly where to find it.
Sort all of the people in your group alphabetically by first name.

Describe algorithm!
Instructions

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

Select one representative to write your group's strategy on the board.
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?  
   SPECIFICATION

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

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

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
"Find the first name alphabetically, move it to the front. Then look for the next one, move it, etc."
Selection Sort (MinSort) Pseudocode

Rosen page 203, exercises 41-42

\begin{verbatim}
procedure selection sort(a_1, a_2, ..., a_n: real numbers with n >=2 )
for i := 1 to n-1
    m := i
    for j:= i+1 to n
        if ( a_j < a_i ) then m := j
    interchange a_i and a_m

{ a_1, ..., a_n is in increasing order}
\end{verbatim}
"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($a_1, a_2, ..., a_n$: real numbers with $n \geq 2$)
for $i := 1$ to $n-1$
  for $j := 1$ to $n-i$
    if ($a_j > a_{j+1}$) then interchange $a_j$ and $a_{j+1}$

{ $a_1, ..., a_n$ 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."
**Insertion Sort Pseudocode**

Rosen page 198 (except small modification)

```plaintext
procedure insertion sort(a₁, a₂, ..., aₙ: real numbers with n >=2 )
for j := 1 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

Rosen page 196, 367-370

• If the list has just one element, return.
• Otherwise,
  • Divide list into two pieces:
    \[ L_1 = a_1 \ldots a_{n/2} \quad \text{and} \quad L_2 = a_{n/2+1} \ldots a_n \]
  • \( M_1 = \text{Merge sort} \left( L_1 \right) \)
  • \( M_2 = \text{Merge sort} \left( L_2 \right) \)
  • 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*?
From "How" to "Why"

What makes this algorithm work?

How do you know that the resulting array will be sorted?

*For loop-based algorithms:*

What's the effect of each loop iteration on the array?

Have we made progress?
A **loop invariant** is a property that remains true each time the body of a loop is executed.

For an iterative algorithm:
- Look for a loop invariant
- Show that when the loop is finished, the invariant guarantees that we've reached a solution
Selection Sort (MinSort) Correctness: WHY

**Loop invariant:** After the $k^{th}$ time through the outer loop, the first $k$ elements of the list are the $k$ smallest list elements in order.

If this is true, the program is correct. Why?

How can we show that this loop invariant is true?
Selection Sort (MinSort) Correctness: WHY

Loop invariant: After the $k^{\text{th}}$ time through the outer loop, the first $k$ elements of the list are the $k$ smallest list elements in order.

If this is true, the program is correct. Why?

How can we show that this loop invariant is true?

Induction 😊
Induction variable (k): the number of times through the loop.

Base case: k=0, before the loop

Inductive step: Let k be a positive integer.

**Induction hypothesis:** Suppose the invariant holds after k-1 times through the loop.

**Need to show** that the invariant holds after k times through the loop.
Next Time…

• Counting comparisons for a sorting algorithm
• Does sorting really help?
• Two searching algorithms
Reminders

Read syllabus on class website.
Enroll in Piazza and Gradescope
Register iClicker.
Sign up for discussion section.

HW 1 due **Monday 11:59pm**.

Discussion sections start on Monday: sign up starting 7pm today.