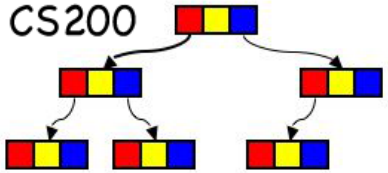
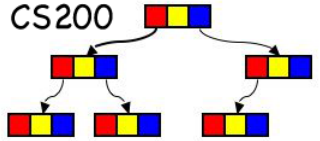


CS200 Fall 2016
Data Structures and Algorithms

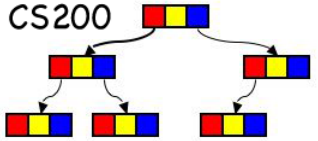


We live in the information age – fueled by computers.
An unprecedented amount of information is freely available.
How many of you have smart phones?
What apps/information do you store, manage and use on a daily basis on that phone?

This course is about the fundamentals of how that information is stored, managed and used
-- the theory and practice of representing and manipulating information

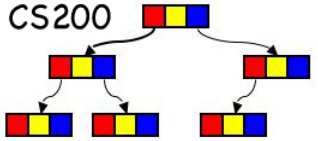
"scientia est potentia"
(knowledge is power)
Sir Francis Bacon or Thomas Hobbes

Class meetings



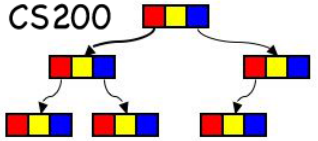
- Lectures
  - Concepts, programming assignment introduction, quizzes, tests.
- Recitations
  - Help with programming and written assignments, practice skills, reinforce/supplement material from lecture, a few programming quizzes.
  - Credit for attending and participating in recitations

Difference from CS160/161



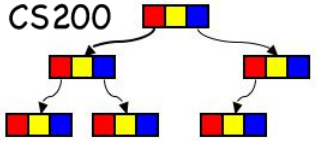
- Data structures and algorithms oriented
  - Complexity and efficiency (Orders of Magnitude) come into play
- Larger program developed incrementally over a number of assignments
  - More freedom in how to structure your program

Grading



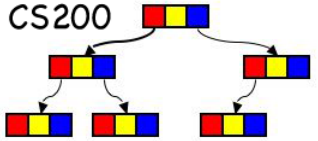
Programming assignments	20%
Written assignments	10%
Quizzes	10%
Recitations	10%
Midterm	25%
Final	25%

More Grading Specifics



- Exams:
  - Make-ups or reschedules for extreme circumstances only **inform us in advance!**
  - Written component in lecture on specified date
    - Closed book
  - Preparation for exam:
    - lectures notes
    - recitations
    - quizzes
    - written home works

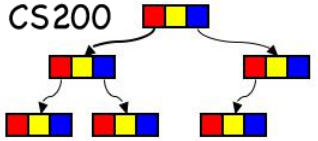
Policies



Be professional. Read the web site on this.

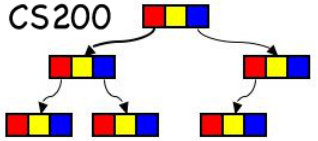
Let's talk about cheating

Cheating



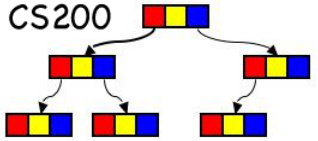
- What is cheating? What is not?
  - **Where would you find a definition?**
- What is gained / lost when cheating?
- What are the consequences?
- When / how does it happen?
  - **How can cheating be avoided?**

Late Policy



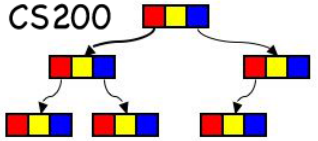
- Programming and Written Assignments
  - By due date/time: full credit
  - Within 48 hours after the deadline: 10% penalty
  - After 48 hours: 0

Distractions in the classroom



- Cell phones
  - Turn off (first choice) or on vibrate
  - If expecting an important call, sit close to the door and step out.
- Laptops & Smart Phones
  - Sit where you will not distract others (back rows)
  - Do try to limit non-class related activities. Psychological evidence shows that we do not multi-task as well as we think we do.

Communication



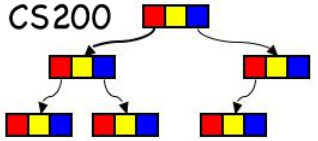
- Check course website often:

  http://www.cs.colostate.edu/~cs200

- Let's go check it out

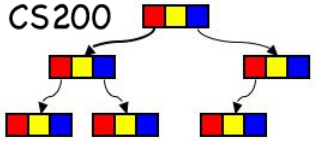
- Canvas will be used minimally
  - to post grades

Course Goals



- CS160: mostly procedural programming, using objects, logic
- CS161: objects, linear data structures, inheritance, induction, counting
- CS200
  - Logical view
    - Program = Algorithms + Data Structures
    - Understand their relationship and use them correctly, efficiently
  - Implementation
    - Program = Objects + Methods
    - Practice design and implementation of object-oriented programs in **Java**
  - Connect theory to programming concepts, complexity

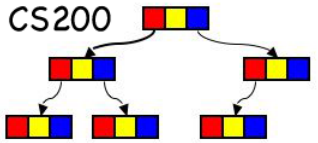
Course Goals



- An understanding of a variety of common data structures
- A practical understanding of where they are applicable
- Understanding the complexity of programs
  - Time complexity: what is the Order of Magnitude time this algorithm takes given an input of size n
  - Space complexity: what is the Order of Magnitude space this algorithm takes given an input of size n

What does order of magnitude mean?

Programming Assignments



- **warm up: stacks**

  1: Implementing recursion using an explicit run time stack

- **expressions** and **assignments**

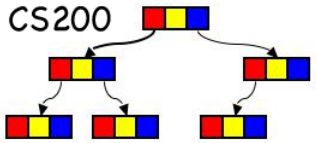
  2: Postfix expressions and evaluation

  3: Infix expressions, parsing, representation, evaluation

  4: Assignments, symbol tables

  5: Analysis: dependences

Design for Change Principle



- Anticipate how systems will evolve and design to accommodate change.
  - Lack of attention to this principle can result in changes that make system unstructured and difficult to understand and maintain.
- How do we do this?
  - Decompose the solution into logically coherent parts, and make these their own classes.
  - In each class, have separate methods for logically coherent operations on the objects created by the class.