Computer Science 210: Data Structures

Introduction

Welcome to Data Structures!

- The class is about designing, analyzing, implementing and using fundamental data structures in solving problems.

- In a nutshell, it's about program development, from problem solving, algorithm and analysis, to implementation, debugging and testing.

- Bottom line: You will do lots of programming.

About the class

- Prerequisites:
  - csci 101 (at Bowdoin or in high-school)
  - In other words
    - beginner knowledge of programming (in Java)
    - enjoy programming and problem solving
    - have a desire to learn more of it

More about the class

- A continuation of 101, at a faster pace, with more complex problems.
- 101
  - You learnt how to use the basic constructs in Java
  - The focus was on learning the tools available when writing a program
    - syntax, conditionals, loops, arrays, etc
- 210
  - You'll learn more tools
  - Most importantly you'll learn to put them together to create a large program
  - And...you'll learn to LEARN

About the class

- Instructor: Laura Toma
    - Office: Searles 219
- TAs:
    - Jack Morrison
    - Jesus Navarro
    - Lizzie Mamantov
- Office hours:
    - Laura: Mon, Tue 3-4:30pm in my office (quick questions any time I am in the office)
    - Jack: TBD (in Searles 224)
    - Jesus: TBD (in Searles 224)
    - Lizzie: TBD (in Searles 224)

About the class

- Website:
  - http://www.bowdoin.edu/~ltoma/teaching/cs210/spring11/
  - class not maintained on Blackboard
- Textbook: none
- Useful books:
  - online: Sedgewick & Wayne, Programming in Java
  - Goodrich and Tamassia Data Structures and Algorithms in Java
  - Lewis and Chase, Java software structures: Designing and using data structures
- Class email:
  - csci210 at bowdoin
  - use it!

Work and grading policy

- Class work:
    - lab assignments (approx. 50%)
    - 2 exams (approx. 50%)
    - in-class and group behavior
- Lab assignments are not meant to be finished during lab time. You (generally) have one week (sometimes longer) to complete them.
    - Handing in: hard copy + email
    - hard copy signed that you have followed class honor code
- Labs: team of <= 2 people
- Late policy: 25% per day
    - Why? it is absolutely essential that you do not fall behind
    - failure to turn in a lab ==> fail the class

      ==> better turn in incomplete lab

Honor code

- Students are expected to follow the Bowdoin Computer Use Policy and the Academic Honor Code.
- You are encouraged to discuss ideas and techniques broadly with other class members, but not specifics of assigned problems except as part of group projects.
- Discussions should be limited to questions that can be asked and answered without using any written medium (e.g. pencil and paper or email).
- This means that at no time should a student read any code written by another student unless they are part of the same group.
- Sharing of code or intermediate designs is expressly prohibited.
- The same rules apply once you have finished the course: sharing your code with other students will be considered a violation of Bowdoin's honor code.
- Violation of this policy is grounds for me to initiate an action that would be filed with the Dean's office and would come before the J Board.
- If you have any questions about this policy, PLEASE do not hesitate to contact me. This will be a zero-tolerance policy.

Course outline

- Week 1: Searching and sorting.
- Week 2: Analysis of algorithms.
- Week 3: Object-oriented (OO) concepts.
- Week 4: Vectors and linked lists.
- Week 5,6: Recursion.
- Week 7: Stacks and queues.
- Week 8: Exam 1
- Week 9,10: Searching and backtracking using recursion, stacks and queues. Breadth- and depth-first search.
- Week 11: Maps and hash tables.
- Week 12: Trees and search trees.
- Week 10: Balanced binary search trees.
- Week 13: Graphs.
- Week 14: Python.
- -------- Exam 2

Pong

Breakout

Sudoku

Boggle

Tetris

Terrains

...

More about the class

- You'll do lots of programming. However, it's NOT about programming
  - but about programming methodology: design + analysis + programming + debugging
- Programming language: Java.
  - makes graphics and web applications easy
  - available on all platforms
  - relatively new language, high level, good to start with, must know
- Most of the class will be independent of Java
  - maybe next semester in Python?
  - at the beginning you'll ask lots of Java questions, and gradually learn to check the Java doc pages, and focus instead on conceptual questions
- Java graphics is NOT the core of the class
  - Java graphics will be used to improve the interface of your programs

Labs

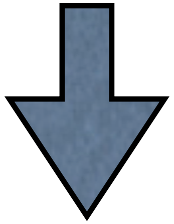
- The labs are not meant to be finished during lab time
  - Labs due one week after they are assigned; they are your homework
- Labs are not all equal
  - in general, progressively harder
- The labs are not always connected to the topic studied in class that week
  - the focus is on solving a problem
- The labs are often harder than they look. You'll spend a lot of time understanding what the task is. It is a good idea to read the lab beforehand, so that you can ask many questions during lab time.
- Labs are challenging and fun. They are the core of the class
  - you will REALLY learn while working out the labs
  - at times the process may seem painful, and occasionally you will find a lab unfair because it will not give you all the information (that you think it should). That is exactly the point.
- However, at the end of the class you'll find that you've learned a lot.

Expectations

- TOGETHER
  - During class time we'll talk about data structures concepts, we'll analyze various options and we'll work out the implementation details for some of these options
  - Often during class-time we'll program together as a group
  - Occasionally there'll be in-class assignments and team work
- YOU
  - You'll start your lab in a timely manner
  - You need to develop your code so that it can be debugged!!
    - think, plan, analyze, develop incrementally, debug, test
  - Structure your code so that you implement one feature at a time, you debug it and test it, and then move on.
  - You'll learn to figure it out
  - You'll learn to think like a computer scientist

More expectations

- Problem: various backgrounds
  - 101 A vs. 101B
  - 101 vs. highschool
  - highschool 1 vs. highschool 2



- Willingness to work in a group
- Patience with material that is not new and when class is slow
  - participate, share
- Ask plenty of questions when something is unclear
- Goal: we want to work as a class

Class Outcomes

- You will learn the fundamental data structures:
    - lists, vectors stacks, queues, priority queues, trees, hash tables and maps
- Design: you will learn to model and come up with a solution to a problem
    - modularity, data abstraction, building blocks
- Analysis: you will learn to analyze the efficiency of your solution
    - you will learn to use efficiency considerations to decide the choice of data structures
- Program development: you will learn the importance of each step in getting a program to work: design, debug, test
    - Practice of programming:
        - Simplicity
        - clarity
        - generality
- You'll learn to figure the language details out on your own

This being said...

- 210 will be occasionally challenging and lots of work
- But, most of the people who take 210
    - like it
    - say it is one of the most fun classes they took
    - continue with Computer Science
- If you like 210, you should think about majoring or minoring in computer science
- You are all here because you liked 101. Welcome!