3013-Algorithms

General Course Info

- **Days:** TTh 12:30 p.m. - 1:50 p.m.
- **Location:** BO 320
- **Semester:** Monday January 10th - Friday April 29rd
- **Holidays:**
  - **Martin Luther King Jr. Day** Monday January 17th
  - **Spring Break :** Monday March 14st - Friday March 19th
- **Last Day for "W":** Friday March 21st
- **Last Day of Class:** Friday April 29th
- **Final Exam:** Thursday May 5th @ 10:30am in BO 320

Resources

Here are some open source books for the course. I hope you guys appreciate the amount of effort it takes to put material together and then put it on the internet for free.

- Discrete Structures for Computer Science: Counting, Recursion, and Probability
  - Thanks To: Michiel Smid
- Open Data Structures
  - Code available HERE
  - Thanks To: Pat Morin
- Algorithms
  - Thanks To: Jeff Erickson
- Wikipedia Collection of Data Structures

Assumed

- This course assumes you know what `array based data structures` and `list based data structures` are.
- For example you should be able to write (from scratch) an `array based stack` and `queue` along with a `list based stack` and `queue`. If you cannot, go study.
- You should have a general understanding of recursive functions.
- You should have a general understanding on graph structures more specifically be able to write and traverse a basic Binary Search Tree (BST).
- Basic OOP skills. Mainly encapsulation and implementation hiding, in other words packaging a data structure with the methods to manipulate that data structure.

Note About This Courses

I will try to go over the list of topics (see below) in the order they are listed, and I have a path that I like to follow when introducing these topics. However, I approach each course with the hopes that student interaction and feedback will steer the course in a direction that may not be what was previously planned. This is stressful for some of my more "organized" students, and I can appreciate their angst. So I always provide a study guide before each exam to ensure everyone is on the same page with the topics I expect you to study. The study guide could have actual test type questions, small programs to implement, or vocabulary and topics.

Algorithm Categories

- Backtracking
- Brute Force
- Dynamic Programming
- Greedy Algorithms
- Recursion

Topics List

As I stated above, we will try to stay on task, but let me give you an example of why things can't always be taught sequentially. There are always multiple ways to implement every data structure. For example there is a data structure called a **priority queue**. It can be implemented in many ways, some better than others. Some of these won't make sense right now, but you will get the idea. Here are a few:

1. Use an **array**, and order the items in the array using the "priority" value.
2. Use a **singly linked list**, and order the nodes using the "priority" value.
3. Use an **array based binary tree**, called a **binary heap** and using the "priority" value to order the heap.
4. Use a **doubly circular linked list**, and follow the algorithm known as a **Fibonacci Heap** to keep the items in order.

So ultimately based on class input and questions that come up in discussion, I may jump around a bit so an explanation will make sense. Remember though ... *I make study guides for each test*!!

- [ ] Array Based vs List Based Structures
- [ ] Array Based Implementations
- [ ] Complexity
  - [ ] Introduction
  - [ ] Will be discussed with each data structure
- [ ] Linked List Types
  - [ ] Singly Linked List
  - [ ] Doubly Linked List
  - [ ] Circular List
- [ ] Stack, Queue, Priority Queue, Deque
- [ ] Array Based Binary Search
- [ ] Binary Tree's
  - [ ] Components
  - [ ] Array Based
- [ ] Binary Heap (Array Based)
- [ ] Fibonacci Heap (**Possibly**)
- [ ] Binary Tree Implementation (List Based)
- [ ] Trie
- [ ] Balanced Tree's
  - [ ] AVL
  - [ ] Red Black (**Possibly**)

- [ ] Hash Tables
- [ ] Graphs
    - [ ] Array Based and List Based Implementations
    - [ ] Basic Graph Algorithms
    - [ ] DFS (Depth-First Search)
    - [ ] BFS (Breadth-First Search)
    - [ ] Minimum Spanning Trees
        - [ ] Prim's Algorithm (Minimum Spanning Tree)
        - [ ] Kruskal's Algorithm (Minimum Spanning Tree)
    - [ ] Shortest Path
        - [ ] Dijkstra's Algorithm (One Way Shortest Path)
        - [ ] A-Star Algorithm (**Possibly**)
- [ ] Sorting:
    - [ ] O(n^2)
        - [ ] Bubble Sort
        - [ ] Selection Sort
        - [ ] Insertion Sort
    - [ ] O(n lg n)
        - [ ] Merge Sort
        - [ ] Quick Sort
    - [ ] Other
        - [ ] Counting Sort
        - [ ] Radix Sort

Resources

- http://opendatastructures.org/ods-cpp/
- https://github.com/ippeb/ACM-ICPC

Grading

Categories		**Grade**	
Exams (3)	40%	A	89-100
Programs (3-5)[1] & Assignments	30%	B	79-88
Final[2]	20%	C	69-78
Github Portfolio	10%	D	59-68
Project Presentations (time permitting)[3]	10%	F	below 59

> **1.** Despite the low overall value of the programming portion of the course, ALL programs must be turned in running to pass the course. They don't have to be necessarily correct, but they must run and they need to at least approach the solution (a "Hello World" program will not work).
>
> **2.** Plane ticket prices, events like weddings, or trips out of the country are not valid excuses for missing the final exam at its scheduled time. I will not make accommodations for anything other than an issue

> vetted by the dean of students.
>
> **3**. The 10% for project presentations will be taken from Programs & Assignments if we have time to do presentations.

Academic Misconduct Policy & Procedures

Cheating, collusion, and plagiarism (the act of using source material of other persons, either published or unpublished, without following the accepted techniques of crediting, or the submission for credit of work not the individual's to whom credit is given . The Department of Computer Science has adopted the following policy related to cheating (academic misconduct). The policy will be applied to all instances of cheating on assignments and exams as determined by the instructor of the course. (See below for link to MSU definitions.)

- 1st instance of cheating in a course: The student will be assigned a non-replaceable grade of zero for the assignment, project or exam. If the resulting grade does not result in a letter grade reduction, the student will receive a one letter grade reduction in course.
- 2nd instance of cheating in a course: The student will receive a grade of F in course & immediately be removed from course.
- All instances of cheating will be reported to the Department Chair and, in the case of graduate students, to the Department Graduate Coordinator.

Note: Letting a student look at your work is collusion and is academic misconduct!

See Also: MSU Student Handbook: Appendix E: Academic Misconduct Policy & Procedures
https://msutexas.edu/student-life/_assets/files/handbook.pdf.