Syllabus: COSC 3100: Data Structures and Algorithms 2

Spring 2020

Instructor: Thomas Schwarz, SJ

Office Hours: MWF 10:10 - 11:00 and by appointment

Text book (required): Thomas Cormen, Charles Leiserson, Ronald Rivest, Clifford Stein: Introduction to Algorithms, MIT, 2009

Prerequisites: **Good knowledge of Calculus I, Calculus II, Discrete Mathematics.** You should have no difficulties determining simple integrals such as $\int_1^n \frac{1}{x} dx$ and relate such an integral with the sum $\sum_{i=1}^{n} \frac{1}{n}$, you should be able to use the rule of L'Hôpital to derive limits, you should be able to use induction and you should be able to use and apply recursive definitions.

Capability of programming in Python. I will expect you to program algorithms in Python and time them. If you do not know how a Python dictionary works or how to import the time module, then you need to bring up your programming skills very quickly. If you do not know Python, I will allow the use of C / C++, but not Java, since timing experiments in Java are useless.

Capability to produce decent word-processed Mathematical documents: I invite you to spend the time to learn Latex. Latex has become the de facto standard in Computer Science and produces crisp, good-looking documents. While LaTeX is more native to the Unix environment, perfectly fine LaTeX versions such as MikTex exist. If you have a Mac, you can use Pages and insert type-set equations using LaTeX syntax. You can also use a third-party tool like LaTeXiT for Mac. While the quality is lower than for pure LaTeX, it is acceptable. In the Office universe (Windows Office, LibreOffice, FreeOffice), you mighty need to invest in third-party equation type setting software if Equation Editor is not satisfactory. The latest versions of office now allow you to insert equations in LaTeX and that should be sufficient. **All homework needs to be handed in in type-set format and printed form. Handwritten homework will be discarded.** You can use the IEEE Word and Latex templates **https://www.ieee.org/conferences/publishing/templates.html** for easy formatting. I do not expect an abstract or a bibliography of course and I do not insist on double-column format. I am aware that type-setting / word-processing takes a lot of time for a beginner.

Access to you-tube: The authors of the book have put several excellent MIT classes online. Not only should you consume these videos on your own initiative, but I will also ask you to watch certain videos in preparation for class.

Contents:

The exact contents will change.

1. Overview
    1. Relation between algorithms and implementations
2. Finite Automata and Regular Expressions
    1. Deterministic finite automata
    2. Non-deterministic finite automata
    3. Non-deterministic finite automata with ϵ-moves
    4. Regular Expression
    5. Equivalence of DFA, NFA, and Regular Expressions
    6. Mealy and Moody Machines
3. Computational Model and Run-time Evaluation
    1. RAM model
    2. Growth of functions and Landau Notation
4. Correctness of Algorithms
    1. GCD
    2. Euclidean Algorithms
    3. Loop Invariants
5. Recursion and Divide and Conquer Algorithms
    1. Integer multiplication
    2. Binary search
    3. Strassen multiplication
6. Solving Recurrence with the Master Theorem
7. Modern Data Structures
    1. Linear Hashing
    2. B-Tree
8. Dynamic Programming
    1. Dynamic programming principles
    2. Knapsack problems
    3. Memoization
9. Greedy Algorithms
10. Analysis of Graph Algorithms
    1. Elementary graph algorithms
        1. Graph representations
        2. Breadth first search
        3. Depth first search
        4. Topological Search
        5. Strongly connected components
    2. Shortest Path Problems
        1. Bellman Ford algorithms
        2. Single source shortest paths in directed acyclic graphs
        3. Dijkstra’s algorithm and its correctness
    3. Flow problems
11. Limits of Computability — Impossibility Results
    1. Turing Machines
        1. Definition
        2. Turing machines with different types of tapes
    2. Church Turing Thesis
    3. Halting Problem
    4. Philosophical Implications
12. Complexity Classes
    1. Classes P, NP
    2. Existence of Strong Cryptography
    3. P != NP hypothesis

Grading:

Review Section	5%
Weekly Programming Assignments	20%
Weekly Homework (via D2L)	20%
Daily quizzes (in-class / via D2L if presence not possible)	10%
1 Midterm	20%
1 Final	25%

Accommodations, absences, plagiarism cases, etc. will be dealt with strictly according to Marquette University's policies and regulations. In particular, if you miss six (6) classes by being late and not handing in quizzes, you will be dropped.

If a student or the instructor are prevented from attending classes, I will broadcast class via zoom. Previous notification is required.