CS-435-101 Syllabus

Catalog Description

Advanced topics in data structures and algorithms, involving sequences, sets, and graphs such as searching, sorting, order statistics, balanced search tree operations, hash tables, graph traversals, graph connectivity and path problems. Algebraic and numeric algorithms. Performance measures, analysis techniques, and complexity of such algorithms.

Student-friendly Description

Algorithms and data structures affect the lives of billions of people daily. Think about it this way: if you have ever used a cell phone then you have been a user of several algorithms that route calls, decode information, remove noise, sort data, or move graphics around fast. This course introduces undergraduate students to notions and techniques that are used as `building blocks' in the design of algorithms.

We will study runtime analysis, correctness proofs and specific algorithms, using as motivation problems that appear in programming interviews. We will also discuss problems that seem to be impossible to be solved efficiently, posing one of the most challenging problems of our time.

Contact Information

Instructor: Ioannis Koutis
Email: ikoutis+cs435@njit.edu
Office: GITC 4314
Course Time and Location: Tuesday 6:00-9:00pm (CKB219)
Office Hours: Tuesday 4:30-5:30, Monday: 2:30-3:30 and by appointment

Course Assistant: TBA
Recitation and Location: Monday 1:00-1:55pm (CKB206)

Prerequisites and Textbook

Prerequisites: CS 241, CS 288

Textbook: T.C.Cormen, C.E.Leiserson, R.L.Rivest, and C. Stein. "Introduction to Algorithms", 3rd edition, MIT Press, ISBN-10 : 0262033844. We abbreviate it in class as *CLRS*.

Alternative Reading: S. Dasgupta, C. Papadimitriou, U. Vazirani. "Algorithms", McGraw-Hill Education, ISBN-10: 0073523402. We abbreviate it in class as *DPV*.

Additional material such as lecture slides and notes will be posted on Moodle

Coursework and Evaluation

Evaluation will be based on **tasks,** including six homework assignments, one mini-project and two exams (midterm and final). Each **task** will be graded on a 1-100 scale. This raw grade will be converted to 0-4 scale grade, corresponding to letter grades as follows: [4=A, 3.5=B+, 3=B, 2.5=C+, 2=C, 1=D, 0=F]. The conversion will be based on grouping the raw grades into clusters using the k-means method, and then assigning a letter grade to each cluster. The letter grade assignment will be in accordance to the undergraduate grade legend (https://www.njit.edu/registrar/policies/grading.php). This implies that there is no limit on the percentage of students who can get any given letter grade (everyone can get an A). The final numerical grade will be a weighted average of the scale-4 grades received in the various tasks. The weighting will be as follows:

Homework Assignments [30%]. There will be 6 homework assignments. Only the best 5 assignment grades will be used in the weighted average, each contributing 6%. This means that you have the flexibility to skip one assignment. Try to make wise use of the option!

Mini-Project [20%]. There will be one nearly semester-long programming project. We will be slowly building the required components during the semester following certain milestones. The project will be announced in the second week of classes. Evaluation will include a short presentation to the instructor.

Midterm Exam [20%]. Open-textbook only. You can bring your own copy of the textbook, but you are not allowed to borrow one during the exam.

Final Exam [30%]. Cumulative. Open-textbook only. You can bring your own copy of the textbook, but you are not allowed to borrow one during the exam.

The final numerical grade (g) will be converted to the **final letter grade** as follows:
[**A:** g>3.5, **B+:** 3<g<3.5 , **B:** 2.5<g<3, **C+**: 2<g<2.5, **D:** 1<g<2, **F:** g<1]

Lateness Policy. Each student starts the semester with a **total budget** of 72 hours of combined delay. If the budget is exceeded, 1 raw point will be subtracted for each hour of delay.

Tentative Course Schedule

Week 1:	09/05	Hw1 is out
Week 2:	09/12	Mini-Project is out
Week 3:	09/19	Hw1 is due, Hw2 is out
Week 4:	09/25	Hw1 solutions posted, discussed in recitation
Week 5:	10/03	Hw2 is due, Hw3 is out
Week 6:	10/09	Hw2 solutions posted, discussed in recitation
Week 7:	10/17	Hw3 is due
Week 8:	10/23	Hw3 solutions posted, discussed in recitation
	10/24	**Midterm Exam** (first half exam, second half solutions)
Week 9:	10/31	Hw4 is out
Week 10:		
Week 11:	11/14	Hw4 is due, Hw5 is out
Week 12:	11/20	Hw4 solutions posted, discussed in recitation
Week 13:	11/28	Hw5 is due, Hw6 is out
Week 14:	12/04	Hw5 solutions posted, discussed in recitation
Week 14:	12/05	Mini-project due
Week 15:	12/08	Hw6 due
Week 15:	12/11	Hw6 solutions posted, discussed in recitation
Week 16:		**Final Exam** (date and time set by registrar)

Any modifications will be done in consultation with the attending students and will be announced on Moodle.

Tentative Topics and Time Management

Weeks 1-2: Basics
Basic Instructions, Asymptotic Notation, Running time Analysis, Brute-force solutions

Weeks 3-6: Algorithmic Techniques
Greedy Algorithms, Divide and Conquer, Recursion, Dynamic Programming, Randomness

Weeks 7-8: Simple Data Structures
Sorting, Heaps, Hashing

Weeks 10-12: Advanced Data Structures
Binary Trees, Red-Black Trees, AVL trees

Weeks 13-14:
Algorithmic Surprises, Hard Problems, Coping with Hardness

Week 15: Problem Solving
Problems on Graphs, Problems on Strings and Lists

Course Policies

Email
Use of your NJIT email is strongly encouraged to avoid delivery problems.

Moodle Announcements
All course announcements will be posted on Moodle. So, check it regularly, or even better **subscribe** to the corresponding forums to receive e-mail notifications.

Moodle Discussion Forums
We will use Moodle to have informal and friendly conversations about topics related to the course, including assignments, problems, ideas, etc. You are encouraged to participate. Please be absolutely assured that any question or idea is welcome; even if you are afraid it may be silly, it can lead to interesting and beneficial discussions.

Mobile Devices
Turn-off and place in a bag any cell phones, or other mobile devices, including laptops (unless the instructor explicitly permits you otherwise).

Assignment Delivery
Be careful to read and follow any formatting and delivery instructions that come with each assignment.

Grading
Written work will be graded for conciseness and correctness. Use formal arguments. Be brief and to the point and write clearly. Material covered in class and appearing in the relevant notes and chapters of the designated textbook can be used without proof. **Do not use pencils** in the exams.

Grade Corrections
Check the marks in course work and report errors promptly. Please try and resolve any issue within one week of the mark notification.

Absenteeism

If you miss a class, it's up to you to make up for lost time. Missing both exams leads to an automatic F in the course. If you miss one exam you **must contact** the Dean of Students (DOS) within 2 working days from the day the reason for the absence is lifted with all necessary documentation. If DOS approves, your missing exam grade will be set equal to the non-missing exam grade.

Exam Conflicts

CS435 is a high-numbered required course. In case of multiple exams on a same day, this exam has priority even if it is the last exam of the day.

Incomplete

A grade of I (incomplete) is given in rare cases where work cannot be completed during the semester due to documented long-term illness or unexpected absence for other serious reasons. A student needs to be in good standing (i.e. passing the course before the absence) and receives a provisional I if there is no time to make up for the documented lost time; a letter (or email) with a timeline of what is needed to be done will be sent to the student. Note that for most cases an I would be resolved within few days, not months and not the following semester! Not showing up in the final will probably get you an F rather than an I.

Collaboration and External Resources for Assignments

Some homework assignment problems will be challenging. You are advised to first try and solve all the problems **on your own**. For problems that persist you are welcome to talk to the course assistant or the instructor. You are also allowed to collaborate with your classmates and search for solutions online. But you should use such solutions only if you understand them completely (admitting that you don't understand something is way better than copying things you don't understand). Also make sure to give the appropriate credit and citation.

Collaboration in the mini-Project and Exams

Such collaboration is strictly prohibited. Any submitted code (even few lines) obtained through the Internet or otherwise, or is product of someone else's work or is common with another student submission, in the same or other section/course, risks severe punishment, as outlined by the University; all parties of such interaction receive automatically 0 and grade is lowered by one or two levels.

The NJIT Academic Integrity (Honor) Code will be upheld; violations will be reported to the Dean of Students (DOS).

Course Objectives

A1. Learn how and be able to asymptotically compare functions and be able to solve recurrences using the master, the iteration/recursion tree, and the substitution methods.
A2. Learn how and be able to describe the asymptotic performance of algorithms and data structure operations.
A3. Learn how and be able to understand fundamental algorithms and data structures and be able to trace their operations for problems such as sorting, searching, selection, operations on numbers, polynomials and matrices, and graphs.
B1. Learn how and be able to understand the input and output specifications of a problem, the relationship between input and output and identify, define, and describe the algorithmic requirements that formulate a solution for those problems.
C1. Learn and be able to identify the performance characteristics of algorithms and data structures for problems such as sorting, searching, selection, operations on numbers, polynomials and matrices, and graphs; be able to select among multiple available solutions to meet desired needs.
C2. Learn how fundamental algorithm design techniques work and be able to understand how to use them to design, implement and evaluate a variety of algorithmic problems.
F1. Learn how and be able to describe in writing algorithm operations and reason about their behavior and performance succinctly.
I1. Learn how and be able to effectively apply the knowledge gained in the course in dealing and interacting with software libraries that offer same or alternative implementations of algorithms and data structures.
J1. Learn how and be able to model, design, and construct algorithms and data structure operations and analyze and derive their asymptotic performance and provide tradeoff solutions (eg. space vs time).
J2. Learn how and be able to understand the operations of fundamental algorithms and data structures, their characteristics, and be able to choose among a variety of similar ones based on problem/program specification and requirements in building more complex algorithms and data structures, and deciding trade-offs between space and time requirements.
K1. Learn how and be able to compose more complex algorithms and data structures using as building blocks the fundamental algorithms and data structures introduced in class.
K2. Learn how and be able to compose more complex algorithms using the algorithmic design techniques introduced in class. 2