IE 405: Computing for ISE

Fall 2025 TR 3:30 - 4:50 PM CT

114 Transportation Building

(Syllabus)

Instructor:	Jugal Garg (jugal@illinois.edu), 216B Transportation Building
TA:	TBD
Course website:	`canvas.illinois.edu`
Slack link:	TBD
Office hours (Zoom):	TBD

Course Communication

All announcements, assignments, lecture slides, and other materials will be done through the course website on CANVAS.

Course Description

This course will introduce students to algorithm design, computer programming in Python, and database SQL queries. It will provide the fundamental methods, concepts and principles of these topics to give students enough breadth to use these techniques in their jobs and to prepare them to pursue advanced topics in these areas. There will be weekly programming assignments to implement algorithms and SQL covered in the class.

Course Overview

This course aims to cover the breadth of three different topics, namely computer programming in Python, algorithm design and SQL queries. In particular it will cover fundamental techniques such as divide and conquer, greedy algorithms, basic graph algorithms, and dynamic programming. Also, it will cover how to analyze the cost of computing and the limits of what we can and cannot compute in a reasonable amount of time. Furthermore, the basic principles of computer programming in Python, database design and SQL queries will be covered.

Prerequisites

- CS 101 or equivalent, IE 310 or equivalent

References

1. *Algorithm Design* by Jon Kleinberg and Éva Tardos. Pearson Education (2013)
2. *Introduction to Algorithms* by Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, Clifford Stein. 3rd Edition, MIT Press (2009)

3. *Database System Concepts* by Abraham Silberschatz, Henry Korth, S. Sudarshan. 6th edition, McGraw-Hill (2010)
4. *Database Management Systems* by Raghu Ramakrishnan, Johannes Gehrke. 3rd Edition, McGraw-Hill (2002)

Course Objectives

Students completing this course will be able to:

1. write computer programs in Python
2. design algorithms using greedy, dynamic programming and divide-and-conquer paradigm, and prove its correctness.
3. analyze an algorithm and how to count number of steps it takes
4. understand basic data structures, e.g., array, list, queue, stack, etc.
5. implement algorithms, designed in the *theory* class, in Python, e.g., Gale-Shapley algorithm for stable matching, Dijkstra's algorithm for shortest path, Kruskal's and Prim's algorithm for minimum spanning tree, check whether a graph is bipartite using Breadth-First-Search (BFS) algorithm, dynamic programming algorithm for knapsack problem, Merge sort, etc.
6. understand complexity classes `P/NP` and computational intractability
7. understand the role of a database management system in an organization
8. understand the structure and operation of the relational database model
9. construct simple and intermediate level database queries using Structured Query Language (SQL)

Course Project

The goal is to develop a fully-fledged Python program in a collaborative, team environment, written entirely from the scratch with proper documentation and instructions on how to use the program. Another option is to choose a topic in database design and SQL. More details will be provided in the middle of the course.

Course Topics

- Algorithm Design
  - Stable Matching (Gale-Shapley Algorithm), Basics of Algorithm Analysis (Computational Tractability, Asymptotic Order of Growth, Big O Notation), Graphs (Basic Definitions and Applications, Graph Connectivity, Breadth-First-Search (BFS), Testing Bipartiteness: an application of BFS), Greedy Algorithms (Interval Scheduling, Scheduling to Minimize Lateness, Shortest Paths in a Graph, Minimum Spanning Tree Problem), Divide and Conquer (Merge sort, Integer Multiplication), Dynamic Programming (Rod Cutting Problem, Weighted Interval Scheduling, Knapsack), `P/NP` and Computational Intractability (Polynomial Time Reductions, Efficient Certification and the Definition of `NP`, `NP`-complete Problems).
- Computer Programming in Python
- Database Design and SQL Queries

  - Relational Model, Entity-Relationship Model, Database Schema, Keys, Schema Diagrams, Basic SQL (SQL Data Definitions, Set Operations, NULL Values, Aggregate Functions, Nested Subqueries, Modification of a Database), Intermediate SQL (Join Expressions, Integrity Constraints, SQL Data Types and Schemas).

Required Work and Grading Policy

- Undergraduate Credit
  1. Homework – Weekly assignments (40%)
  2. 3 exams (20% each)
- Graduate Credit
  1. Homework – Weekly assignments (35%)
  2. 3 exams (15% each)
  3. Project (20%)

Contact Hours

1 Theory and 1 Lab class (80 minutes each) per week.

Academic Integrity

We will follow Student Code Part 4 1-401 through 1-406 (`https://studentcode.illinois.edu/article1/part4/1-401/`).