CSE 431/531: Algorithm Analysis and Design (Spring 2024)

Introduction and Syllabus

Lecturer: Kelin Luo

Department of Computer Science and Engineering
University at Buffalo

Outline

1. Syllabus

2. Introduction
   - What is an Algorithm?
   - Example: Insertion Sort
   - Analysis of Insertion Sort

3. Asymptotic Notations

4. Common Running times

CSE 431/531: Algorithm Analysis and Design

- Time & Location : Tue-Thu, 02:00pm - 03:20pm, Cooke 121
- Instructor: Kelin Luo, kelinluo@buffalo.edu
- TAs:
    - Ibrahim Bahadir Altun, ialtun@buffalo.edu
    - Yuxin Liu, yuxinliu@buffalo.edu
    - Chen Xu, chenxu@buffalo.edu
    - Wen Zhang: wzhang59@buffalo.edu
- Office hour

CSE 431/531: Algorithm Analysis and Design

- Course Webpage (contains schedule, policies, and slides): https://cse.buffalo.edu/~kelinluo/teaching/cse431A:531A-spring24/index.html
- Please sign up course on Piazza via link https://piazza.com/buffalo/spring2024/cse431531a on course webpage
  - homeworks, solutions, announcements, asking/answering questions

Acknowledgement: The course design and information primarily draw inspiration from the Algorithm Analysis and Design course by Prof. Shi Li in Fall 2022 and Kelin Luo in Fall 2023.

CSE 431/531: Algorithm Analysis and Design

Introduces basic elements of the design and analysis of algorithms.

- Topics include asymptotic notations and analysis, algorithm frameworks, NP-completeness, and approximation algorithms.

CSE 431/531: Algorithm Analysis and Design

Introduces basic elements of the design and analysis of algorithms.

- Topics include asymptotic notations and analysis, algorithm frameworks, NP-completeness, and approximation algorithms.
- For each topic, beside in-depth coverage, we discuss one or more representative problems and algorithms.

CSE 431/531: Algorithm Analysis and Design

Introduces basic elements of the design and analysis of algorithms.

- Topics include asymptotic notations and analysis, algorithm frameworks, NP-completeness, and approximation algorithms.
- For each topic, beside in-depth coverage, we discuss one or more representative problems and algorithms.
- Learn discrete mathematics problem solving skills essential for computer scientists and engineers.

You should already have/know:

You should already have/know:

- Mathematical Background
  - basic reasoning skills, inductive proofs

You should already have/know:

- Mathematical Background
  - basic reasoning skills, inductive proofs
- Basic data Structures
  - linked lists, arrays
  - stacks, queues

You should already have/know:

- Mathematical Background
  - basic reasoning skills, inductive proofs
- Basic data Structures
  - linked lists, arrays
  - stacks, queues
- Some Programming Experience
  - e.g. Python, C, C++ or Java

You Will Learn

- Classic algorithms for classic problems
  - Sorting, shortest paths, minimum spanning tree, $\cdots$

You Will Learn

- Classic algorithms for classic problems
  - Sorting, shortest paths, minimum spanning tree, $\cdots$
- How to analyze algorithms
  - Correctness
  - Running time (efficiency)

You Will Learn

- Classic algorithms for classic problems
  - Sorting, shortest paths, minimum spanning tree, $\cdots$
- How to analyze algorithms
  - Correctness
  - Running time (efficiency)
- Meta techniques to design algorithms
  - Greedy algorithms
  - Divide and conquer
  - Dynamic programming
  - $\cdots$

You Will Learn

- Classic algorithms for classic problems
  - Sorting, shortest paths, minimum spanning tree, $\cdots$
- How to analyze algorithms
  - Correctness
  - Running time (efficiency)
- Meta techniques to design algorithms
  - Greedy algorithms
  - Divide and conquer
  - Dynamic programming
  - $\cdots$
- NP-completeness

Tentative Schedule

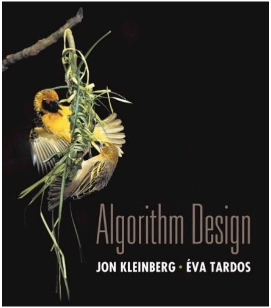
- 75 Minutes/Lecture × 28 Lectures

Introduction	3 lectures
Graph Basics	2 lectures
Greedy Algorithms	4 lectures
Divide and Conquer	5 lectures
Dynamic Programming	5 lectures
Graph Algorithms	4 lectures
NP-Completeness	4 lectures
Final Review	1 lectures

Textbook

Textbook (Highly Recommended):

- Algorithm Design, 1st Edition, by *Jon Kleinberg* and *Eva Tardos*



Other Reference Books

- Introduction to Algorithms, Third Edition, *Thomas Cormen, Charles Leiserson, Rondald Rivest, Clifford Stein*

Reading Before and After Classes

- Highly recommended: read the correspondent sections from the textbook (or reference book or previous slides) before classes
  - Sections for each lecture can be found on the 2023 Fall course webpage.

Reading Before and After Classes

- Highly recommended: read the correspondent sections from the textbook (or reference book or previous slides) before classes
  - Sections for each lecture can be found on the 2023 Fall course webpage.
- Slides are posted on course webpage after the classes. After the lecture, you can review the updated slides on the course webpage (and Piazza) and access the course recordings in Brightspace.

Reading Before and After Classes

- Highly recommended: read the correspondent sections from the textbook (or reference book or previous slides) before classes
  - Sections for each lecture can be found on the 2023 Fall course webpage.
- Slides are posted on course webpage after the classes. After the lecture, you can review the updated slides on the course webpage (and Piazza) and access the course recordings in Brightspace.
- In last lecture of a major topic (Greedy Algorithms, Divide and Conquer, Dynamic Programming, Graph Algorithms), we will discuss in-class quiz (+ presentation) problems.

Grading

- 10% for participation
  - In-class discussions or Brightspace quizzes will be given randomly.
    (We choose the best 10 scores out of 12-15 quizzes.)
- 40% for theory homeworks
  - 8 points × 5 theory homeworks
    (We choose the best 5 scores out of 6 homeworks.)
    (typed PDF submissions, e.g. Microsoft Word, latex.)
- 20% for programming projects
  - 10 points × 2 programming assignments
    (Programming: Python3 only)
- 30% for final exam (closed-book, closed-note)
  (Final exam: May 09, Thursday, 15:30-18:30)

Learning Outcomes and Method of Assessment

- Understand and apply asymptotic notations and analysis
  Quiz, Homeworks, Final Exam

Learning Outcomes and Method of Assessment

- Understand and apply asymptotic notations and analysis
  Quiz, Homeworks, Final Exam
- Understand and apply algorithm analysis and design techniques
  Quiz, Homeworks, Projects, Final Exam

Learning Outcomes and Method of Assessment

- Understand and apply asymptotic notations and analysis
  Quiz, Homeworks, Final Exam
- Understand and apply algorithm analysis and design techniques
  Quiz, Homeworks, Projects, Final Exam
- Solve algorithmic problems arising in applications
  Quiz, Homeworks, Projects, Final Exam

Learning Outcomes and Method of Assessment

- Understand and apply asymptotic notations and analysis
  Quiz, Homeworks, Final Exam
- Understand and apply algorithm analysis and design techniques
  Quiz, Homeworks, Projects, Final Exam
- Solve algorithmic problems arising in applications
  Quiz, Homeworks, Projects, Final Exam
- Understand NP-completeness and approximation algorithms
  Quiz, Homeworks, Final Exam

Learning Outcomes and Method of Assessment

- Understand and apply asymptotic notations and analysis
  Quiz, Homeworks, Final Exam
- Understand and apply algorithm analysis and design techniques
  Quiz, Homeworks, Projects, Final Exam
- Solve algorithmic problems arising in applications
  Quiz, Homeworks, Projects, Final Exam
- Understand NP-completeness and approximation algorithms
  Quiz, Homeworks, Final Exam
- Demonstrate the hardness of simple NP-complete problems
  Homeworks, Final Exam

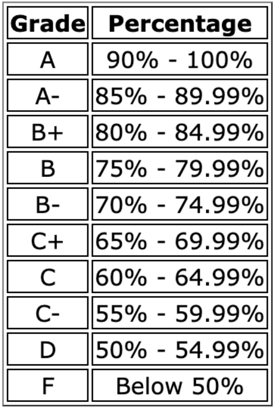
Re-grading

Question about the grading of any piece of work:

- First consult with the teaching assistant who graded your work on Piazza.
- If you cannot resolve your questions with the teaching assistant, you should consult with the instructor.
- Any questions about the grading of a piece of work must be raised within one week of the date that the work was returned by the teaching assistant or the instructor.

Grading policy

- The following outlines the grade breakdown that will be utilized for assigning grades in the course.



Grade	Percentage
A	90% - 100%
A-	85% - 89.99%
B+	80% - 84.99%
B	75% - 79.99%
B-	70% - 74.99%
C+	65% - 69.99%
C	60% - 64.99%
C-	55% - 59.99%
D	50% - 54.99%
F	Below 50%

- Note that these ranges may be subject to adjustment at the end of the semester to address any inconsistencies or hardships that may arise.

For Homeworks, You Are Allowed to

- Use course materials (textbook, reference books, lecture notes, etc)
- Post questions on Piazza
- Ask me or TAs for hints
- Collaborate with classmates
  - Think about each problem for enough time before discussions
  - Must write down solutions on your own, in your own words
  - Write down names of students you collaborated with

For Homeworks, You Are Not Allowed to

- Use external resources
  - Can't Google or ask questions online for solutions
  - Can't read posted solutions from other algorithm course webpages
- Copy solutions from other students
- Use of Artificial Intelligence Technologies like OpenAI's ChatGPT, Google Bard, and AI models within search interfaces like Google or Bing, etc.

For Homeworks, You Are Not Allowed to

- Use external resources
  - Can't Google or ask questions online for solutions
  - Can't read posted solutions from other algorithm course webpages
- Copy solutions from other students
- Use of Artificial Intelligence Technologies like OpenAI's ChatGPT, Google Bard, and AI models within search interfaces like Google or Bing, etc.

If cheating is found, you will get an "F" for the course. The case will be reported to the department.

For Programming Projects

- Use Python version $\geq$3.4
- Need to implement the algorithms by yourself
- Can not copy codes from others or the Internet
- We use turnitin (`https://www.turnitin.com/`) to detect similarity of programs, review the codes

For Programming Projects

- Use Python version $\geq$3.4
- Need to implement the algorithms by yourself
- Can not copy codes from others or the Internet
- We use turnitin (`https://www.turnitin.com/`) to detect similarity of programs, review the codes

If cheating is found, you will get an “F” for the course. The case will be reported to the department.

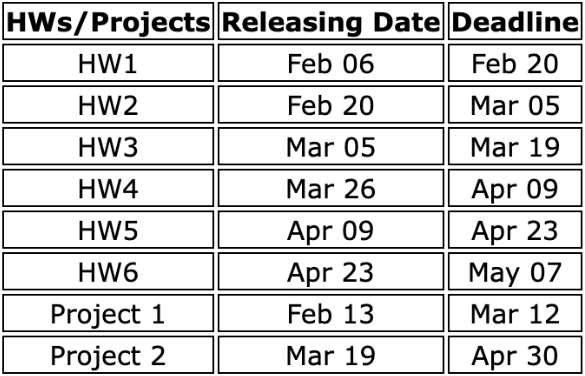
Academic Integrity (AI) Policy for the Course

- minor violation:
  - 0 score for the involved homework/prog. assignment, and
  - 1-letter grade down
- 2 minor violations = 1 major violation
  - failure for the course
  - case will be reported to the department and university
  - further sanctions may include academic dishonesty mark on transcript or expulsion from university

- I. Notify the Student of the Concern
- II. Consult with the Student
- III. Decide the Sanction (Stop or AI violation)
- IV. Send the Decision Letter
- V. Assign the Grade (F with AD)

Late Policy

- No late submissions will be accepted.
- 11:59PM EST. Please submit it before the deadline.



HWs/Projects	Releasing Date	Deadline
HW1	Feb 06	Feb 20
HW2	Feb 20	Mar 05
HW3	Mar 05	Mar 19
HW4	Mar 26	Apr 09
HW5	Apr 09	Apr 23
HW6	Apr 23	May 07
Project 1	Feb 13	Mar 12
Project 2	Mar 19	Apr 30

Course Sign up and Final exam date

- Last Day to Drop/Add a Course: Jan 31
- Resign Date: April 16

- Final exam: May 09, Thursday, 15:30-18:30

Course Sign up and Final exam date

- Last Day to Drop/Add a Course: Jan 31
- Resign Date: April 16

- Final exam: May 09, Thursday, 15:30-18:30
- Final exam conflict

Course Sign up and Final exam date

- Last Day to Drop/Add a Course: Jan 31
- Resign Date: April 16

- Final exam: May 09, Thursday, 15:30-18:30
- Final exam conflict
  - Three or more final exams scheduled on the same day.

Course Sign up and Final exam date

- Last Day to Drop/Add a Course: Jan 31
- Resign Date: April 16

- Final exam: May 09, Thursday, 15:30-18:30
- Final exam conflict
  - Three or more final exams scheduled on the same day.
  - Two final exams occurring at the same time.

Course Sign up and Final exam date

- Last Day to Drop/Add a Course: Jan 31
- Resign Date: April 16

- Final exam: May 09, Thursday, 15:30-18:30
- Final exam conflict
  - Three or more final exams scheduled on the same day.
  - Two final exams occurring at the same time.
  - When a student's final exam occurs contemporaneously with his or her commencement ceremony for spring or summer conferral.

General Resources

Here are some of the University's available free resources:

- If you need help with writing, check UB Center for Excellence in Writing the Writing Support Services
- If you have issues with your device, the UB University Libraries provides access to computers, as well as equipment loans, see the Equipment Loans
- Your well-being is highly important, if you have any concerns, please check the Counseling Service

Accessibility Resources

- If you have any disability which requires reasonable accommodations to enable you to participate in this course, please contact the Office of Accessibility Resources in 60 Capen Hall, 716-645-2608 and also the instructor of this course during the first week of class.
- The office will provide you with information and review appropriate arrangements for reasonable accommodations, which can be found on the web at the Accessibility Resources.

Piazza Post Rule

You can post all questions related to lectures, quiz, and assignments on Piazza. Instructor posts announcement and course materials on piazza.

- For general questions about the course schedule, lectures, assignments, etc., please make your post visible to **everyone**.
- For personal inquiries regarding re-grades of homework or projects, please post and include both **me and the corresponding TA** in your post.
- For other personal queries related to this course, please post and include both **me and all the TAs** in your post.

Questions

Questions?

Questions

Questions?

Further questions, please post on Piazza!

Outline

1. Syllabus

2. Introduction
   - What is an Algorithm?
   - Example: Insertion Sort
   - Analysis of Insertion Sort

3. Asymptotic Notations

4. Common Running times

Outline

1 Syllabus

2 Introduction

- What is an Algorithm?
- Example: Insertion Sort
- Analysis of Insertion Sort

3 Asymptotic Notations

4 Common Running times

What is an Algorithm?

- Donald Knuth: An algorithm is a finite, definite effective procedure, with some input and some output.

What is an Algorithm?

- Donald Knuth: An algorithm is a finite, definite effective procedure, with some input and some output.
- Computational problem: specifies the input/output relationship.
- An algorithm solves a computational problem if it produces the correct output for any given input.

What is an Algorithm?

- Computational problem

1. What is the shortest route from classroom Nsc 215 to classroom Cooke 121?

What is an Algorithm?

- Computational problem

1. What is the shortest route from classroom Nsc 215 to classroom Cooke 121?
2. Which restaurant has the highest rating?

What is an Algorithm?

- Computational problem

1. What is the shortest route from classroom Nsc 215 to classroom Cooke 121?
2. Which restaurant has the highest rating?
3. True or False? 2^{10} is greater than 5^2

What is an Algorithm?

- Computational problem

1. What is the shortest route from classroom Nsc 215 to classroom Cooke 121?
2. Which restaurant has the highest rating?
3. True or False? 2^{10} is greater than 5^2

- Computational problem: specifies the input/output relationship.

What is an Algorithm?

- Computational problem

1. What is the shortest route from classroom Nsc 215 to classroom Cooke 121?
2. Which restaurant has the highest rating?
3. True or False? 2^{10} is greater than 5^2

- Computational problem: specifies the input/output relationship.
- An algorithm solves a computational problem if it produces the correct output for any given input.

Examples

Greatest Common Divisor

Input: two integers $a, b > 0$

Output: the greatest common divisor of a and b

Examples

Greatest Common Divisor

Input: two integers $a, b > 0$

Output: the greatest common divisor of a and b

Example:

- Input: 210, 270
- Output: 30

Examples

Greatest Common Divisor

Input: two integers $a, b > 0$

Output: the greatest common divisor of a and b

Example:

- Input: 210, 270
- Output: 30

- Algorithm: Euclidean algorithm

Examples

Greatest Common Divisor

Input: two integers $a, b > 0$

Output: the greatest common divisor of a and b

Example:

- Input: 210, 270
- Output: 30

- Algorithm: Euclidean algorithm
- $\gcd(270, 210) = \gcd(210, 270 \bmod 210) = \gcd(210, 60)$

Examples

Greatest Common Divisor

Input: two integers $a, b > 0$

Output: the greatest common divisor of a and b

Example:

- Input: 210, 270
- Output: 30

- Algorithm: Euclidean algorithm
- $\gcd(270, 210) = \gcd(210, 270 \bmod 210) = \gcd(210, 60)$
- $(270, 210) \to (210, 60) \to (60, 30) \to (30, 0)$

Examples

Sorting

Input: sequence of n numbers $(a_1, a_2, \cdots, a_n)$

Output: a permutation $(a'_1, a'_2, \cdots, a'_n)$ of the input sequence such that $a'_1 \le a'_2 \le \cdots \le a'_n$

Examples

Sorting

Input: sequence of n numbers $(a_1, a_2, \cdots, a_n)$

Output: a permutation $(a'_1, a'_2, \cdots, a'_n)$ of the input sequence such that $a'_1 \leq a'_2 \leq \cdots \leq a'_n$

Example:

- Input: $53, 12, 35, 21, 59, 15$
- Output: $12, 15, 21, 35, 53, 59$

Examples

Sorting

Input: sequence of n numbers $(a_1, a_2, \cdots, a_n)$

Output: a permutation $(a'_1, a'_2, \cdots, a'_n)$ of the input sequence such that $a'_1 \leq a'_2 \leq \cdots \leq a'_n$

Example:

- Input: $53, 12, 35, 21, 59, 15$
- Output: $12, 15, 21, 35, 53, 59$

- Algorithms: insertion sort, merge sort, quicksort, ...