CSE 317: Design and Analysis of Algorithms (Fall 2025)

School of Computer Science and Mathematics, IBA-Karachi

1 Logistics

Instructor	**Shahid Hussain (SH)**
	Email: shahidhussain@iba.edu.pk
	Office: 207, Tabba, IBA-Karachi, Main campus
	Phone: 3044
	Office hours: TBA
Teaching Assistants	**Danial Haseeb (DH), Zuha Aqib (ZA)**
	Emails: –, z.aqib.26106@khi.iba.edu.pk
	Office hours: DH: TBA, ZA: TBA
Schedule	**Lectures, SH:** Mondays and Wednesday: 10:30 – 11:45am
	Tutorials, DH: Tuesdays and Thursdays: 8:30am – 9:45am
Miscellaneous	**LMS:** https://lms.iba.edu.pk/x/6cszby
	Discord: https://discord.gg/dxcgxVkf

2 Course Objectives

This is a fundamental course on Algorithms in Computer Science. The main goal of the course is to develop tools and techniques that aid in designing correct and efficient algorithms for computational problems and analyzing their correctness and running time. We will study powerful design techniques, such as dynamic programming and randomization, and useful analytical tools, such as recurrence relations and average case analysis. The algorithms under study are either quite useful, instructive, beautiful or any combination thereof. After taking this course, the student will be conversant in the language of algorithms, which is a chief component of technical interviews and software related jobs.

3 Program Learning Outcomes/Graduate Attributes

Graduate attributes (program learning outcomes - PLO's) taken from https://www.seoulaccord.org/document.php?id=79.

PLO-1. Academic Education

[Educational depth and breadth]

Completion of an accredited program of study designed to prepare graduates as computing professionals

PLO-2. Knowledge for Solving Computing Problems

[Breadth and depth of education and type of knowledge, both theoretical and practical]

Apply knowledge of computing fundamentals, knowledge of a computing specialization, and mathematics, science, and domain knowledge appropriate for the computing specialization to the abstraction and conceptualization of computing models from defined problems and requirements

PLO-3. Problem Analysis

[Complexity of analysis]

Identify, formulate, research literature, and solve complex computing problems reaching substantiated conclusions using fundamental principles of mathematics, computing sciences, and relevant domain disciplines

PLO-4. Design / Development of Solutions

[Breadth and uniqueness of computing problems, i.e., the extent to which problems are original and to which solutions have previously been identified or codified]

Design and evaluate solutions for complex computing problems, and design and evaluate systems, components, or processes that meet specified needs with appropriate consideration for public health and safety, cultural, societal, and environmental considerations

PLO-5. Modern Tool Usage

[Level and appropriateness of the tool to the type of activities performed]

Create, select, adapt and apply appropriate techniques, resources, and modern computing tools to complex computing activities, with an understanding of the limitations

PLO-6. Individual and Team Work

[Role in, and diversity of, the team]

Function effectively as an individual and as a member or leader in diverse teams and in multi-disciplinary settings

PLO-7. Communication

[Level of communication according to type of activities performed]

Communicate effectively with the computing community and with society at large about complex computing activities by being able to comprehend and write effective reports, design documentation, make effective presentations, and give and understand clear instructions

PLO-8. Computing Professionalism and Society

[No differentiation in this characteristic except level of practice]

Understand and assess societal, health, safety, legal, and cultural issues within local and global contexts, and the consequential responsibilities relevant to professional computing practice

PLO-9. Ethics

[No differentiation in this characteristic except level of practice]

Understand and commit to professional ethics, responsibilities, and norms of professional computing practice

PLO-10. Life-long Learning

[No differentiation in this characteristic except level of practice]

Recognize the need, and have the ability, to engage in independent learning for continual development as a computing professional

4 Course Learning Outcomes

The cognition levels are based on Bloom's revised taxonomy.[1]

CLO	Description	Cognition
CLO-1	Apply common algorithmic techniques to standard computational problems	Cog-3
CLO-2	Analyze computational complexity of common/standard algorithms	Cog-4
CLO-3	Design new algorithms for different computational problems	Cog-5
CLO-4	Construct proofs of correctness and time complexity of various algorithms	Cog-6

[1] Anderson, Lorin W.; Krathwohl, David R., eds. (2001). A taxonomy for learning, teaching, and assessing: A revision of Bloom's taxonomy of educational objectives. Allyn and Bacon

	Academic Education	Knowledge for Solving computing Problems	Problem Analysis	Design / Development of Solutions	Modern Tool Usage	Individual and Team Work	Communication	Computing Professionalism and Society	Ethics	Life-long Learning
	PLO-1	**PLO-2**	**PLO-3**	**PLO-4**	**PLO-5**	**PLO-6**	**PLO-7**	**PLO-8**	**PLO-9**	**PLO-10**
CLO-1		✓								
CLO-2			✓							
CLO-3				✓						
CLO-4										✓

5 Format and Procedures:

The LMS site will be used to share the syllabus, give out assignments, and to share other course resources. However, we will heavily rely on Discord for communication, discussions, and announcements.

The University's standard policies on attendance, inclusivity, office hours, and academic integrity apply in this course. These are described in later sections below.

6 Course Requirements

- Class participation policy: Background reading for next session and active participation in class discussions.
- Course readings will be made available online through the LMS as classes progress. The following are good references for the material that will be covered.
  - *Algorithms*, Sanjoy Dasgupta, Christos Papadimitriou, Umesh Vazirani.
  - *Introduction to Algorithms*, Thomas Cormen, Charles Leiserson, Ronald Rivest and Clifford Stein.

  Supplemental reading:

  - *Algorithm Design*, Jon Kleinberg and Eva Tardos.
  - *Randomized Algorithms*, Rajeev Motwani and Prabhakar Raghavan.
  - *Algorithm Design Manual*, Steven Skiena.
  - *Introduction to Analysis of Algorithms*, Robert Sedgewick and Philippe Flajolet.

7 Grading Policy

The course will adapt absolute grading. The final grade will be based on the following components:

Homework assignments	15%
Programming assignments	15%
Quizzes	20%
Midterm exam	20%
Final exam	30%

We expect participation and engagement from all students and expect that students strictly follow the deadlines. All quizzes will be in-person and on-campus. Midterm and final exam will follow the University's policy and will be scheduled by appropriate offices, separately.

7.1 Late Work and Makeup Policy

No late solutions will be accepted and no make-up for exams (midterm or final) or any of the quizzes will be given.

8 Attendance Policy

IBA attendance policy applies.

9 Academic Integrity

Each student in this course is expected to abide by the IBA Code of Conduct. Scholastic dishonesty shall be considered a serious violation of these rules and regulations and is subject to strict disciplinary action as prescribed by IBA regulations and policies. Scholastic dishonesty includes, but is not limited to, cheating on exams, plagiarism on assignments, and collusion.

Kindly refer to `https://examination.iba.edu.pk/CheatingPlagiarism.php` for more details.

- **PLAGIARISM:** Plagiarism is the act of taking the work created by another person or entity and presenting it as ones own for the purpose of personal gain or of obtaining academic credit. Plagiarism includes the submission of or incorporation of the work of others without acknowledging its provenance or giving due credit according to established academic practices. This includes the submission of material that has been appropriated, bought, received as a gift, downloaded, or obtained by any other means. Students must not, unless they have been granted permission from all faculty members concerned, submit the same assignment or project for academic credit for different courses.

- **CHEATING:** The term cheating shall refer to the use of or obtaining of unauthorized in- formation in order to obtain personal benefit or academic credit.

- **COLLUSION:** Collusion is the act of providing unauthorized assistance to one or more person or of not taking the appropriate precautions against doing so.

  Any student violating academic integrity a second time in this course will receive a failing grade for the course, and additional disciplinary sanctions may be administered.

- **SHARING CREDENTIALS:** It has been observed that some students share their credentials (log in id's and passwords) of LMS, portal, email, etc., with with other students. These credentials are private and confidential and not to be shared with anyone. Any violation will be considered as aiding in plagiarism/collusion/cheating and appropriate action might be taken against such students.

10 Office hours

Office hours will be scheduled, circulated, and posted soon. During these hours the course instructors will be available to answer questions or provide additional help.

11 Tentavie list of topics

1. Oh-notations and mathematical preliminaries
2. Sorting and searching
3. Recursion and induction
4. Divide and conquer algorithms
5. Dynamic programming
6. Graphs algorithms
7. Greedy algorithms
8. Randomized algorithms
9. Computational complexity and hardness results
10. Algorithms to cope hardness
11. Introduction to parallel algorithms/computation

12 Weekly breakdown of classes

Week 1: 25/8 – 31/8	**Introduction**
	Oh-notaions
	Mathematical preliminaries
	Merge sort and insertion sort
Week 2: 1/9 – 7/9	**Divide-and-Conquer**
	Closest pair of points
	Integer multiplication
	Strassen's matrix multiplication
	Matrix exponentiation
Week 3: 8/9 – 14/9	**Greedy Algorithms**
	Activity selection and greedy scheduling
	Huffman coding
	Matroid theory basics and greedy correctness
	Counterexamples
Week 4: 15/9 – 21/9	**Dynamic Programming I**
	Longest common subsequence (LCS)
	Edit distance
	Knapsack problem
Week 5: 22/9 – 28/9	**Dynamic Programming II**
	Bellman-Ford algorithm
	Floyd-Warshall algorithm
	Matrix-chain multiplication
Week 6: 29/9 – 5/10	**Some Data Structures**
	Dynamic arrays
	Binary heaps and heap operations
	Priority queues
Week 7: 6/10 – 12/10	**Graphs and Graph Algorithms**
	Topological sort
	Strongly connected component
	Dijkstra's algorithm
	Prim's algorithm
Week 8: 13/10 – 26/10	**Midterm**
Week 9: 20/10 – 26/10	**Mid-semester break**

Week 10: 27/10 – 2/11	**String Algorithms**
	String matching: naive, Rabin-Karp
	Finite automata-based matching
	Knuth-Morris-Pratt (KMP)
	Z-algorithm
Week 11: 3/10 – 9/10	**Randomized Algorithms I**
	Introduction to probability
	Las Vega and Monte Carlo
	Randomized Quicksort and Quick Select
Week 12: 10/11 – 16/11	**Randomized Algorithms II**
	String equality testing
	Random sampling
	Pattern matching
	Primality testing
Week 13: 17/11 – 23/11	**Computational Complexity I**
	P, NP, NP-complete, and NP-hard problems
	Polynomial time reductions
	SAT, 3SAT, Vertex Cover, CLIQUE, etc.
Week 14: 24/11 – 30/11	**Computational Complexity II**
	coNP
	Introduction to space complexity
Week 15: 1/12 – 7/12	**Approximation Algorithms I**
	Performance bounds
	Polynomial approximation schemes
Week 16: 8/12 – 14/12	**Approximation Algorithms II**
	Fully polynomial-time approximation scheme (FPTAS)
	Introduction to parameterized complexity
Week 17-18: 15/12 – 28/12	**Final Exams**