CS315, Algorithm Analysis and Design, Semester (471), year 2025/1447

Guidelines to prepare the course project

Total marks = 20%, five students in each project

Project Overview:

In this project, **groups of 5 students** will engage in a comprehensive study of algorithm design and analysis. Each group will identify a computational problem, develop two algorithmic solutions (a naive approach and an optimized approach for the same idea), implement the solutions, analyze their complexities, and evaluate their performance both theoretically (as in lecture 1) and empirically.

Project Objectives:

- To apply knowledge of algorithm design and analysis.
- To understand the differences between naive and optimized solutions.
- To gain experience with empirical analysis and compare results with theoretical expectations.
- To enhance collaboration, problem-solving, and communication skills.

Group Formation:

Students can choose their groups of 5 within one week or they will be randomly assigned. Groups should appoint a leader to coordinate and communicate with the instructor.

Project Tasks:

1. **Problem Identification:** Select a problem that is suitable for algorithmic solutions.
2. **Algorithm Development:** Propose a naive and an optimized algorithm to solve the problem.
3. **Implementation:** Code both algorithms in a chosen programming language, preferably Python or C++.
4. **Theoretical Analysis:** Determine the asymptotic time and space complexities of both algorithms.
5. **Empirical Analysis:** Test both algorithms with various input sizes and document performance results.
6. **Results Comparison:** Compare theoretical and empirical results, and explain any discrepancies.
7. **Final Report:** Document the project findings in a thorough report.
8. **Presentation:** Present the project to the class, highlighting the approach, analysis, and conclusions.

Deliverables:

The deliverables will consist of the following files, packaged in a zipped file. An example of

1. **Readme file** that includes only the names of the students and their IDs.
2. **Source code** files for both algorithms.
   - The files will be named ***algorithm1.yy*** and ***algorithm2.yy***, where yy is the extension of the source file.
   - If there are multiple files for the same algorithm, name them as ***algorithm1a.yy, algorithm1b.yy,*** and so forth.
3. A **final report in a PDF** format that includes the following:
   - A clear problem description and three illustrative input-output examples.
   - The pseudocode of each algorithm and its description.
   - Complexity analysis.
   - Empirical results.
   - Comparison discussion.

Additional Notes:

- Failure to comply with the guidelines for organizing and naming deliverables, as well as missing the deadline, will result in a deduction of grades.
- Simplistic problems that only require basic algorithms, such as sorting, will lead to a deduction of grades.
- Each group must have its original idea.
- Students are **not** required to make additional materials for the presentation; presenting the report alone will suffice. The emphasis of the presentation is on oral discussion.
- Regular progress checks with the instructor are encouraged.
- Peer review within the group is recommended to improve the quality of deliverables.
- The final grade will be based on the collective output of the group, recognizing the importance of teamwork. Individual contributions may be assessed based on peer evaluations.
- **All group members may receive the same grade.**

This project simulates a real-world scenario where teamwork, problem-solving, and algorithmic skills come together to achieve a common goal. Students are expected to engage deeply with algorithmic concepts and work collaboratively to produce a high-quality academic output.

Project Rubric:

Criteria		points	grading
Problem Identification		**10**	
Clarity	Clear and concise description of the problem.	2	
Relevance	Problem relevance and suitability for algorithmic solutions.	3	
Originality	Uniqueness and originality of the selected problem	2	
Complexity	Appropriateness of problem complexity level	3	
Algorithm Development		**20**	
Naive Algorithm	Correctness and completeness of the naive algorithm	**5**	
Optimized Algorithm	Correctness and completeness of the optimized algorithm	**5**	
Explanation	Clear explanation of the rationale behind each algorithm	**5**	
Pseudocode	Quality and clarity of the pseudocode provided for both algorithms	**5**	
Implementation		**20**	
Functionality	Both algorithms work correctly and efficiently	10	
Code Quality	Code readability, structure, and adherence to standards	5	
Documentation	Adequate comments and README file.	5	
Theoretical Analysis		**20**	
Complexity Analysis	Accurate determination of the time and space complexities	10	
Explanation	Clear explanation of the analysis process	5	
Correctness	Logical and mathematical correctness in the analysis	5	
Empirical Analysis		**10**	
Testing	Comprehensive testing with varied input sizes	5	
Graphs and Tables	Clear and accurate representation of empirical data	5	
Results Comparison		**10**	
Theoretical vs. Empirical	Insightful comparison of theoretical analysis and empirical data	5	
Discrepancies	Well-reasoned explanation of any discrepancies between expected and actual results	5	
Final Report		**10**	
Organization and Writing Quality	Logical structure and organization of the report, as well as clarity, grammar, and spelling	4	
Content	Inclusion of all necessary sections and information	6	
Presentation		**10**	
Content	Clear presentation of problem, algorithms, analysis, and findings	5	
Delivery	Quality of oral communication and engagement with the audience	5	
	Total	**110**	
	Final Total	**20**	