General Instructions (READ CAREFULLY)

1. **Format for Submission:**
    - Solutions to all theoretical problems must be submitted as a PDF document.
    - For coding problems, submit the source code as .py(or other) files (one file per question).
    - Include appropriate comments in your code to explain your logic.
2. **Submission Guidelines:**
    - Submit your homework via the LMS (Learning Management System).
    - If there are any issues with uploading to the LMS, you must email your submissions to me.
3. **Email Structure (if only LMS is not working):**
    - **Subject: Algorithm Analysis Homework 2 Submission - [Your Full Name and Student ID]**

      Dear Dr,

      I am attaching my submission for the Algorithm Analysis homework assignment.

      Attached files:

      1. PDF file containing theoretical solutions.

      2. Separate coding files for coding questions.

      Thank you.

      Best regards,

      [Your Full Name] [Your Student ID]

Deadlines:

- Homework 2 (Dynamic Programming and Greedy Algorithms): [29.11.2024 at 23:59]
- No extensions will be granted, so ensure timely submission.

Evaluation Criteria:

- Correctness of solutions.
- Clarity of explanations and code readability.

- Proper formatting and adherence to submission instructions.
- Following the homework instructions.

Plagiarism:

- All work must be your own. Any copied or plagiarized work will result in zero credit for the assignment.

Homework 2: Dynamic Programming and Greedy Algorithms

Submission Deadline: 29.11.2024 at 23:59.
Instructions:

- Solve all questions and include code for implementation tasks.
- Provide detailed explanations of your solutions and approaches.
- Use Python or a preferred programming language for coding questions.

Part 1: Dynamic Programming

1. **Longest Increasing Subsequence (LIS)**
    - Write a program to find the length of the longest increasing subsequence in a given array.
    - Input: An array of integers.
    - Output: The length of the LIS.
    - Example:
      Input: arr=[10,9,2,5,3,7,101,18] Output: 4 (LIS = [2,3,7,101])
    - *Additional Task*: Explain the time complexity of your solution.
2. **Knapsack Problem**
    - Solve the 0/1 Knapsack Problem using dynamic programming.
    - Input: Weights and values of n items and the capacity of the knapsack.
    - Output: Maximum value that can be achieved and the selected items.
    - Example:
      Input:Weights = [2,3,4,5], Values = [3,4,5,6] , Capacity = 5
      Output: Max Value = 7, Selected items = [Item 2]
      Additional Task: Discuss how this problem can be adapted for a fractional knapsack (Greedy approach).

3. **Coding: Edit Distance**
    - Implement a dynamic programming algorithm to compute the minimum edit distance between two strings X and Y.
    - Input: Two strings.
    - Output: Minimum number of operations (insert, delete, replace) required to convert X into Y.
    - Example: Input: X="sunday",Y="saturday"
    - Output: 3
      Additional Task: Provide an optimized version using space reduction techniques.

Part 2: Greedy Algorithms

4. **Huffman Encoding**
    - Implement the Huffman coding algorithm to compress a given set of characters with their frequencies.
    - Input: A list of characters and their respective frequencies.
    - Output: Huffman codes for each character.
    - Example:
      Input:
      Characters = [a,b,c,d,e,f], Frequencies = [5,9,12,13,16,45],
    - Output: Huffman codes (e.g., a:110,b:111,...)
      Additional Task: Discuss how Huffman coding can fail for non-optimal frequency distributions.
5. **Job Scheduling Problem**
    - Write a greedy algorithm to solve the job scheduling problem with deadlines and profits.
    - Input: A list of jobs with deadlines and profits.
    - Output: The sequence of jobs that maximizes profit.
    - Example:
      Input:Jobs = [Job1,Job2,Job3], Deadlines = [2,1,2], Profits = [100,50,200]
      Output: Maximum profit = 250, Job sequence = [Job3,Job1]

6. **Discussion**
    - Compare the effectiveness of Dynamic Programming and Greedy algorithms. Provide real-world examples where one approach is more suitable than the other.