UCS415 – Design and Analysis of Algorithms
Lab Assignment 2 (Greedy Strategy)
Week 4 and 5

Q1: You are given N activities, each having a start time and a finish time. A single person (or machine) can perform only one activity at a time. Two activities are said to be compatible if the start time of one activity is greater than or equal to the finish time of the other activity. Write a program using greedy strategy to select the maximum number of non-overlapping activities that can be performed by the person.
E.g.:
Input:
N = 6
start = [1, 3, 0, 5, 8, 5]
finish = [2, 4, 6, 7, 9, 9]
Output:
Maximum number of activities = 4
Selected activities: (1, 2), (3, 4), (5, 7), (8, 9)

Q2 Given the arrival and departure times of all trains reaching a railway station on the same day, write a program to determine the minimum number of platforms required so that no train has to wait for a platform. For each train, the arrival time is always different from its departure time, but the arrival time of one train may be equal to the departure time of another train. At any given instant, a single platform cannot be used simultaneously for the departure of one train and the arrival of another train; therefore, in such cases, separate platforms must be allocated. [Minimum Platforms | Practice Problems]
Input:
Train = [T1, T2, T3, T4, T5]
AT = [09:00, 09:10, 09:20, 11:00, 11:20]
DT = [09:40, 12:00, 09:50, 11:30, 11:40]
Output:
Minimum number of platforms required = 3

Q3: You are given N items, where each item has a value and a weight. You are also given a knapsack with a maximum capacity W. Unlike the 0/1 Knapsack problem, you are allowed to take fractions of an item. Write a program to maximize the total value in the knapsack without exceeding its capacity.
E.g.:
Input:
N = 3
value = [100, 60, 120]
weight = [20, 10, 40]
W = 50
Output:
Maximum value = 220

Q4: Given two arrays deadline[ ] and profit[ ], where deadline[i] represents the last time unit by which the *i-th job* must be completed, and profit[i] represents the profit earned from completing it. Each job takes exactly 1 unit of time, and only one job can be scheduled at a time. Write a program to schedule the jobs in such a way that the total profit is maximized while ensuring that each selected job is completed on or before its deadline. Find the number of jobs completed and maximum profit.
E.g.:
Input:
N = 5
Jobs = [J1, J2, J3, J4, J5]
deadline = [2, 1, 2, 1, 3]
profit = [100, 19, 27, 25, 15]
Output:
Selected Jobs: [J1, J3, J5]
Maximum Profit = 142

Q5: Given a set of characters and their corresponding frequencies, write a program to construct the Huffman Tree and generate Huffman codes for each character such that the total number of bits required for encoding is minimized.

E.g.:
Input:
Characters = [a, b, c, d, e, f]
Frequencies = [5, 9, 12, 13, 16, 45]
Output:

Character	Huffman Code
a	1100
b	1101
c	100
d	101
e	111
f	0

Additional Questions

Q1: A chef has opened a restaurant that is divided into K distinct compartments, numbered from 1 to K, where each compartment can be occupied by at most one customer at any given time. Each of the N customers visiting the restaurant is characterized by three parameters: an arrival time, a departure time, and a strongly preferred compartment $p(1 \leq p \leq K)$. A customer can be seated only in their preferred compartment, and if that compartment is already occupied at the time of arrival, the customer immediately leaves the restaurant. To maximize the total number of customers who are successfully served, the chef may choose to allow or disallow certain customers. Given the list of all customers with their arrival times, departure times, and preferred compartments, write a program to determine the maximum number of customers that can dine at the restaurant without violating the compartment constraints. [Bon Appetit Practice Coding Problem]
E.g:
Input:
K = 3, N = 6
C = [C1, C2, C3, C4, C5, C6]
Arrival = [1, 2, 3, 5, 4, 6]
Departure = [4, 5, 6, 7, 8, 9]
Preferred = [1, 1, 2, 1, 2, 3]
Output: Maximum number of customers that can dine = 4

Q2: Alice gives Bob a rectangular board composed of $m \times n$ wooden squares and asks him to determine the minimum total cost required to break the board down into its individual squares by making cuts only along the horizontal and vertical grid lines. Bob can perform horizontal and vertical cuts across the entire current board, where each horizontal cut has an associated cost x_i and each vertical cut has an associated cost y_j. The cost of making a cut is equal to the given cut cost multiplied by the number of board segments it crosses at that time. As the board is progressively divided, the number of segments increases, thereby increasing the effective cost of subsequent cuts. The total cost of reducing the board into individual squares is defined as the sum of the costs of all cuts performed in sequence. The objective is to design a greedy strategy–based algorithm that selects the order of horizontal and vertical cuts such that the overall cutting cost is minimized. [Cutting Boards | HackerRank]
E.g.:
Input: m = 3, n = 3
Output:
Horizontal cut costs (x): 2 1
Vertical cut costs (y): 3 1

Q3: Given a string s consisting only of lowercase English characters and an integer k, design a greedy strategy–based algorithm to generate the lexicographically smallest possible string after removing exactly k characters from the string, with the additional constraint that the value of k must first be modified based on the length of the string. If the length of the string is a power of 2, the value of $k$is reduced to $k/2$; otherwise, the value of $k$is doubled to $2k$. After modifying k, exactly k characters can be removed from any positions in the string. The objective is to ensure that the resulting string is the smallest in lexicographical order among all possible valid removals. [Lexicographically smallest String by removing exactly K characters - GeeksforGeeks]
E.g.:
Input: S = "fooland", k = 2
Output: "and"
Explanation: As the size of the string = 7, which is not a power of 2, hence K = 4. After removing 4 characters from the given string, the lexicographically smallest string is "and".

Q4: There are *n* gas stations along a circular route, where the amount of gas at the i^{th} station is gas[i]. You have a car with an unlimited gas tank and it costs cost[i] of gas to travel from the i^{th} station to its next $(i + 1)^{th}$ station. You begin the journey with an empty tank at one of the gas stations. Given two integer arrays gas and cost, return the starting gas station's index if you can travel around the circuit once in the clockwise direction, otherwise return -1. If there exists a solution, it is guaranteed to be unique. [Gas Station - LeetCode]
E.g.:
Input: gas = [1,2,3,4,5], cost = [3,4,5,1,2]
Output: 3

Q5: Chefland has a renowned university that offers N courses, where each course is conducted over a continuous range of days. The i^{th} course starts on day $start_i$ and ends on day end_i. Chef wishes to enroll in the university but is unsure about the exact duration of study and therefore considers Q different tentative study plans. Each plan is defined by a starting day $plan\_start_j$ and an ending day $plan\_end_j$. For each study plan, Chef wants to determine the maximum number of courses he can complete if he follows that plan. Chef can attend at most one course on any given day, and a course is considered completed only if Chef attends all its classes, meaning the entire duration of the course must lie within the selected study plan. Design an efficient algorithm to compute the maximum number of courses that Chef can complete for each of his study plans. [Chef and his study plans Practice Coding Problem]
E.g.:
Input:
N = 6
St_i = [1, 2, 4, 6, 5, 8]
End_i = [3, 5, 6, 7, 9, 10]
Output:
Q = 3
Plan_st_j = [1, 2, 4]
Plan_end_j = [6, 7, 10]