**DSA Questions**

**Lab Assignment 1:**  
*Problem1:*  
Simran is running up a staircase with N steps, and can hop(jump) either 1 step, 2 steps or 3 steps at a time. You have to count, how many possible ways Simran can run up to the stairs.  
Input Format:  
Input contains integer N that is number of steps  
Output Format:  
Output for each integer N the no of possible ways w.  
Constraints  
1≤N≤30  
SAMPLE INPUT: 4  
SAMPLE OUTPUT: 7  
  
*Problem2:*  
Patlu and Motu works in a building construction, they have to put some number of bricks  in a building construction, they have to put some number of bricks N from one place to another, and started doing their work. They decided , they end up with a fun challenge who will put the last brick. one place to another, and started doing their work. They decided , they end up with a fun challenge who will put the last brick.  
They to follow a simple rule, In the i'th round, Patlu puts i bricks whereas Motu puts ix2 bricks. There are only N bricks, you need to help find the challenge result to find who put the last brick you need to help find the challenge result to find who put the last brick.  
Input: First line contains an integer N.  
Output: Output  
"Patlu" (without the quotes) if  (without the quotes) if Patlu puts the last bricks ,"Motu"(without the quotes) otherwise. the quotes) otherwise.  
Constraints: 1 ≤ N ≤ 10000  
10000  
Sample Input: 13  
Sample Output: Motu

**Lab Assignment 2:**  
  
*Program 1:*

Write  a program  that  prompts  for and reads  in  a string, then prints  a message  saying whether it  is  a palindrome.

*Program 2:*

Write a program that reads in two integers and uses  recursion to compute the first raised to the second power.

**Lab Assignment 3:**  
  
*Program 1:*

Write a Program to Reverse a given stack of integers using recursion. Create a recurrence relation and calculate its Time Complexity.  
  
*Program 2:*

Given a string 'S'. Write a program to find an array of all possible strings in any order that can be formed by transforming every letter individually to the lowercase or the uppercase.  
Note:  
1. You can print the array in any order.  
2. The string 'S' only contains English alphabets and digits.  
  
Sample Input 1:  
a1b  
Sample Output 1:  
A1B A1b a1B a1b  
  
Sample Input 2:  
0  
Sample Output 2:  
0  
Explanation: There is no alphabet in the string, so we get the output the same as the input string.

**Lab Assignment 4:**  
*Program 1:*

Given a Stack 'S'. Write a Python Program to SORT the stack recursively in decreasing order.  
Note: Looping through the stack is not allowed. You need to return a stack that is sorted in decreasing order.  
  
Example:  
Given stack: 5 1 2 6 4  
Output: 6 5 4 2 1  
  
  
*Program 2:*

Given a text string containing characters only from lowercase alphabetic characters and a pattern string containing characters only from lowercase alphabetic characters and two other special characters '.' and '\*'.  
  
Write a Python program for pattern matching that returns true if pattern is matched with text otherwise returns false using follow the Brute force approach. The matching must be exact, not partial.  
  
Explanation of the special characters:'.' - Matches a single character from lowercase alphabetic characters.'\*' - Matches zero or more preceding character. It is guaranteed that '\*' will have one preceding character which can be any lowercase alphabetic character or special character '.'. But '\*' will never be the preceding character of '\*'. (That means "\*\*" will never occur in the pattern string.)\ '.' = "a", "b", "c", ... , "z"\ a\* = "a", "aa", "aaa", "aaaa",... or empty string("")\ ab\* = "a", "ab", "abb", "abbb", "abbbb", ...  
Example One {  
"text": "abbbc",  
"pattern": "ab\*c"  
}  
Output:1  
  
Given pattern string can match: "ac", "abc", "abbc", "abbbc", "abbbbc", ...  
Example Two {  
"text": "abcdefg",  
"pattern": "a.c.\*.\*gg\*"  
}  
Output:1  
  
  
Example Three {  
"text": "abc",  
"pattern": ".ab\*.."  
}  
Output:0

**Lab Assignment 5:**  
*Problem:*

Given a sorted array A containing N integers both positive and negative.  
You need to create another array containing the squares of all the elements in A and return it in non-decreasing order.  
Write a Python program to implement this in O(N) time.  
  
Example Input  
  
Input 1:  
A = [-6, -3, -1, 2, 4, 5]  
Input 2:  
A = [-5, -4, -2, 0, 1]  
  
Example Output  
Output 1:  
[1, 4, 9, 16, 25, 36]  
Output 2:  
[0, 1, 4, 16, 25]

**Lab Assignment 6:**

*Problem 1:*

Given an M × N matrix, which is row-wise and column-wise sorted (with all strictly increasing elements in any row or column), write a program in Python to report all occurrences of a given element in it in linear time.

For example,

Input:

[ -4 -3 -1  3  5 ]

[ -3 -2  2  4  6 ]

[ -1  1  3  5  8 ]

[  3  4  7  8  9 ]

Output:

Element 3 is found at position (0, 3)

Element 3 is found at position (2, 2)

Element 3 is found at position (3, 0)

*Program 2:*

Write a Python program to perform Matrix multiplication using Strassen's method

**Lab Assignment 7:**

*Problem 1:*

As you saw in class, greedy algorithms are often used when we have an optimization problem involving a series of choices. In today’s lab, we will explore one such example problem. We would like to maximize the number of jobs that can be done over a period of time.

If we are given, say, 10 jobs, each with its start time and finish time, we want to select the largest possible subset of these jobs in such a way that no two jobs in the subset overlap in time. In other words, we can only work on one job at any one time. For simplicity, assume that time is measured by whole numbers. The smallest interval of time we will measure will be between consecutive integers. It’s okay for one job to begin at the same instant another job finishes.

We’re going to solve this problem using a greedy algorithm, as we did in class. Here is the pseudocode:

1. Sort the jobs ascending by finish time.

2. Let W be our chosen subset of jobs to-do, and initialize it to be empty.

3. for j = 1 to n

if job j does not overlap in time with W

add j to W

To test this algorithm, we need an initial set of candidate jobs. So, we’ll create a collection of 10 jobs. Their start and finish times will be random. For example, the start time for each job can be a random integer value from 0 to 9 inclusive. And we can set the duration of each job to be a random integer from 2 to 6 inclusive. But these constants are just implementation parameters that have no effect on the optimality of the algorithm, and we can tweak these numbers later if necessary.

Your script should contain:

Attributes for start time, duration of job, and how many other jobs it conflicts with

• Function that sets attributes

• Methods for start time, duration, finish time, number of conflicts.

Method for the number of conflicts (because at the time of creation we won’t know this value).

• Methods to give the start time, ending time and the number of conflicts.

Main Script should do the following:

Create a list of 10 jobs.

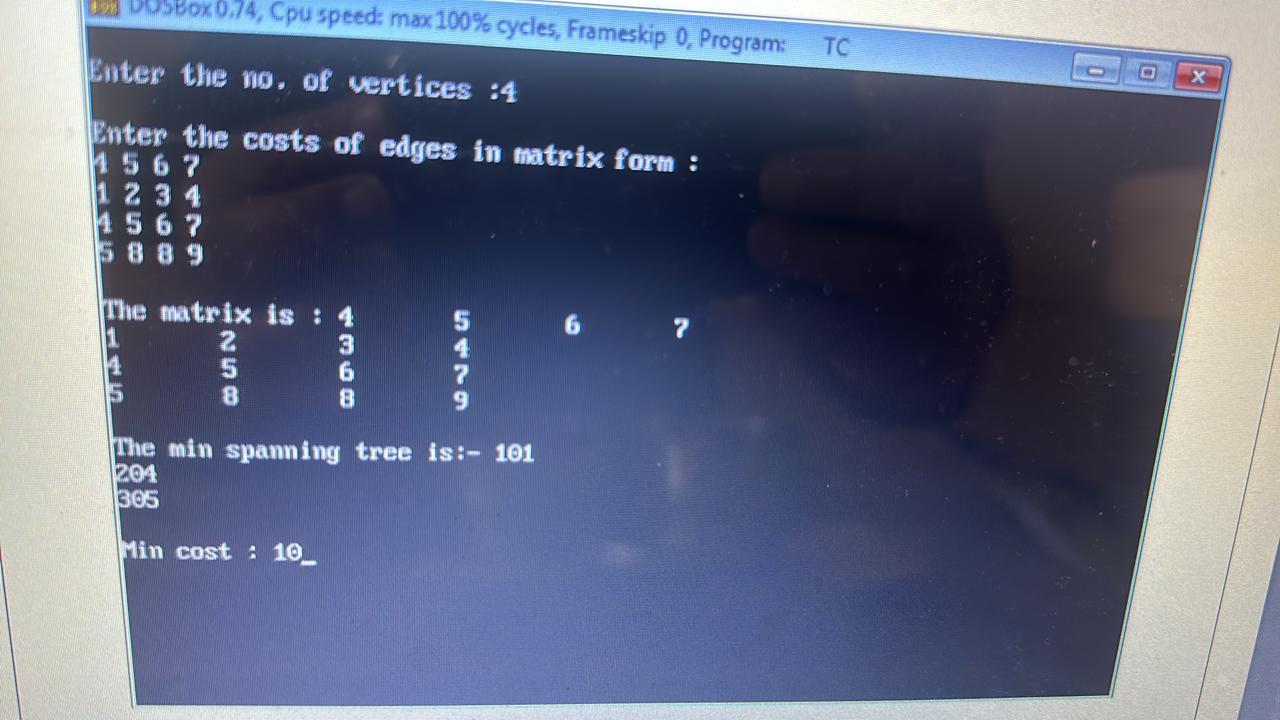
• Perform the greedy algorithm above to generate a “to-do” subset list of jobs to schedule.

• Output the to-do list.

• Functions to determine: o whether a job has a conflict with an existing to-do list o for each job, the number of conflicts it has with the other jobs in the entire list.

*Problem 2:*

Ask them to write a program to implement prims algorithm.

****

**Lab Assignment 8:**

*Program 2:*

Given a text string containing characters only from lowercase alphabetic characters and a pattern string containing characters only from lowercase alphabetic characters and two other special characters '.' and '\*'.

Write a Python program for pattern matching that returns true if pattern is matched with text otherwise returns false. The matching must be exact, not partial.

Explanation of the special characters:

'.' - Matches a single character from lowercase alphabetic characters.

'' - Matches zero or more preceding character. It is guaranteed that '' will have one preceding character which can be any lowercase alphabetic character or special character '.'. But '' will never be the preceding character of ''. (That means "\*" will never occur in the pattern string.)\ '.' = "a", "b", "c", ... , "z"\ a = "a", "aa", "aaa", "aaaa",... or empty string("")\ ab\* = "a", "ab", "abb", "abbb", "abbbb", ...

*Problem1:*

Given two strings A and B, find the minimum number of steps required to convert A to B. (each operation is counted as 1 step.)

You have the following 3 operations permitted on a word:

Insert a character

Delete a character

Replace a character

Input Format:

The first argument of input contains a string, A. The second argument of input contains a string, B.

Output Format:

Return an integer, representing the minimum number of steps required.

Constraints:

1 <= length(A), length(B) <= 450

Examples:

Input 1: A = "abad" B = "abac" Output 1: 1 Explanation 1: Operation 1: Replace d with c. Input 2: A = "Anshuman" B = "Antihuman" Output 2: 2 Explanation 2: => Operation 1: Replace s with t. => Operation 2: Insert i.

**Lab Assignment 9:**

*Problem1:*

Write a program in Python to solve 0 1 knapsack knapsack using Dynamic programming.

*Program 2*

￼

Tushar's Birthday Bombs

Medium

37.1% Success

113

41

Bookmark

Max Score: 200

Hint

Solution Approach

Complete Solution

Problem Description

It’s Tushar’s birthday today and he has N friends.

Friends are numbered [0, 1, 2, ...., N-1] and ith friend have a positive strength B[i].

Today being his birthday, his friends have planned to give him birthday bombs (kicks).

Tushar's friends know Tushar's pain bearing limit and would hit accordingly.

If Tushar’s resistance is denoted by A (>=0) then find the lexicographically smallest order of friends to kick Tushar so that the cumulative kick strength (sum of the strengths of friends who kicks) doesn’t exceed his resistance capacity and total no. of kicks hit are maximum.

Also note that each friend can kick an unlimited number of times (If a friend hits x times, his strength will be counted x times)

Return the lexicographically smallest array of maximum length where the ith index represents the index of the friend who will hit.

NOTE:

[a1, a2, ...., am] is lexicographically smaller than [b1, b2, .., bm] if a1 < b1 or (a1 = b1 and a2 < b2) ... .

Input cases are such that the length of the answer does not exceed 100000.

Problem Constraints

1 <= N <= 100

1 <= A <= 15 \* 106

1 <= B[i] <= 105

Input Format

The first argument contains an integer, A of length N.

The second argument contains an array of integers B.

Output Format

Return an array of integer, as described in the problem statement.

Example Input

Input 1:

A = 12 B = [3, 4]

Input 2:

A = 11 B = [6, 8, 5, 4, 7]

Example Output

Output 1:

[0, 0, 0, 0]

Output 2:

[0, 2]