Indian Institute of Technology Ropar

Data Structure and Algorithms

CS506 and CS205

Lab Assignment 05 | | The Final Assignment

[Deadline: 3rd November 2024]

Motivation:

In this final assignment, you need to implement Greedy and Dynamic Programming approaches in some standard problems.

Problems:

(A) Longest Common Subsequence Problem (LCS Problem):

Given three strings str1, str2, & str3 **(all will be in lower case only)** of length p, q, and r respectively, return their longest common subsequence. If there is no common subsequence, return **<NONE>** as a string.

Constraints:

Time complexity: O(p*q*r)

(B) Knapsack Problem:

While given the knapsack capacity, number of items, and their corresponding weights and valuation, you need to implement various versions of Knapsack Problems mentioned below:

(a) Fractional Knapsack Using Greedy Algorithm
(b) 0/1 Knapsack Using Greedy Algorithm
(c) 0/1 Knapsack Using Dynamic Programming

Please Note, that for all the KS Problems, input will be **Integer Only**.
But output will be in the form of **floating point numbers with 2 decimal places** as Fractional KS can include a portion of an item**.**

(C) Johnson's Algorithm for All Pair Shortest Path:

Given the adjacency matrix of a weighted directed graph, you need to find the all-pair shortest path using Johnson's algorithm.

Input and output will be Integer data only. The graph will not contain a negative weight cycle. And it will not be disconnected.

References:
https://en.wikipedia.org/wiki/Johnson%27s_algorithm
https://www.javatpoint.com/johnsons-algorithm

Sample Input, Output Format:

(A) LCS Problem:

Format:

First Line will be the Number of Test Cases
Then for each test case, there will be three lines for three strings

Need to print the LCS

Input:

2
rejoy
madhav
shradha
shruti
smriti
kirti

Output:

<None>
rti

(B) Knapsack Problem:

Format:

The first line will be the number of Test Cases,
then for each test case,
First, choose which algorithm wants to perform:
(1) Fractional KS using Greedy
(2) 0/1 KS using Greedy
(3) 0/1 KS using DP
Number of items N
Total Knapsack Weight W
N valuation of each item
N weights of each item

Need to print the optimal solution

Input:

```
3
1               #Selecting FKS using Greedy
3
50
60 100 120
10 20 30
2               #Selecting 01KS using Greedy
3
50
60 100 120
10 20 30
3               #Selecting 01KS using DP
3
```

50
60 100 120
10 20 30

Output:

240.00
160.00
220.00

(C) Johnson's Algorithm

Format:

First number of test cases
Then for each test case,
Number of nodes N in the graph
Then N rows of input with N column values

Need to print the distance matrix from each node to each node
(If not reachable then print X)

Input:

2
4
0 -3 0 2
5 0 3 0
1 0 0 0
-1 0 4 0
4
0 -5 2 3
0 0 4 0
0 0 0 1
0 0 0 0

Output:

0 -3 0 2
4 0 3 6
1 -2 0 3
-1 -4 -1 0
0 -5 -1 0
X 0 4 5
X X 0 1
X X X 0

Programming Languages Allowed:

C/C++

(You may use STL libraries for stack, queues, list, vector etc.)

(But using any library/function which directly solves the problem itself, is not allowed)

Expected File(s):

One single file of .c / .cpp

Naming Convention of Files

FirstName_EntryNo_CS506_Lab05_ProgA.c (or .cpp)
FirstName_EntryNo_CS506_Lab05_ProgB.c (or .cpp)
FirstName_EntryNo_CS506_Lab05_ProgC.c (or .cpp)
**Students of CS205 will write CS205 instead of CS506*

Example:-

Rejoy_2023CSM1011_CS506_Lab05_ProgA.c (or .cpp)
Rejoy_2023CSM1011_CS506_Lab05_ProgB.c (or .cpp)
Rejoy_2023CSM1011_CS506_Lab05_ProgC.c (or .cpp)

Instructions for Programming:

- Don't put unnecessary pieces of information or codes.
- Good documentation in the codes is highly appreciated. (You may use comments)
- Please remember to make modular codes.

Plagiarism Checking

Do not copy from each other. Plagiarism will be checked. After a certain threshold, a high penalty or even an F grade can be imposed, depending on the plagiarism rate.

Feel Free To Contact The TAs:

Rejoy Chakraborty
Madhav Mishra
Shradha Sharma