Data Structures and Algorithms (CS F211)
Second Semester 2022-2023
Lab Sheet 1

Instructions:

1. The programs are to be written only in C programming language.
2. Each problem statement is associated with sample inputs and outputs. You have to adhere to these input and output formats.
3. For all integer variables, long int should be of sufficient size.
4. Assume maximum string (single as well as multi-word) sizes of 1000.

1. Write a C program that takes an m x n matrix A as input from the user where m and n are any positive integers and are not necessarily equal. Note that m and n are to be input by the user. Matrix A can contain any integer (positive as well as negative) as its elements. Your program should find another m x n matrix B such that B[i][j] = A[i][j] if i = 0, j = 0 and B[i][j] = A[0][0] + A[0][1] + … + A[i][j] if i ≠ 0 and j ≠ 0. For e.g., If A is a 3 x 3 matrix, then B[1][2] = A[0][0] + A[0][1] + A[0][2] + A[1][0] + A[1][1] + A[1][2]. Here, the calculation of only B[1][2] is shown just to illustrate the process of calculating the matrix B. The user also gives a row index ri, a column index ci, a row height rh and a column height ch as inputs (ri ≥ 0, ci ≥ 0, rh ≥ 1, ch ≥ 1). You have to find out the sum of the elements of the sub-matrix of B starting at B[ri][ci] and of size rh x ch. Note that while finding out the sub-matrix of B, you have to move right as well as down from B[ri][ci]. You will also have to check whether it is possible to create a sub-matrix of B of size rh x ch by moving right and down from B[ri][ci]. Your code should display the sub-matrix of B and the sum.

Input format: The first line contains the value of m followed by the value of n given as 2 space separated integers. The subsequent m lines contain the elements of A, each line containing n values separated by spaces. The last line contains 4 integers in the following order, separated by spaces – ri, ci, rh, ch.

Output format: The elements of B are displayed in m lines with each line displaying n space separated values. If it is not possible to create the sub-matrix of B, then print NOT POSSIBLE in the next line. If it is possible to create the sub-matrix of B, then display the sum of the elements of the sub-matrix in the subsequent line.

Input 1:
3 3
1 2 3
9 4 0
1 8 5
0 1 3 2
Output 1:
1 3 6
15 19 19
20 28 33
108

Input 2:
3 3
1 2 3
9 4 0
1 8 5
2 2 3 2
Output 2:
1 3 6
15 19 19
20 28 33
NOT POSSIBLE

2. Write a C program that takes two matrices, A and B of size m x n and p x q respectively. Note that m, n, p, q and the elements of A and B are given as inputs from the user. The matrices A and B can be either square or rectangular. Your program has to print YES if matrix B is a sub-matrix of A, i.e., all elements of B are present in A in contiguous rows and columns. Your program should print NO if no row of B matches with any row of A and no column of B matches with any column of B. If only a sub-set of the rows and/or columns of B matches with A, then print PARTIAL. In cases of outputs YES and PARTIAL, the row and/or column matchings have to be contiguous, i.e., the elements should be present contiguously in the row or column. You do not have to consider the cases of wrapping around the rows and columns.

Input format: The first line contains the value of m followed by the value of n given as 2 space separated integers. The subsequent m lines contain the elements of A, each line containing n values separated by

spaces. The next line contains the value of p followed by the value of q given as 2 space separated integers. The next p lines contain the elements of B, each line containing q values separated by spaces.

Output format: Print YES, NO or PARTIAL

Input 1:
3 4
1 2 6 10
3 4 7 11
5 6 9 12
2 2
4 7
6 9
Output 1:
YES

Input 2:
3 4
1 2 6 10
3 4 7 11
5 6 9 12
2 2
0 7
5 10
Output 2:
NO
Explanation: Though 3 elements of B match with A, but note that the rows (0, 7) and (5, 10) and the columns (0, 5) and (7, 10) do not match with any row or column of A respectively.

Input 3:
3 4
1 2 6 10
3 4 7 11
5 6 9 12
2 2
0 7
1 10
Output 3:
NO
Explanation: Though 3 elements of B match with A, but note that the rows (0, 7) and (1, 10) and the columns (0, 1) and (7, 10) do not match with any row or column of A respectively in a contiguous fashion. In other words, though 1 and 10 are present in the same row of A, they are not present in contiguous columns.

Input 4:
3 4
1 2 6 10
3 4 7 11
5 6 9 12
2 2
4 7
5 12

Output 4:
PARTIAL
Explanation: Only the row (4, 7) matches contiguously.

3. Let A be an array of n positive integers. A positive integer s is said to be a subset sum of A if there exist indices $0 \le i_1 < i_2 < \ldots < i_k \le n - 1$ such that $s = A[i_1] + A[i_2] + \ldots + A[i_k]$. Note that the array A need not be sorted, and may contain duplicate entries. The indices constituting the subset sum may not be contiguous. The values of n and s and the elements of A are given as user inputs. The given sum s may be realized in multiple ways. If the subset sum is possible, your program should print POSSIBLE and at least one way (the sequence of the indices of A in sorted order) in which the subset sum is possible, otherwise print NOT POSSIBLE. Note that a subset sum is also trivially possible such that $s = A[i]$ $(0 \le i \le n - 1)$. However, you cannot use an element of A more than once to construct the subset sum. Note that it is not mandatory that i_1 will always start from 0.

Input format: The first line contains the value of n The next line contains the elements of A separated by white spaces. The last line contains the value of s.

Output format: If the subset sum is possible, print POSSIBLE in the first line and the sequence of indices in sorted order separated by white spaces in the next line. Otherwise, print NOT POSSIBLE.

Input 1:
5
6 4 7 4 9
23
Output 1:
POSSIBLE
0 1 3 4
Explanation:
23 = A[0] + A[1] + A[3] + A[4] = 6 + 4 + 4 + 9

Input 2:
5
6 4 7 4 9
12
Output 2:
NOT POSSIBLE

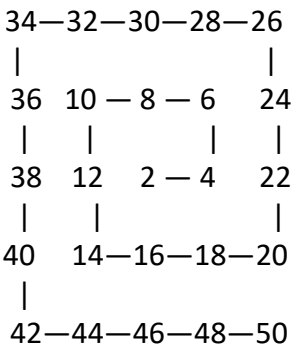
4. Let A be an array of n integers. All the elements of A are non-zero. A contains any mix of positive and negative integers. The value of n and the elements of A are given as user inputs. A can contain duplicates. A is guaranteed to always contain a mix of positive and negative integers. However, the number of positive and negative integers need not be equal. Write a C program that modifies A itself so that its negative elements come before its positive elements. You are not allowed to use any additional array or data structure. Moreover, you need to preserve the relative ordering of the positive and negative elements of A, i.e., if both A[i] and A[j] are positive (or negative) elements and A[i] precedes A[j] in the initial input, then in the final output also A[i] should precede A[j].

Input format: The first line contains the value of n. The next line contains the elements of A separated by white spaces.

Output format: The elements of the modified array A in a single line. The elements are separated by white space characters.

Input 1:
8
7 -1 -6 8 4 -13 9 7
Output 1:
-1 -6 -13 7 8 4 9 7
(This is the only correct modified A. Any other ordering of elements will be considered incorrect as the relative ordering of elements will not be preserved.)

5. Write a C program to create a number spiral of the form depicted below arranged as an n x n matrix. n is given as user input. n can be as large as 100.



Input format: The single line of input contains the value of n.

Output format: The n x n number spiral as depicted above. Note that it is not enough to only print the numbers. You have to display the spiral.

Input 1:
5
Output 1:
34—32—30—28—26
36 10 — 8 — 6 24
38 12 2 — 4 22
40 14—16—18—20
42—44—46—48—50

6. Given two numbers A and B, write a C program to find the digit which occurs the maximum number of times in the primes between A to B (both A and B inclusive). If multiple answers are possible, print any one of them.

Input format: The only line of input contains two space-separated integers A and B ($2 \le A \le B \le 2^{30}$, $0 \le |B-A| \le 106$).

Output format: Print two space-separated integers X, Y with X denoting the digit which occurs the maximum number of times in the primes from A to B (both inclusive) and Y denoting the frequency of X.

Input 1:
37 101
Output 1:
7 7

Input 2:
2692 15859
Output 2:
1 1367

7. You are given a sentence with many words and the words are separated by white spaces. Assume that the sentence contains only lowercase alphabets/letters. You need to find the minimum window in that sentence that contains all the letters given in another word. Both this word and the sentence are given as user inputs. Note that the minimum window substring can contain a few other letters other than the letters in the given word, but strictly contains all the letters of the given word in the required frequency and has minimum substring length among all such windows.

Input format: The first line contains a single string S denoting the sentence. The next line contains the word W whose letters have to be found. W is always going to be a single word with no white space characters.

Output format: If a minimum window exists, print the starting and ending indices of the window (1-indexed, both ends inclusive). In the following line, print the substring itself preserving all the relevant white space characters. If there are multiple correct answers, print any. If no such window exists, print NO WINDOW.

Input 1:
this is a test string for the problem
tist
Output 1:
13 18
t stri
(*Note that W contains two t's. So in the final output, the minimum window also contains two t's.*)

Input 2:
this is a test string for the problem
tixy
Output 2:
NO WINDOW

8. Given a sentence string S, your task is to print all the words in the string in reverse order and then, find all the palindromic words in the sentence. S can contain a single word or multiple words. If S contains multiple words, the words are white space separated. S can contain duplicate words as well. Moreover, S can contain duplicate palindromes as well, in which case, you need to print all such palindromes in the output. Do not consider a single character word as a palindrome (like a or I).

Input format: The only line contains a single string S, either single word or multi-word.

Output format: In the first line print the modified string S with all its word(s) reversed. In the following line(s), print the palindromic word(s) in the original S. The palindromic words can be printed in any order. If there are no palindromic words in the sentence, print NO PALINDROMES.

Input 1:
the kayak could not be detected on the radar
Output 1:
eht kayak dluoc ton eb detceted no eht radar
kayak radar

Input 2:
add a level after the last level
Output 1:
dda a level retfa eht tsal level
level level

Input 3:
apples and oranges
Output 3:
selppa dna segnaro
NO PALINDROMES

Input 4:
madam
Output 4:
madam
madam

9. You are given an array A of n positive integers in the range 1 to k (1 and k are inclusive in the range). The values of n and k and the elements of A are given as user inputs. The elements of the array need not be present in sorted order. Note that it is guaranteed that n and k are odd integers. Moreover, it is also guaranteed that the element (1+k)/2 is present in the array. However, it can be present in the array at any position. Let us call this element as mid. Rearrange the elements of A such that mid occupies the middle position (not the central index) of the array, all elements lesser than the mid are placed to the left of mid and

all elements greater than mid are placed to the right of mid (See sample input-output for better clarity). You are not allowed to use any additional array or data structure. Note that mid is unique in the array. However, there can be duplicates of other elements of the array. In the final output, you should maintain the relative ordering of the elements of the original input array A. It is not necessary that the number of elements lesser than mid and the number of elements greater than mid are equal.

Input format: The first line of input contains the value of k. The second line of input contains the value of n. The last line of input contains the elements of A separated by white spaces.

Output format: The rearranged contents of A in a single line separated by white spaces.

Input 1:
501
9
25 251 350 400 181 237 412 399 8
Output 1:
25 181 237 8 251 350 400 412 399
(*Any other ordering will be incorrect since it will not preserve the relative ordering*)

Input 2:
987
13
13 5 510 1080 890 393 45 494 123 680 510 257 188
Output 2:
13 5 393 45 123 257 188 494 510 1080 890 680 510
(*Note how the two 510s are arranged. Also, note that the number of elements to the left of mid is 7 and the number of elements to the right of mid is 5.*)

10. An anagram is a word or phrase formed by rearranging the letters of a different word or phrase using all the original letters exactly once. You are given 2 single word strings S and A. You need to find out if A is an anagram of S. If yes, then print POSSIBLE and a sequence of the indices of the characters of S which produces A. If A is not an anagram of S, then print NOT POSSIBLE. Assume 1-based indexing for both the strings, i.e., the first characters of both S and A have index 1. Also, assume that S and A consist of only lowercase characters.

Input format: The first line contains the string S. The second line contains the string A. Both S and A are guaranteed to be of the same size.

Output format: If A is an anagram of S, then print POSSIBLE in the first line and the sequence of characters of S (in terms of indices of S) that creates A as a space separated list in the next line. If A is not an anagram of S, then print NOT POSSIBLE.

Input 1:
part
trap
Output 1:
4 3 2 1

Input 2:
earth
heart
Output 2:
5 1 2 3 4

Input 3:
pitch
porch
Output 3:
NOT POSSIBLE

****