Data Structures and Algorithms (CS F211)
Second Semester 2018-2019
Lab Sheet 6

1. Given an array ***P*** with elements $p_1, p_2, ..., p_n$ and an ***n×n*** sized binary symmetric matrix ***A***, you have to find the lexicographically minimum permutation which can be achieved by swapping two distinct elements p_i and p_j where the condition ***A[i][j]= 1*** holds and $1 \le i, j \le n$.

Input:
First line contains ***n***.
Second line contains ***n*** elements of array ***P***.
Each of the next ***n*** lines contains n^2 integers which denote matrix ***A*** as a whole. ***A[i][j] = 1*** means p_i and p_j can be swapped, otherwise no swapping can be done.

Output:
The lexicographically minimum permutation that can be achieved by swapping.

Example:
Input:
7
5 2 4 3 6 7 1
0001001
0000000
0000010
1000001
0000000
0010000
1001000

Output:
1 2 4 3 6 7 5

2. You want to sort ***n*** different arrays (each of size ***m***) simultaneously. You can choose a pair of distinct indices ***i*** and ***j*** ($1 \le i, j \le m, i \ne j$). Then, in each array, the values at positions ***i*** and ***j*** are swapped only if the value at position ***i*** is strictly greater than the value at position ***j***. You want to find an array containing pairs of distinct indices that, chosen in order, will sort all of the ***n*** arrays in ascending order or descending order.

Input:
Two integers ***n*** (no. of arrays) and ***m*** (size of each array) and ***k*** (***k*** is zero if sorting is to be done in ascending order and 1 if sorting is to be done in descending order).
Each of the next ***m*** lines contains the arrays of size ***m***.

Output:
In the first line of the output, print an integer ***p***, the size of the array (***p*** can be at most ***m*(m-1)/2**). Each of the next ***p*** lines must contain two distinct integers ***i*** and ***j*** ($1 \le i, j \le m, i \ne j$), representing the chosen indices.

Example:
Input:
2 5 0
1 3 2 5 4
1 4 3 2 5

Output:
3
2 4
2 3
4 5

3. You are a dog lover and thus you don't like cats. You live by an unusual park. The park is a rooted tree of ***n*** vertices with root node as 1. Vertex 1 also contains your house. Unfortunately, some vertices also contain cats and you know those vertices. You want to travel to each of the leaves of the tree starting from your house. But since you don't like cats, if you encounter ***m*** consecutive cats in your path from your house to a leaf node then you drop the path.

Input:
The first line contains two integers, ***n*** and ***m*** — the number of vertices of the tree and the maximum number of consecutive vertices with cats that is still ok for you.
The second line contains ***n*** integers $a_1, a_2, ..., a_n$ where each a_i either equals to 0 (then vertex ***i*** has no cat), or equals to 1 (then vertex ***i*** has a cat).
Each of the next ***n-1*** lines contains the edges of the tree in the format x_i, y_i, where x_i and y_i are the vertices of the tree, connected by an edge.

Output:
A single integer — the number of distinct leaves of a tree, the path to which from your home contains at most ***m*** consecutive vertices with cats.

Example:
Input:
4 1
1 1 0 0
1 2
1 3
1 4
Output:
2

4. You have ***n*** different socks with you. Each of the socks is of a colour. The total colours are ***k***. Your mom is going out for a vacation and has made you a schedule for the ***m*** days she is going out for. For each of the ***m*** days, she has written down ***l[i], r[i]*** (***l[i]*** denoting the left sock and ***r[i]*** denoting the right sock) which you will wear for the day. Now she was in a hurry and didn't realise that you can't wear socks of different colours on the same day. However, you possess ***k*** jars of paint – one for each colour. You want to fix the situation by swapping socks between the given pairs or/and recolouring some of them. Note that you want to change the colours of minimum number of socks.

Input:
The first line of input contains three integers ***n, m*** and ***k***— the number of socks, the number of days and the number of available colours respectively.
The second line contain ***n*** integers $c_1, c_2, ..., c_n$ $(1 \le c_i \le k)$ — current colours of your socks.
The third line contains the number of socks for each colour.
Each of the following ***m*** lines contains two integers l_i and r_i— indices of socks which you should wear during the ***i***-th day.

Output:
Print one integer — the minimum number of socks that should have their colours changed in order to be able to obey the instructions.

Example:
Input:
4 3 3
1 2 3
1 2 1
1 2
2 3
3 1

Output:
1

Explanation: Put '2' from second pair to the first pair. The first pair is then 2 2 and the second pair is 1 3 and the third pair is 3 1. Now, recolour sock '1' to '3' then your second pair is 3 3 and the third pair is also 3 3. You will be wearing the same colour of socks for two consecutive days. Thus, the situation is fixed by coloring only one sock.

5. Merging two arrays is an essential step in merge sort. Can we do the same in case of linked list? Given 2 sorted linked lists, devise a way to merge them while only using O(1) memory ?

Input:
First line contains the size of the first sorted array.
Second line contains the elements of the first sorted array.
Third line contains the size of the second sorted array.
Fourth line contains the elements of the second sorted array.

Output:
The final merged sorted array.

Example:
Input:
5
1 2 3 4 5
6
2 2 4 4 5 5

Output:
1 2 2 2 3 4 4 4 5 5 5

6. Sachin loved Gully Boy and he is now inspired to become a rapper. His friend Saurav gives him a string and asks him to rap it in his own way. Sachin, after some time, figures out that if there is a substring in that string which reads same even backwards sounds amazing. In order to impress the crowd, he decided to find the longest substring possible that contains half of the characters in proper alphabetical order. In other words, if the desired substring is of size ***2k***, then the first ***k*** (***k*** > 1, i.e., the substring cannot be of size 2) characters are in proper alphabetical order. If there are two substrings possible with the same longest length, then print the one which comes first in alphabetical order. If no such substring can be found, print -1.

Input:
String s

Output:
The desired longest substring.

Example:
Input:
aaaabaaa

Output:
aaabaaa

Input:
abdccdba

Output:
-1 (Though the string is the same even when read backwards, but 'd' appears before 'c' in first half of the string. Note that 'cc' is not considered as the required substring.)

Input:
adsarttriopiuxyyxx

Output:
rttr (Here, both rttr and xyyx read the same when read backwards. However, rttr precedes xyyx alphabetically.)

7. Sharat is currently at a car rental service, and he wants to reach a cinema hall. The film he has bought a ticket for starts in ***t*** minutes. There is a straight road of length ***s*** from the car rental service to the cinema. Let's introduce a coordinate system so that the car rental service is at the point 0, and the cinema is at the point ***s***. There are ***k*** gas stations along the road, and at each of them, you can fill a car with any amount of fuel for free! Consider that this operation doesn't take any time, i.e. can be carried out instantly. There are ***n*** cars in the rental service, ***i***-th of them is characterized with two integers c_i and v_i — the price of renting this car and the capacity of its fuel tank in litres. It is not possible to fuel a car with more fuel than its tank capacity v_i. All cars are have fuel filled to the full tank capacity at the car rental service. Each of the cars can be driven in one of two speed modes: normal or accelerated. In the normal mode, a car covers 1 kilometre in 2 minutes, and consumes 1 litre of fuel. In the accelerated mode, a car covers 1 kilometre in 1 minute, but consumes 2 litres of fuel. The driving mode can be changed at any moment and any number of times. Your task is to choose a car with minimum price such that Sharat can reach the cinema before the show starts, i.e. not later than in ***t*** minutes. Assume that all cars are completely fuelled initially.

Input:
The first line contains four positive integers ***n***, ***k***, ***s*** and ***t*** — the number of cars at the car rental service, the number of gas stations along the road, the length of the road and the time in which the film starts.
Each of the next ***n*** lines contains two positive integers c_i and v_i— the price of the ***i***-th car and its fuel tank capacity.
The next line contains ***k*** distinct integers $g_1, g_2, \ldots, g_k$— the positions of the gas stations on the road in arbitrary order.

Output:
Print the minimum rent price of an appropriate car, i.e. a car so that Sharat will be able to reach the cinema before the film starts (not later than ***t*** minutes). If there is no appropriate car, print -1.

Example:
Input:
3 1 8 10
10 8
5 7
11 9
3

Output:
10

8. Captain America hasn't given up the hope and is now looking for some ways to get back the infinity stones from Thanos. He goes to Titan to search for him but instead is stuck in a 2D string grid. After sending a distress signal to Captain Marvel, he receives a word. If that word exists in the grid, then Captain can come out otherwise not. The word can be constructed from horizontally or vertically neighbouring letters. Refer to sample cases for more clarity.

Note - The cell itself does not count as an adjacent cell. The same letter cell may be used more than once.

Input:
N (Number of strings) ***M*** (length of each strings)
Each of the next ***N*** lines would be strings with ***M*** length
T (Number of test cases)
Each of the next ***T*** lines would be words received from Captain marvel

Example:
Input:
3 4
ABCE
SFCS
ADEE
5
ABCCED
SEE
ABCB
ABFSAB
ABCD

Output:
Yes
Yes
Yes
Yes
No

9. Imagine you have a special keyboard with the following keys:
Key 1: Prints 'A' on screen
Key 2: (Ctrl-A): Select screen

Key 3: (Ctrl-C): Copy selection to buffer
Key 4: (Ctrl-V): Print buffer on screen appending it after what has already been printed.

Suppose you can only press the above mentioned four keys of the keyboard at most ***N*** times. Write a program to produce maximum numbers of A's with ***N*** keystrokes. That is to say, the input parameter is ***N*** (no. of times you can press the keys), and the output is ***M*** (no. of A's that you can produce).

Input:
The first line is ***N*** where N is the number of times key is pressed.

Output:
Print maximum number of As. Print -1, if ***N*** is greater than 75.

Constraints:
$1 \leq N \leq 75$

Example:
Input:
3

Output:
3

Explanation: Type 3 As by pressing Key 1 3 times.

Input:
7

Output:
9

Explanation:
Key 1 - print A, buffer - A
Key 2 - print A, buffer - AA
Key 3 - print A, buffer - AAA
Key 4 - select, buffer - AAA
Key 1 - copy, buffer - AAA
Key 1 - paste, buffer - AAAAAA
Key 1 - paste, buffer - AAAAAAAAA

****