'''

As the summer season is started, the people in Hyderbad will suffer due to

water crisis. To help the citizen of Hyderabad, GHMC planned to supply the

water using water tanks. As a pilot project, GHMC selected R routes and

covering L locations.

To serve a water tank request by the citizen, the tank has to take

the planned routes by GHMC, Each Route[i] => ['src','des','amt'], start

from a source 'src' to a destination 'des' and the amount to supply

water'amt' between src and dest.

Now help the citizen of hyderabd to find a best route to order a water tank

with a best deal in such a way that, if you are given all the locations and

routes information, combined with starting location as source ‘LOC1’ and

the destination ‘LOC2’.

Your task is to find the best deal to supply the water tanks from source to

destination with at most ‘H’ halts. If there is no desired route found to

supply water tank, print -1.

NOTE: if there are L locations, the locations are indexed as: 0,1,2,..,L-1.

Input Format:

-------------

Line-1: Two integers L and R, L number of locations, and R routes.

Next R lines: 3 space separated integers, src, des, amt per water tank.

Next line: 3 space separated integers, LOC1, LOC2, H.

Output Format:

--------------

Print an integer, best deal to get a water tank.

Sample Input-1:

---------------

3 3

0 1 50

1 2 50

0 2 250

0 2 1

Sample Output-1:

----------------

100

Explnation:

-------------

source location is '0' and destination is '2', and no.of halts are 1.

Option-1 : you can start at location-0 to location-1, 1 halt at location-1

and the start from location-1 to location-2,

amount per water tank is 100

Option-2 : you can start at location-0 to location-2 directly,

amount per water tank is 250, So best deal is 100.

Sample Input-2:

---------------

3 3

0 1 50

1 2 50

0 2 250

0 2 0

Sample Output-2:

----------------

250

Explnation:

-------------

source location is '0' and destination is '2', and no halts.

So, you have to start at location-0 to location-2 directly,

amount per water tank is 250. So best deal is 250.

'''

l,r=list(map(int,input().split()))

lr=[]

graph=[]

for i in range(r):

graph.append(list(map(int,input().split())))

req=list(map(int,input().split()))

g=[[] for i in range(l)]

for i in graph:

g[i[0]].append([i[1],i[2]])

s=[]

s.append([0,0,0])

vis=[False for i in range(l)]

p=1e9

while s:

src,h,c=s.pop(0)

if(src==req[1] and h<=req[2]+1):

p=min(p,c)

continue

vis[src]=True

for i in g[src]:

if(vis[i[0]]==False):

s.append([i[0],h+1,c+i[1]])

if(p==1e9):

print(-1)

else:

    print(p)

The integer array arr[] is given and print the resultant value of the bitwise OR

of the sum of all the possible sub-arrays of the given array arr[].

Note: A sub-array is a sequence of array-elements that could be drawn from other

sequence of array-elements by removing zero or more elements without changing

the order of the remaining elements.

Input Format

-------------

Line-1: Read the size of array P.

Line-2: Read the array elements(space separated values).

Output Format

--------------

Print the resultant value.

Sample input-1:

---------------

4

0 2 1 3

Sample output-1:

----------------

7

Explanation

The sums of sub-arrays are

0 1 2 3 4 5 6.

and bitwise OR operation

0 OR 1 OR 2 OR 3 OR 4 OR 5 OR 6 = 7.

Sample input-2:

---------------

4

0 0 0 0

Sample output-2:

----------------

0

Explanation

Only one sum sub-arrays i.e. 0.

import java.util.\*;

public class Main{

public static void main(String[] args){

Scanner sc=new Scanner(System.in);

int n=sc.nextInt();

int[] arr=new int[n];

for(int i=0;i<n;i++){

arr[i]=sc.nextInt();

}

int res=0;

int sum=0;

for(int i=0;i<n;i++){

sum=0;

for(int j=i;j<n;j++){

sum+=arr[j];

res|=sum;

}

}

System.out.println(res);

    }

}

A Kid built a structure using building blocks,

by placing the building-blocks adjacent to each other.

A building-block is a vertical alignment of blocks.

\_\_\_

here one block each represents as |\_\_\_|.

The following structure made up of using building blocks

\_\_\_

\_\_\_|\_\_\_| \_\_\_

|\_\_\_|\_\_\_|\_w\_|\_\_\_|\_\_\_ \_\_\_

\_\_\_|\_\_\_|\_\_\_|\_\_\_|\_\_\_|\_\_\_| w \_w\_ w |\_\_\_|

\_\_\_|\_\_\_|\_\_\_|\_\_\_|\_\_\_|\_\_\_|\_\_\_|\_w\_\_|\_\_\_|\_w\_|\_\_\_|\_\_\_\_\_\_\_\_\_\_\_\_

0 1 3 4 2 3 2 0 1 0 2

Once the structure is completed, kid pour water(w) on it.

You are given a list of integers, heights of each building-block in a row.

Now your task How much amount of water can be stored by the structure.

Input Format:

-------------

Space separated integers, heights of the blocks in the structure.

Output Format:

--------------

Print an integer,

Sample Input:

-------------

0 1 3 4 2 3 2 0 1 0 2

Sample Output:

--------------

6

Explanation:

-----------

In the above structure, 6 units of water (w represents the water in the structure)

can be stored.

import java.util.\*;

public class Main{

public static void main(String[] args){

Scanner sc=new Scanner(System.in);

String s=sc.nextLine();

String[] ss=s.split(" ");

int[] arr=new int[ss.length];

for (int i=0;i<arr.length;i++){

arr[i]=Integer.parseInt(ss[i]);

}

// System.out.println(Arrays.toString(arr));

int capacity=0;

for(int i=1;i<arr.length-1;i++){

int maxl=arr[i];

for(int j=0;j<=i;j++){

maxl=Math.max(maxl,arr[j]);

}

int maxr=arr[i];

for(int j=i;j<arr.length;j++){

maxr=Math.max(maxr,arr[j]);

}

capacity+=Math.min(maxl,maxr)-arr[i];

}

System.out.println(capacity);

}

}

In a school in Telangana students like to play with strings by splitting them.

If they are given two strings s1 and s2 of the same length.

They split both strings at the same choosen index , splitting s1 into

two strings: prefix and suffix where s1 = prefix + suffix, and splitting s2

into two strings: prefix1 and suffix1 where s2 = prefix1 + suffix1.

Check if prefix + suffix1 or prefix1 + suffix forms a palindrome.

For example, if s = "xyz", then "" + "xyz", "x" + "yz", "xy" + "z" ,

and "xyz" + "" are valid splits.

Return true if it is possible to form a palindrome string,

otherwise return false.

Notice that x + y denotes the concatenation of strings x and y.

Example 1:

----------

Input: s1 = "a", s2 = "b"

Output: true

Explaination: If either s1 or s2 are palindromes the answer is true

since you can split in the following way:

s1prefix = "", s1suffix = "a"

s2prefix = "", s2suffix = "b"

Then, s1prefix + s2suffix = "" + "b" = "b", which is a palindrome.

Example 2:

-----------

Input: a = "zdfgh", b = "zgecd"

Output: false

Example 3:

----------

Input: s1 = "vmbdge", s2 = "kjabmv"

Output: true

Explaination: Split them at index 3:

prefix = "vmb", suffix = "dge"

prefix1 = "kja", suffix1 = "bmv"

Then, prefix + suffix1 = "vmb" + "bmv" = "vmbbmv", which is a palindrom

Sample Input-1:

---------------

a b----> two strings

Sample Output-1:

----------------

true

Sample Input-2:

---------------

vmbdge kjabmv ------->two strings

Sample Output-2:

----------------

True

import java.util.\*;

public class Main{

public static void main(String[] args){

Scanner sc=new Scanner(System.in);

String s1=sc.next();

String s2=sc.next();

boolean flag=false;

for(int i=0;i<s1.length();i++){

if(palin(s1.substring(0,i)+s2.substring(i,s1.length()))){

flag=true;

// System.out.println(true);

// return;

}

if(palin(s2.substring(0,i)+s1.substring(i,s2.length()))){

flag=true;

// System.out.println(true);

// return;

}

}

System.out.println(flag);

}

public static boolean palin(String s){

int i=0;

int j=s.length()-1;

while(i<j){

if(s.charAt(i)!=s.charAt(j)){

return false;

}

i+=1;

j-=1;

}

return true;

}

}