'''

Murali playing a mobile game, Blast the letters.

In the game he is given a word W and value R.

Murali has to perform the blasting operation as follows:

- He has to blast the mimeograph M of length R in W,

a mimeograph is a string such that each letter in it should be same.

- After blasting the mimeograph, the rest of the string on its

left side and right side, concatenated together.

Murali has to perform the blasting operation repeatedly,

until no more blasting is possible. Your task is to return

the final string after all the blast operations have been done.

Input Format:

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

Line-1: A string and an integer, W and R.

Output Format:

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

Print a string, the final string after all the blast operations have been done.

Sample Input-1:

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

ababbaaab 3

Sample Output-1:

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

aba

Sample Input-2:

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

caaabbbaacdddd 2

Sample Output-2:

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

cabc

'''

s,n=list(input().split(" "))

n=int(n)

i=0

j=1

while(i<len(s) and j<len(s)):

if(s[i]==s[j]):

count=1

jj=j

while(jj<len(s) and s[i]==s[jj]):

count+=1

jj+=1

if(count>=n):

s=s[0:i]+s[i+n:len(s)]

i=0

j=1

continue

# print(s)

i+=1

j+=1

print(s)

alternate method to solve the above problem in o(n) time complexity is by using a stack ,u can use a list of tuples which stores characters and their frequency and can perform push , pop and update frequency operations

There are N number of small cartoons and each cartoon will have a balloon of

some color identified by a number. You are asked to select fixed number of

P-cartoons as a subset every time and find the number of distinct number of

colors of balloons in the number of cartoons "P" you pick everytime.

The resultant cartoons ans will be N-P+1 and return it.

The number of distinct number of balloons is:

nums[i:i+p-1] = [nums[i], nums[i+1], ..., nums[i+p-1]].

Input Format:

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

Line-1: Two space separated integers, N and P

Line-2: N space separated integers, nums[]

Output Format:

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

Print the (N-P+1) integers as output.

Sample Input-1:

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

7 4

1 2 2 3 3 4 4

Sample Output-1:

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

3 2 3 2

Explanation:

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

The number of distinct elements in each subarray of size P goes as follows:

- nums[0:3] = [1,2,2,3] so ans[0] = 3

- nums[1:4] = [2,2,3,3] so ans[1] = 2

- nums[2:5] = [2,3,3,4] so ans[2] = 3

- nums[3:6] = [3,3,4,4] so ans[3] = 2

Sample Input-2:

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

6 3

1 1 1 1 1 1

Sample Output-2:

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

1 1 1 1

Explanation:

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

The number of distinct elements in each subarray goes as follows:

- nums[0:2] = [1,1,1] so ans[0] = 1

- nums[1:3] = [1,1,1] so ans[1] = 1

- nums[2:4] = [1,1,1] so ans[2] = 1

- nums[3:5] = [1,1,1] so ans[3] = 1

Sample Input-3:

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

7 3

1 2 3 4 2 1 3

Sample Output-3:

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

3 3 3 3 3

Using sliding window

import java.util.\*;

public class Main{

public static void main(String[] args){

Scanner sc=new Scanner(System.in);

int n=sc.nextInt();

int m=sc.nextInt();

int[] arr=new int[n];

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

arr[i]=sc.nextInt();

}

HashSet<Integer> hs=new HashSet<>();

for(int i=0;i<n-m+1;i++){

hs.clear();

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

hs.add(arr[j]);

}

System.out.print(hs.size()+" ");

}

}

}

To do the problem in o(n) time complexty u have to use sliding window algorithm along with the hashmap

Budget Padmanabham planned to visit the tourist places. There are N tourist

places in the city. The tourist places are numbered from 1 to N.

You are given the routes[] to travel between the tourist places in the city.

where routes[i]=[place-1, place-2, price], A route is a bi-directional route.

You can travel from place-1 to place-2 or place-2 to place-1 with the given price.

Your task is to help Budget Padmanabham to find the cheapest route plan, to

visit all the tourist places in the city. If you are not able to find such plan,

print -1.

Input Format:

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

Line-1: Two space separated integers N and R,number of places and routes.

Next R lines: Three space separated integers, start, end and price.

Output Format:

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

Print an integer, minimum cost to visit all the tourist places.

Sample Input-1:

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

4 5

1 2 3

1 3 5

2 3 4

3 4 1

2 4 5

Sample Output-1:

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

8

Explanation:

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

The cheapest route plan is as follows:

1-2, 2-3, 3-4 and cost is 3 + 4 + 1 = 8

Sample Input-2:

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

4 3

1 2 3

1 3 5

2 3 4

Sample Output-2:

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

-1

import java.util.\*;

public class Main{

static int[] parents;

public static void main(String[] args){

Scanner sc=new Scanner(System.in);

int n=sc.nextInt();

int m=sc.nextInt();

int[][] arr=new int[m][3];

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

arr[i][0]=sc.nextInt()-1;

arr[i][1]=sc.nextInt()-1;

arr[i][2]=sc.nextInt();

}

Arrays.sort(arr,new Comparator<int[]>(){

public int compare(int[] a,int[] b){

return a[2]-b[2];

}

});

int ans=0;

parents=new int[n];

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

parents[i]=i;

}

for(int[] a:arr){

if(find(a[0])!=find(a[1])){

union(a[0],a[1]);

ans+=a[2];

}

}

int x=parents[0];

int flag=1;

for(int i: parents){

if(i!=x){

System.out.println(-1);

flag=0;

break;

}

}

if(flag==1){

System.out.println(ans);

}

}

public static int find(int a){

if(parents[a]==a){

return a;

}

return parents[a]=find(parents[a]);

}

public static void union(int i,int j){

int a=find(i);

int b=find(j);

parents[a]=b;

}

}

There is a competetion conducted at a school where students are given string

of English letters. String consits of lower case letters.

A student can alter the string and create a new string so that, when reversed,

the altered and and its reveresed strings should be equal, designating a winner.

There is a constraint that each string can be altered atmost twice.

In one operation, you can change any character of string to any other character.

The student has to return "true" if he is winner, otherwise return "false"

Input Format:

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

A string, S

Output Format:

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

Print a boolean result.

Sample Input-1:

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

xyzwyx

Sample Output-1:

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

true

Explanation:

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

One way to alter the string such that it matches the given criteria,

using 1 operation is:

- Change s[2] to 'w'. Now, s = "xywwyx".

One operation could be performed to make our required string so return true.

Sample Input-2:

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

pp

Sample Output-2:

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

true

Explanation:

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

One way to alter the string such that it matches the given criteria,

using 2 operations is:

- Change s[0] to 'q'. Now, s = "qp".

- Change s[1] to 'q'. Now, s = "qq".

Two operations could be performed to make our required string so return true.

Sample Input-3:

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

mnopqr

Sample Output-3:

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

false

Explanation:

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

It is not possible to make string that matches the given criteria using

one or two operations. So, return false.

import java.util.\*;

public class Main{

public static void main(String[] args){

Scanner sc=new Scanner(System.in);

String s=sc.next();

int i=0;

int j=s.length()-1;

int count=0;

while(i<=j){

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

count+=1;

}

i+=1;

j-=1;

}

if(count<=2){

System.out.println(true);

}

else{

System.out.println(false);

}

}

}