'''

A number is called self-supportive if all the digits of the number are factors

of the number. For example, 48 is a self-supportive number because 48 % 4 == 0,

and 48 % 8 == 0.

A number is not a self-supportive if it has any digit as zero.

Given two Positive numbers start and end, return a set of all

the self-supportive numbers in between start and end (both inclusive).

1<=start<=end<=10^4

Input Format:

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

Line: 2 space seperated integers start and end.

Output Format:

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

Print a space seperated list.

Sample Input-1:

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

20 25

Sample Output-1:

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

22 24

Explanation:

-----------

20 has 0 as digit so it's not self - supportive.

21 is not divisible by 2. so it's not self - supportive.

22 is divisible 2. so it's self - supportive.

23 is not divisible by both the digits 2 and 3. so it's not self - supportive.

24 is divisible by both 2 and 4. so it is self - supportive.

So 22 and 24 are self-supportive.

Sample Input-2:

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

50 80

Sample Output-2:

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

55 66 77

Write your python code below

'''

n1,n2=list(map(int,input().split()))

l=[]

def fun(n):

k=n

r=0

while(n>0):

r=n%10

if(r==0 or k%r!=0):

return False

n=n//10

return True

for i in range(n1,n2):

if(fun(i)):

l.append(i)

print(\*l)

Vihan is given a number N and He wants to check whether N is a converse number

or not. The binary form of number N is said to be converse number, if it obeys

the following property: "every pair of adjacent digits are different".

Your task is to help Vihan to find N is a converse number or not.

If yes, print 'true', otherwise print 'false'.

Input Format:

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

An integer N, the positive number.

Output Format:

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

Print a boolean result.

Sample Input-1:

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

85

Sample Output-1:

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

true

Explanation:

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

Binary Rep of 85 is 1010101

Sample Input-2:

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

87

Sample Output-2:

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

false

Explanation:

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

Binary Rep of 87 is 1010111

import java.util.\*;

public class Main{

public static void main(String[] args){

Scanner sc=new Scanner(System.in);

int n=sc.nextInt();

String s=Integer.toBinaryString(n);

boolean flag=true;

for(int i=0;i<s.length()-1;i++){

if(s.charAt(i)==s.charAt(i+1)){

flag=false;

}

}

System.out.println(flag);

}

}

Given a classic mobile phone, and the key pad of the phone looks like below.

1 2 3

abc def

4 5 6

ghi jkl mno

7 8 9

pqrs tuv wxyz

\* 0 #

You are given a string S contains digits between [2-9] only,

For example: S = "2", then the possible words are "a", "b", "c".

Now your task is to find all possible words that the string S could represent.

and print them in a lexicographical order.

Input Format:

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

A string S, consist of digits [2-9]

Output Format:

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

Print the list of words in lexicographical order.

Sample Input-1:

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

2

Sample Output-1:

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

[a, b, c]

Sample Input-2:

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

24

Sample Output-2:

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

[ag, ah, ai, bg, bh, bi, cg, ch, ci]

import java.util.\*;

public class Main{

public static void main(String[] args){

Scanner sc=new Scanner(System.in);

String s=sc.next();

char[] arr=s.toCharArray();

HashMap<Character,String> h=new HashMap<>();

h.put('1',"");

h.put('2',"abc");

h.put('3',"def");

h.put('4',"ghi");

h.put('5',"jkl");

h.put('6',"mno");

h.put('7',"pqrs");

h.put('8',"tuv");

h.put('9',"wxyz");

h.put('\*',"");

h.put('0',"");

h.put('#',"");

ArrayList<String> l=new ArrayList<>();

System.out.println(fun(arr,h,l,arr.length-1));

}

public static ArrayList<String> fun(char[] arr,HashMap<Character,String> h,ArrayList<String> l,int n){

if(n==0){

for(int i=0;i<h.get(arr[n]).length();i++){

l.add(String.valueOf(h.get(arr[n]).charAt(i)));

}

// return new ArrayList<String>(Arrays.asList(h.get(arr[n])));

return l;

}

l=fun(arr,h,l,n-1);

ArrayList<String> lk=new ArrayList<>();

for(int i=0;i<l.size();i++){

for(int j=0;j<h.get(arr[n]).length();j++){

String k=String.valueOf(l.get(i)+h.get(arr[n]).charAt(j));

lk.add(k);

}

}

return lk;

}

}

