'''

Arjun is playing a word game in his tab. When he start the game , It displays a

word and he has to check for any two adjacent characters which are same.If found

he has to select those two characters by tapping, so that both characters will

be deleted from the word and the word size shrinks by 2. This process to be

repeated until the word has no two adjacent characters are same.

You task is to help Arjun to perform the above task and Return the final word

obtained.

Input Format:

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

Line-1: A string represents the word.

Output Format:

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

Return a string or empty string.

Constraints:

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

1 <= word.length <= 10^5

word consists only lower case letters.

Sample Input-1:

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

pqqprs

Sample Output-1:

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

rs

Explanation:

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

Initially , Delete two 'q's then the word will be 'pprs'. Now Delete two 'p's

then the word will be rs.

Sample Input-2:

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

pqqqprrs

Sample Output-2:

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

pqps

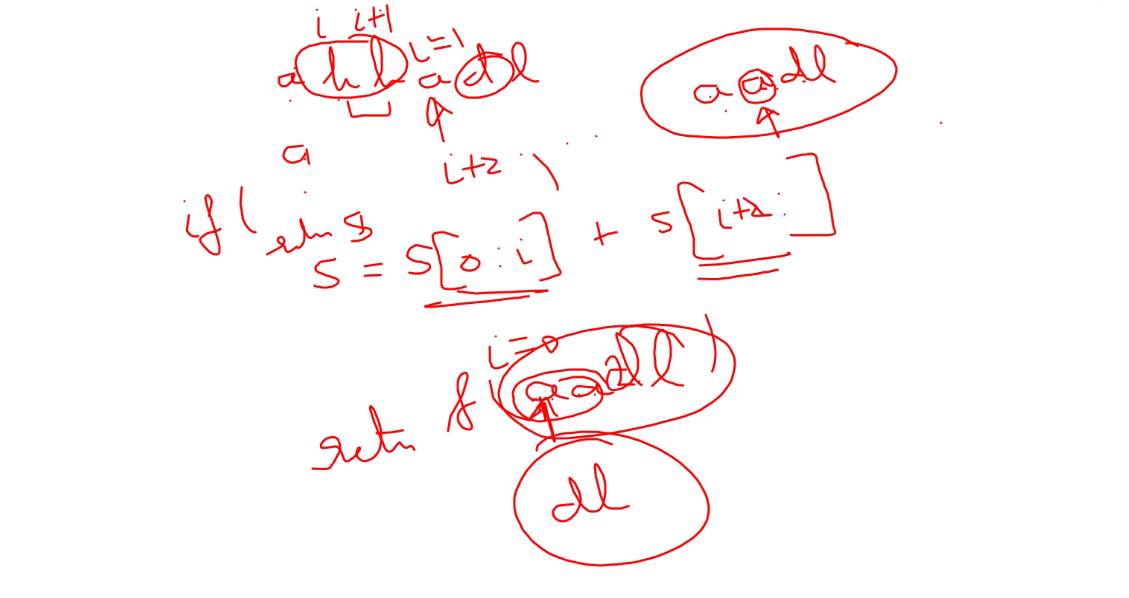
Explanation:

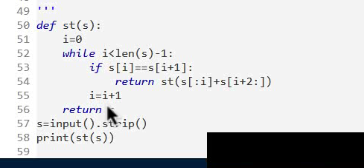
-------------

Initially , Delete two 'q's then the word will be 'pqprrs'. Now Delete two 'r's

then the word will be pqps.

'''





Mr Black Noir is working with Numbers. He is given a number N.

He wants to findout how many times each digit appeared in N.

Your task is to help Mr Black Noir, in computing the desired result.

Input Format:

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

An integer N

Output Format:

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

Print a boolean value.

Sample Input:

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

92233720368547758

Sample Output:

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

0:1

1:0

2:3

3:3

4:1

5:2

6:1

7:3

8:2

9:1

My soln

import java.util.\*;

public class Main{

public static void main(String[] args){

Scanner sc=new Scanner(System.in);

long n=sc.nextLong();

int[] arr=new int[10];

int r=0;

while(n>0){

r=(int)(n%10);

arr[r]=arr[r]+1;

n=n/10;

}

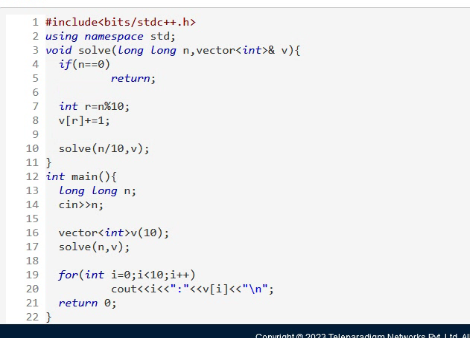
for(int i=0;i<10;i++){

System.out.println(i+":"+arr[i]);

}

}

}



You are given a rule to build a special sequence as follows:

The rule is:

seq(0) = 0

seq(n) = seq(n-1) - n, if seq(n-1) - n > 0 and is not already present in

the sequence

= seq(n-1) + n, otherwise.

In this sequence, seq(n) is (n+1)th term. as we are starting seq(0) as 1st term.

You are given an integer N, you need to find the sequence upto seq(n).

Input Format:

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

An integer N

Output Format:

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

Print the sequence.

Sample Input-1:

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

5

Sample Output-1:

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

0 1 3 6 2

Sample Input-2:

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

8

Sample Output-2:

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

0 1 3 6 2 7 13 20

My soln:

import java.util.\*;

public class Main{

public static ArrayList<Integer> seq(int n1,int n2,ArrayList<Integer> k){

if(n1==n2){

return k;

}

if(n1==0){

k.add(0);

return seq(n1+1,n2,k);

}

int num1=k.get(n1-1)-n1;

if(num1>0 && !k.contains(num1)){

k.add(num1);

// return seq(n1+1,n2,k);

}

else {

num1=k.get(n1-1)+n1;

k.add(num1);

// return seq(n1+1,n2,k);

}

return seq(n1+1,n2,k);

}

public static void main(String[] args){

Scanner sc=new Scanner(System.in);

int n=sc.nextInt();

System.out.println(seq(0,n,new ArrayList<Integer>()));

}

}