There are some cubes, and each cube has an alphabet (from A to Z)

printed on it. You can construct different, non-empty words using these

cubes and each of the word length should be 0 < length <= number of cubes.

You are given a string of alphabets S,

Your task is to findout number of possible non-empty distinct words

Input Format:

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

A string S, consist of A-Z letters only.

Output Format:

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

Print an integer, number of possible non-empty distinct words.

Sample Input-1:

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

EGG

Sample Output-1:

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

8

Explanation:

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

The possible distinct words are "E", "G", "EG", "GG", "GE", "EGG", "GEG", "GGE".

Sample Input-2:

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

MADAM

Sample Output-2:

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

89

import java.util.\*;

public class Main{

static Set<String> hs=new HashSet<>();

static Set<String> ss=new HashSet<>();

public static void main(String[] args){

Scanner sc=new Scanner(System.in);

String s=sc.next();

fun("",s);

// System.out.println(hs.size());

for(String i : ss){

if(!i.equals("")){

fun2("",i);

}

}

// System.out.println(fun("",s));

System.out.println(hs.size());

}

public static void fun(String p,String up){

if(up.isEmpty()){

// System.out.println(p);

ss.add(p);

return;

}

fun(p+up.charAt(0),up.substring(1));

fun(p,up.substring(1));

// return a+b;

}

public static void fun2(String p,String up){

if(up.isEmpty()){

// System.out.println(p);

hs.add(p);

return;

}

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

fun2(p.substring(0,i)+up.charAt(0)+p.substring(i,p.length()),up.substring(1));

}

}

}

Ganesh is working on numbers, He is given a list of integers 1,2,3,...,N

and the list is indexed from 1 to N.

Now he can shuffle the list in whatever way he want. Shuffled list is said

to be valid, if one of the following is true for i-th position in the list.

- The integer at the i-th position is divisible by i.

- 'i' is divisible by the integer at the i-th position.

Your task is to find out, How many valid shuffles can Ganesh do?

Input Format:

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

An integer N, where N<=20.

Output Format:

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

Print an integer, number of ways ganesh can shuffle.

Sample Input-1:

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

1

Sample Output-1:

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

1

Explanation:

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

The only shuffled list is [1]

Sample Input-2:

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

2

Sample Output-2:

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

2

Explanation:

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

The first shuffled list is [1, 2]:

Integer at the 1st position (i=1) is 1, and 1 is divisible by i (i=1).

Integer at the 2nd position (i=2) is 2, and 2 is divisible by i (i=2).

The second shuffled list is [2, 1]:

Integer at the 1st position (i=1) is 2, and 2 is divisible by i (i=1).

Integer at the 2nd position (i=2) is 1, and i (i=2) is divisible by 1.

Sample Input-3:

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

3

Sample Output-3:

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

3

Explanation:

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

The first shuffled list is [1, 2, 3]:

Integer at the 1st position (i=1) is 1, and 1 is divisible by i (i=1).

Integer at the 2nd position (i=2) is 2, and 2 is divisible by i (i=2).

Integer at the 3rd position (i=3) is 3, and 3 is divisible by i (i=3).

The second shuffled list is [3, 2, 1]:

Integer at the 1st position (i=1) is 3, and 3 is divisible by i (i=1).

Integer at the 2nd position (i=2) is 2, and 2 is divisible by i (i=2).

Integer at the 3rd position (i=3) is 1, and i is divisible by 1 (i=3).

The third shuffled list is [2, 1, 3]:

Integer at the 1st position (i=1) is 2, and 2 is divisible by i (i=1).

Integer at the 2nd position (i=2) is 1, and i is divisible by 1 (i=2).

Integer at the 3rd position (i=3) is 3, and 3 is divisible by i (i=3).

**The soln which I have commented is correct but it is becoming tle and mle so try to use the other solution**

import java.util.\*;

public class Main{

static int count=0;

public static void main(String[] args){

Scanner sc=new Scanner(System.in);

int n=sc.nextInt();

StringBuilder up=new StringBuilder("");

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

up.append(String.valueOf(i));

}

// fun("",up.toString());

fun2(new ArrayList<Integer>(),n,0);

System.out.println(count);

}

public static void fun2(ArrayList<Integer> l,int n,int i){

if(i==n){

count+=1;

return;

}

for(int j=1;j<=n;j++){

if((j%(i+1)==0 || (i+1)%j==0) && !l.contains(j)){

l.add(j);

fun2(l,n,i+1);

l.remove(i);

}

}

}

// public static void fun(String p,String up){

// if(up.isEmpty()){

// for(int i=1;i<=p.length();i++){

// if((p.charAt(i-1)-'0')%i!=0 && i%(p.charAt(i-1)-'0')!=0){

// return;

// }

// }

// count+=1;

// return;

// }

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

// fun(p.substring(0,i)+up.charAt(0)+p.substring(i,p.length()),up.substring(1));

// }

// }

}