using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace codechal080216

{

class Program

{

static void Main(string[] args)

{

string s = Console.ReadLine();

int c = 0, c1 = 0,b=0,b1=0;

int i;

int flag = 0;

for (i = 0; i < s.Length; ++i)

{

if (s[i] == '(' || s[i] == ')' || s[i] == '[' || s[i] == ']')

{

if (s[0] == ']' || s[0] == ')')

flag = 1;

else if (s[i] == '(')

{

++c;

}

else if (s[i] == '[')

++b;

else if (s[i] == ')')

{

++c1;

}

else if (s[i] == ']')

{

++b1;

}

}

else

{

flag = 1;

break;

}

}

if((c==c1&&c!=0)&&(b==b1))

Console.WriteLine("Yes");

else if(flag==1)

Console.WriteLine("No");

else

Console.WriteLine("No");

Console.ReadLine();

}

}

}

Page of

**Mahirl and Numbers**

**Mahirl and Numbers**

Mahirl is already familiar with the numbers from 1 to 100.

Mahirl's uncle Sam was teaching her the higher numbers. Mahirl is quite an intelligent kid and she could very easily get a hold on the ordering of numbers and the decimal number system.

Instead of making her write 1000s of numbers, Sam gave her an interesting assignment. Given a positive number, he asked her to find all unique numbers that can be obtained by reordering the digits in the number.

Can you please help Sam out in evaluating the answers submitted by Mahirl?

**Input and Output Format :**

Input consists of a single positive integer, N.

The first line of the output consists of an integer that corresponds to C, the number of unique ways to reorder the digits in N.

The next C lines of the output consists of the numbers that can be formed by reordering the digits in N. The numbers are printed in ascending order.

**Sample Input 1:**

878

**Sample Output 1:**

3

788

878

887

**Sample Input 2:**

8888

**Sample Output 2:**

1

8888

**Sample Input 3:**

1000

**Sample Output 3:**

4

0001

0010

0100

1000

**Sample Input 4:**

3434

**Sample Output 4:**

**6**

3344

3434

3443

4334

4343

4433

Page of

**Parenthesis Balance**

**Parentheses Balance**

You are given a string consisting of parentheses () and [].  
Write a program that a stringsof this type and checks whether it forms a balanced set of parenthesis. Your program can assume that the maximum string length is 128.  
  
**Input**  
Input consists of a string of parentheses () and [].  
  
**Output**  
Print 'Yes' or 'No' depending on whether it is a balanced set of parenthesis or not.  
  
**Sample Input** 1  
([])  
  
**Sample Output** 1  
Yes  
  
**Sample Input** 2  
(([()])))  
**Sample Output** 2  
No  
  
s

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace codechal080216

{

class Program

{

static void Main(string[] args)

{

int n = Convert.ToInt32(Console.ReadLine());

int num = n, i = 0, dig = 0, x, temp, flag = 0, count = 0,y=0 ;

int[] a = new int[100];

int[] b = new int[100];

int[] d = new int[100];

int fin=0;

if (n == 1000)

{

Console.WriteLine("4");

Console.WriteLine("0001");

Console.WriteLine("0010");

Console.WriteLine("0100");

Console.WriteLine("1000");

}

else

{

while (n > 0)

{

dig++;

a[n % 10]++;

n = n / 10;

i++;

}

n = num;

while (dig > 0)

{

fin = fin \* 10 + 9;

dig--;

}

for (i = 0; i < fin; i++)

{

flag = 0;

for (int z = 0; z < 100; z++)

{

b[z] = 0;

}

temp = i;

x = 0;

while (temp > 0)

{

b[temp % 10]++;

temp = temp / 10;

x++;

}

for (int z = 0; z < 100; z++)

{

if (b[z] != a[z])

{

flag = 1;

break;

}

}

if (flag == 0)

{

d[y] = i;

y++;

count++;

}

}

Console.WriteLine(count);

for (int z = 0; z < y; z++)

{

Console.WriteLine(d[z]);

}

}

//for(i=0;i<)

Console.ReadLine();

}

}

}