Problem: Kaprekar Number:

A modified *Kaprekar number* is a positive whole number  with  digits, such that when we split its square into two pieces - a right hand piece  with  digits and a left hand piece  that contains the remaining  or  digits, the sum of the pieces is equal to the original number (i.e.  +  = ).

**Note:** r may have leading zeros.

Here's an explanation from Wikipedia about the **ORIGINAL** Kaprekar Number (spot the difference!): *In mathematics, a Kaprekar number for a given base is a non-negative integer, the representation of whose square in that base can be split into two parts that add up to the original number again. For instance, 45 is a Kaprekar number, because 45² = 2025 and 20+25 = 45.*

**The Task**   
You are given the two positive integers  and , where  is lower than . Write a program to determine how many Kaprekar numbers are there in the range between  and  (both inclusive) and display them all.

**Input Format**

There will be two lines of input: , lowest value , highest value

**Constraints**: 

**Output Format**

Output each Kaprekar number in the given range, space-separated on a single line. If no Kaprekar numbers exist in the given range, print INVALID RANGE.

**Sample Input**

1

100

**Sample Output**

1 9 45 55 99

**Explanation**

, , , , and  are the Kaprekar Numbers in the given range.

Solution:

long determineLength(long testCase) //This function determines the length of the digit

{

long counter=0;

while(testCase>0)

{

counter+=1;

testCase=testCase/10;

}

return counter;

}

int check(long testCase) //This function checks for kaprekar number

{

long left=0, right=0;

long square=testCase\*testCase;

long temp=square;

long length=determineLength(square);

if(length%2==0 ?length+=0 : length=length+1);

long counter=0;

while(counter<(length/2) ) //this function splits the square of the number into right

{ part

left=left+(square%10)\*pow(10,counter);

square=square/10;

counter++;

}

right=(temp-left)/pow(10,length/2); //this generates the left part of the square

if(left+right==testCase && left!=0) //test condition for kaprekar number

{cout<<testCase<<" "; return 1;}

return 0;

}

int main()

{

long range1, range2;

cin>>range1 >>range2;

long counter=0; //the counter is used to determine if the range is valid or not

for(long i=range1; i<=range2; i++)

{

counter+=check(i);

}

if(counter==0) //No kaprekar number found hence range is invalid

{cout<<"INVALID RANGE";}

}

* Anshul Aggarwal