When subtracting integers by hand, you can use a strategy called *regrouping*. (This strategy is also sometimes called *borrowing*.) In this strategy, corresponding digits are subtracted from each other one by one, from right to left, unless the digit of the minuend (the number **being subtracted from**) is smaller than the corresponding digit of the subtrahend (the number **being subtracted**). In this case, you have to borrow 10 from the digit of the minuend immediately to the left of the current one. You can see an example of this method at work [here](https://en.wikipedia.org/wiki/Subtraction" \l "Trade_first).

When you subtract using *regrouping*, all the numbers that are being subtracted (the subtrahend) remain the same, while the numbers that are being subtracted from (the minuend) may change. Given two integers of the same length, return an array containing the numbers that are **being subtracted from** when performing subtraction by hand, from right to left.

**Example**

For minuend = 2024 and subtrahend = 1234, the output should be  
subtractionByRegrouping(minuend, subtrahend) = [4, 12, 9, 1].

When subtracting 1234 from 2024, we actually consider the following pairs (going from right to left):

* 4 and 4;
* 12 (10 was borrowed from 0, temporarily making it equal to -1, and added to 2) and 3;
* 9 (again, 10 was borrowed from 2) and 2;
* 1 and 1.

**Input/Output**

* **[time limit] 3000ms (cs)**
* **[input] integer minuend**

The number that the subtrahend is being subtracted from.

*Constraints:*  
1 ≤ minuend ≤ 109.

* **[input] integer subtrahend**

The number being subtracted from the minuend.

*Constraints:*  
1 ≤ subtrahend ≤ minuend.

subtrahend is guaranteed to have the same number of digits as minuend.

* **[output] array.integer**

An array containing the numbers that are being subtracted from when subtracting the subtrahendfrom the minuend using the *regrouping strategy*.

<https://codefights.com/tournaments/RrqS57vhTAkt9WjYq/C/solutions>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication2

{

class Program

{

static int[] subtractionByRegrouping(int minuend, int subtrahend)

{

int n = minuend.ToString().Length;

int[] ms = new int[n];

ms = Array.ConvertAll(minuend.ToString().ToCharArray(), e => int.Parse(e.ToString()));

int[] ss = new int[n];

ss = Array.ConvertAll(subtrahend.ToString().ToCharArray(), e => int.Parse(e.ToString()));

//List<int> res = new List<int>();

for (int i = n - 1; i >= 0; i--)

{

if (ms[i] < ss[i])

{

int k = i;

if (ms[k] == 0)

{

ms[k] = 10;

k--;

while (k >= 0 && ms[k] == 0)

{

ms[k] = 9;

k--;

}

if (ms[k] > 0)

{

ms[k]--;

}

}

else

{

k = i;

ms[i] = int.Parse("1" + ms[i]);

k--;

while (k >=0 && ms[k] < ss[k] && ms[k] !=0 )

{

ms[k]--;

ms[k] = int.Parse("1" + ms[k]);

k--;

}

while (k >= 0 && ms[k] == 0)

{

ms[k] = 9;

k--;

}

if (k >= 0)

{

ms[k]--;

}

}

}

}

Array.Reverse(ms);

return ms;

}

static void Main(string[] args)

{

//int minuend = 51234;

//int subtrahend = 12345; //[14, 12, 11, 10, 4]

//int minuend = 2024;

//int subtrahend = 1234; //[4, 12, 9, 1]

//int minuend = 5000;

//int subtrahend = 4999; //[10, 9, 9, 4]

//int minuend= 20100;

//int subtrahend = 19199;//[10, 9, 10, 9, 1]

foreach (int elem in subtractionByRegrouping(minuend, subtrahend))

{

Console.Write(elem + " ");

}

Console.ReadLine();

}

}

}