You are given an array of integers that you want distribute between several groups. The first group should contain numbers from 1 to 104, the second should contain those from 104 + 1 to 2 \* 104, ..., the 100thone should contain numbers from 99 \* 104 + 1 to 106and so on.

All the numbers will then be written down in groups to the text file in such a way that:

* the groups go one after another;
* each non-empty group has a header which occupies one line;
* each number in a group occupies one line.

Calculate how many lines the resulting text file will have.

**Example**

For a = [20000, 239, 10001, 999999, 10000, 20566, 29999], the output should be  
numbersGrouping(a) = 11.

The numbers can be divided into 4 groups:

* 239 and 10000 go to the 1st group (1 ... 104);
* 10001 and 20000 go to the second 2nd (104 + 1 ... 2 \* 104);
* 20566 and 29999 go to the 3rd group (2 \* 104 + 1 ... 3 \* 104);
* groups from 4 to 99 are empty;
* 999999 goes to the 100th group (99 \* 104 + 1 ... 106).

Thus, there will be 4 groups (i.e. four headers) and 7numbers, so the file will occupy 4 + 7 = 11 lines.

**Input/Output**

* **[time limit] 3000ms (cs)**
* **[input] array.integer a**

*Constraints:*  
1 ≤ a.length ≤ 105,  
1 ≤ a[i] ≤ 109.

* **[output] integer**

The number of lines needed to store the grouped numbers.

<https://codefights.com/arcade/code-arcade/mirror-lake/kGeuCkJNbqczCCqgg>

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace ConsoleApplication1

{

class Program

{

static int numbersGrouping(int[] a)

{

HashSet<int> hs = new HashSet<int>();

foreach (int elem in a)

{

hs.Add(grupo(elem));

}

return hs.Count + a.Length;

}

static int grupo(int n)

{

//int n = 102382103;

//n = 82103;

if (n <= 10000)

{

return 0;

}

string ns = n.ToString();

string g = ns.Substring(0, ns.Length - 5 + 1);

string resto = ns.Substring(ns.Length - 5, 5);

//Console.WriteLine(resto);

string ultimo = "";

ultimo += resto[0].ToString() + new string('0', resto.Length - 1);

// Console.WriteLine(ultimo);

int g1 = int.Parse(g);

if (resto == ultimo)

{

g1--;

}

//Console.WriteLine(resto);

return g1;

}

static void Main(string[] args)

{

//Console.WriteLine(grupo(10238));

// Console.WriteLine(grupo(102382103));

//int[] a = { 20000, 239, 10001, 999999, 10000, 20566, 29999 };

//a = new int[]{10000, 1};

//a = new int[] { 1, 2, 3, 4, 5, 6, 7, 8 };

//a = new int[] {102382103, 21039898, 39823, 433, 30928398, 40283209, 23234, 342534, 98473483, 498398424, 9384984, 9839239 };

int actual = 0;

for (int i = 1; i <= 1000000; i++)

{

//Console.WriteLine(grupo(i));

int cont = 0;

while (grupo(i) == actual)

{

cont++;

i++;

}

actual = grupo(i);

Console.WriteLine(grupo(i) + " " + cont);

}

//Console.WriteLine(grupo(20000));

//Console.WriteLine(numbersGrouping(a));

//Console.WriteLine(grupo(10000));

Console.ReadLine();

}

}

}

------------------- SOLUCION MAS COMPACTA----------------------------

static int numbersGrouping(int[] a)

{

HashSet<int> hs = new HashSet<int>();

foreach (int n in a)

{

//hs.Add(grupo(elem));

int g1 = 0;

if (n <= 10000)

{

hs.Add(0);

continue;

//return 0;

}

string ns = n.ToString();

string g = ns.Substring(0, ns.Length - 5 + 1);

string resto = ns.Substring(ns.Length - 5, 5);

//Console.WriteLine(resto);

string ultimo = "";

ultimo += resto[0].ToString() + new string('0', resto.Length - 1);

// Console.WriteLine(ultimo);

g1 = int.Parse(g);

if (resto == ultimo)

{

g1--;

}

hs.Add(g1 );

}

return hs.Count + a.Length;

}