Brute force solution for vietnamese math problem

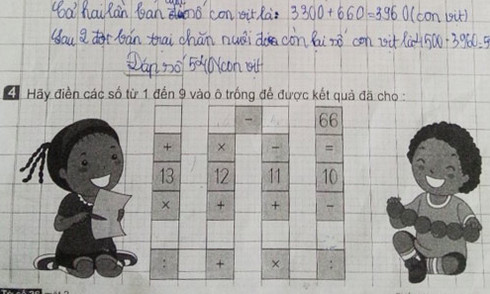
By Jean Paul Ruiz Vallejo

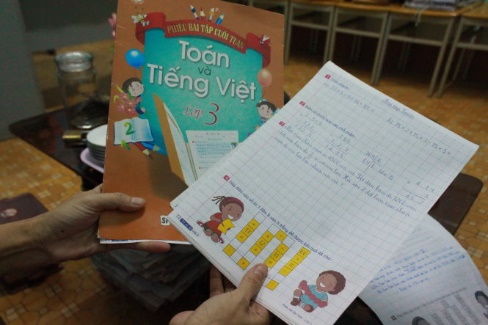
[jpruiz114@gmail.com](mailto:jpruiz114@gmail.com)

Version 1.0

# The problem

In May 18, 2015, a vietnamese website posted an article [1] stating that a math book for 3rd grade had the following problem on it:





The problem also appeared in other websites such as tinmoi.vn [2]

The idea of the problem is to fill the empty squares with the numbers from 1 to 9, non repeating any and using all of them in order to get 66 for result. If you are asking “:” stands for division

The problem generated such a controversy that a local teacher, Dr. Tran Dien Show from the Hanoi National University of Education said that the problem would be a challenge even for a Ph.D not to say for a 3rd grade student [3].

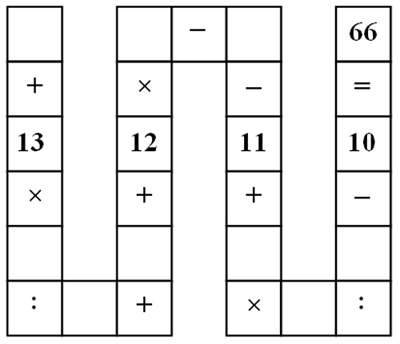
Later, the problem got published on different websites, such as The Guardian [4] and Gizmodo [5].

# Assumptions

* There’s at least one array of values that solve the problem.
* No approximate solutions will be used.

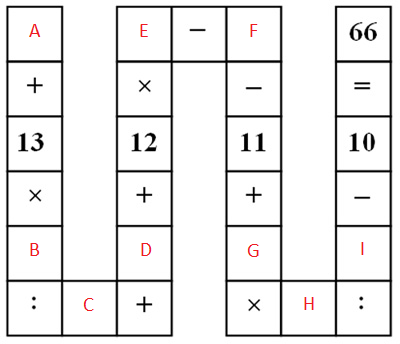
# The solution

Based on the rules of the game, you will have to fill 9 fields with 9 different numbers from 1 to 9, non repeating them.



The first thing we can figure out is that there are 9! possible combinations to solve this. Why?

Consider the variables in red.



A can be any number from 1 to 9.

B can be any number from 1 to 9 but can’t be the same value as A, so it can be 1 of 8.

C can be any number from 1 to 9 but can’t be the same vale as A and B, so it can be 1 of 7.

See where is this going?

D can be 1 of 6.

E can be 1 of 5.

F can be 1 of 4.

G can be 1 of 3.

H can be 1 of 2.

I can be 1.

Knowing this, you will have 9! (9 \* 8 \* 7 \* 6 \* 5 \* 4 \* 3 \* 2 \* 1) or 362.880 possible combinations to evaluate.

## Why brute force and not equations?

You have 9 variables and only 1 equation. The ideal scenario would be to have 9 ecuations for the 9 variables, at least for a linear equation system, but that isn’t the case.

So now we know we have to test with 362.880 possible combinations, but how do we get them?

## First approach

We will be using C++ and Microsoft Excel to solve the problem.

I chose C++ because is faster, the overall speed of C++ applications is greater than that of other languages. C++ will help us get all the possible combinations in a short amount of time.

Microsoft Excel will help us to evaluate the combinations and tell what values solve the problem.

How do we generate the combinations?

Assume for a moment you want to have all the combinations of length 3 for a binay code, 1 and 0. You will have your initial alphabet, 1 & 0. If you combine your alphabet, i.e. your length 1 words with the same length 1 words, you would get the length 2 words, like this:

0 0 00

1 1 01

10

11

Then, if you take your alphabet or length 1 words and combine them with the length 2 words, you would get the length 3 words:

0 00 000

1 01 001

10 010

11 011

100

101

110

111

If you want to get a length n word combination, you need to combine your alphabet or length 1 words with the n – 1 length words, where n has to be equal or higher to 2.

So, if you want to get the length 2 words, combine your alphabet with the 2 – 1 (1) length words.

If you want to get the length 3 words, combine your alphabet with the 3 – 1 (2) length words.

…

If you want to get the length 9 words, combine your alphabet with the 9 – 1 (8) length words.

Notice that we will need to have our alphabet, and the length 2, 3, 4, 5, 6, 7 and 8 words to get the length 9 words.

Something else has to be taken into account here. The words can’t have repeated elements, so, from the length 2 words we will check that we are not adding words with repeated elements.

Our initial C++ solution will look like this:

#include <algorithm>

#include <fstream>

#include <iostream>

#include <string>

#include <sstream>

**using** **namespace** std**;**

bool hasDuplicateCharacters**(**string s**)** **{**

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

**for(**int j **=** i**+**1**;** j **<** s**.**length**();** j**++)** **{**

**if(**s**[**i**]** **==** s**[**j**])** **{**

**return** **true;**

**}**

**}**

**}**

**return** **false;**

**}**

int stringHasUniqueElements**(**string strToCheck**)** **{**

int hasUniqueElements **=** 0**;**

**if(!**hasDuplicateCharacters**(**strToCheck**))** **{**

hasUniqueElements **=** 1**;**

**}**

**return** hasUniqueElements**;**

**}**

int main**(**int argc**,** char**\*\*** argv**)** **{**

ifstream alphabet**(**"1.txt"**);**

string alphabetCurrentLine**;**

int iterationToGet **=** 9**;**

/\* \*\*\*\*\* \*/

int checkUniqueElements **=** 1**;**

int hasUniqueElements = 0;

/\* \*\*\*\*\* \*/

int numberFileToRead = iterationToGet - 1;

stringstream nameOfFileToRead;

nameOfFileToRead << numberFileToRead << ".txt";

cout << "File to read: " << nameOfFileToRead.str() << endl;

string prevIterFileName = nameOfFileToRead.str();

ifstream prevIterFile(prevIterFileName.c\_str());

string prevIterFileCurrentLine;

/\* \*\*\*\*\* \*/

stringstream nameOfFileToCreate;

nameOfFileToCreate << iterationToGet << ".txt";

string strNameOfFileToCreate = nameOfFileToCreate.str();

cout << "File to create: " << strNameOfFileToCreate << endl;

/\* \*\*\*\*\* \*/

ofstream outStream(strNameOfFileToCreate.c\_str());

/\* \*\*\*\*\* \*/

while(getline(alphabet, alphabetCurrentLine)) {

while(getline(prevIterFile, prevIterFileCurrentLine)) {

cout << alphabetCurrentLine << prevIterFileCurrentLine << endl;

if(checkUniqueElements) {

hasUniqueElements = stringHasUniqueElements(alphabetCurrentLine + prevIterFileCurrentLine);

if(hasUniqueElements) {

cout << "WILL ADD " << alphabetCurrentLine << prevIterFileCurrentLine << endl;

outStream << alphabetCurrentLine << prevIterFileCurrentLine << endl;

}

else {

cout << "WILL NOT ADD " << alphabetCurrentLine << prevIterFileCurrentLine << endl;

}

}

else {

outStream << alphabetCurrentLine << prevIterFileCurrentLine << endl;

}

}

prevIterFile.close();

prevIterFile.clear();

prevIterFile.open(prevIterFileName.c\_str());

}

outStream.close();

return 0;

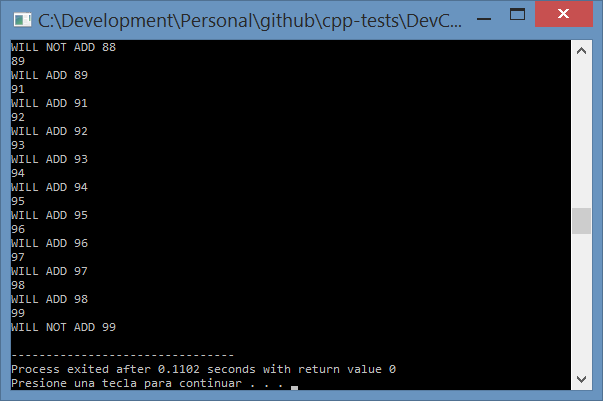
}

This is one approach of many that you could have.

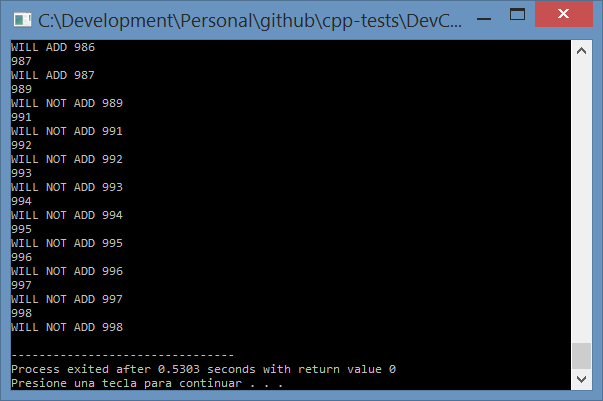
This code will assume you have a local file called 1.txt with your alphabet, having each number on a single line. Then, you will need to run the code for each iteration from 1 to 9 to get the final length 9 words.

Let’s see the times for each combination:

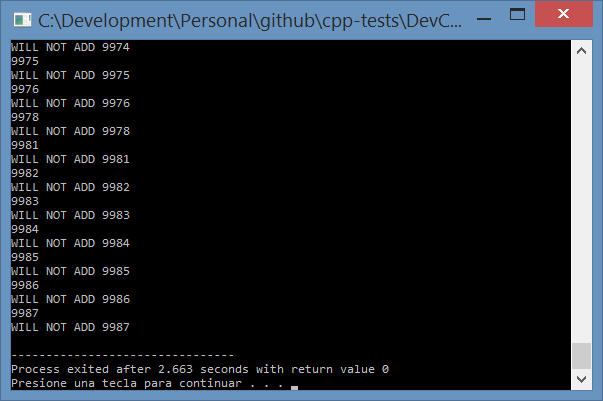
### Length 2



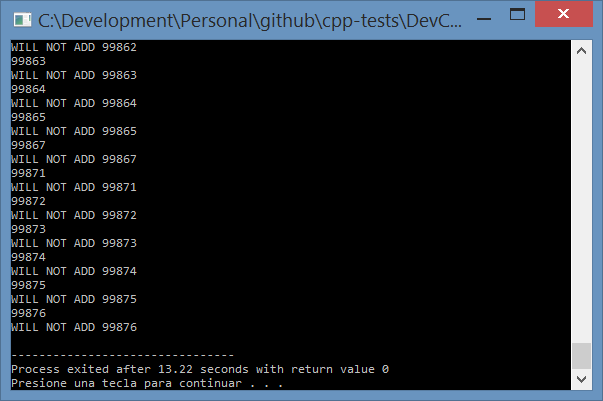
### Length 3



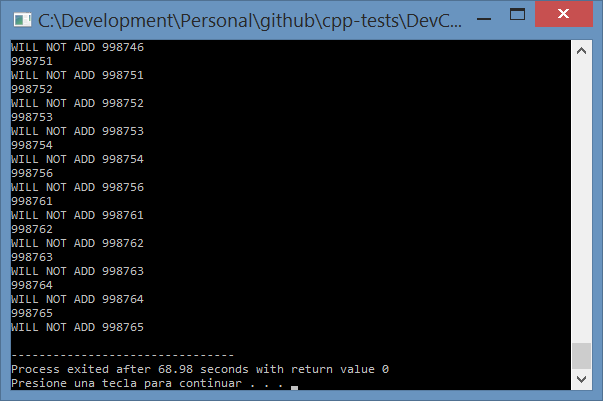
### Length 4



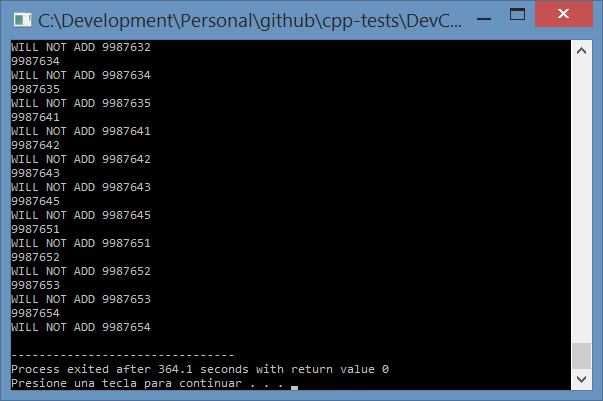
### Length 5



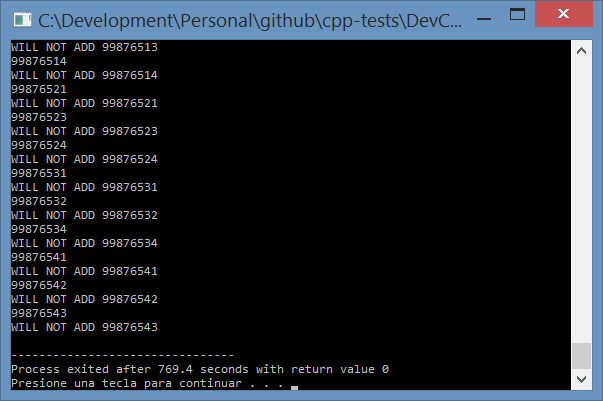
### Length 6



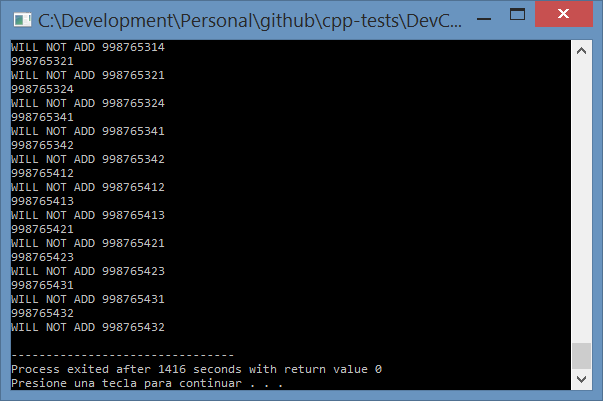
### Length 7



### Length 8



### Length 9



Now we have the length 9 words. With this we can go to Microsoft Excel, copy them, and see which one(s) solve the problem.

|  |  |
| --- | --- |
| **Length** | **Time (Sec.)** |
| 2 | 0,1102 |
| 3 | 0,5303 |
| 4 | 2,663 |
| 5 | 13,22 |
| 6 | 68,98 |
| 7 | 364,1 |
| 8 | 769,4 |
| 9 | 1416 |

Since this C++ implementation generates the words with a recursive algorithm, the execution time grows exponentially.

Now we can get the content from 9.txt and copy it to an empty Microsoft Excel sheet.

In my case, I added some columns names from B2 to M2.

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Combination** | **A** | **B** | **C** | **D** | **E** | **F** | **G** | **H** | **I** | **Result** | **Bingo?** |

From B3 to B362883 I pasted the words.

In order to get each number, from column C to column K I added the following:

|  |  |  |
| --- | --- | --- |
| **Column** | **Variable** | **Value** |
| C | **A** | =EXTRAE(B3;1;1) |
| D | **B** | =EXTRAE(B3;2;1) |
| E | **C** | =EXTRAE(B3;3;1) |
| F | **D** | =EXTRAE(B3;4;1) |
| G | **E** | =EXTRAE(B3;5;1) |
| H | **F** | =EXTRAE(B3;6;1) |
| I | **G** | =EXTRAE(B3;7;1) |
| J | **H** | =EXTRAE(B3;8;1) |
| K | **I** | =EXTRAE(B3;9;1) |

EXTRAE is the MID function in Spanish [6].

This step is completely unnecessary if you get the combinations separated by a tabulation or by a comma and let Excel process them. I just used an old code to generate word combinations.

The full Microsoft Excel file I used can be seen here:

<https://github.com/jpruiz114/cpp-tests/blob/master/DevCppTest02/Solution.xlsx?raw=true>

In the column L I just added the equation, like this:

=C3+13\*D3/E3+F3+12\*G3-H3-11+I3\*J3/K3-10

You might be asking. What about the parentheses? Operators have their priority and you should not need to add any parenthesis so Microsoft Excel know what elements should evaluate first.

The equation has division and multiplication (With the highest priority) and subtraction and addition (With the lowest priority).

The last step is to add a formula on column M like this: =SI(L3=66; "Bingo"; "") – “SI” is the equivalent for the IF function of Microsoft Excel in Spanish [6].

In addition, to make things easy, a filter that shows only the records where column M equals “Bingo” would give you the 136 answers

In the end, you get the following solutions:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 1 | 2 | 6 | 4 | 7 | 8 | 3 | 5 | 9 | | 1 | 2 | 6 | 4 | 7 | 8 | 5 | 3 | 9 | | 1 | 3 | 2 | 4 | 5 | 8 | 7 | 9 | 6 | | 1 | 3 | 2 | 4 | 5 | 8 | 9 | 7 | 6 | | 1 | 3 | 2 | 9 | 5 | 6 | 4 | 7 | 8 | | 1 | 3 | 2 | 9 | 5 | 6 | 7 | 4 | 8 | | 1 | 3 | 4 | 7 | 6 | 5 | 2 | 9 | 8 | | 1 | 3 | 4 | 7 | 6 | 5 | 9 | 2 | 8 | | 1 | 3 | 6 | 2 | 7 | 9 | 4 | 5 | 8 | | 1 | 3 | 6 | 2 | 7 | 9 | 5 | 4 | 8 | | 1 | 3 | 9 | 4 | 7 | 8 | 2 | 5 | 6 | | 1 | 3 | 9 | 4 | 7 | 8 | 5 | 2 | 6 | | 1 | 4 | 8 | 2 | 7 | 9 | 3 | 5 | 6 | | 1 | 4 | 8 | 2 | 7 | 9 | 5 | 3 | 6 | | 1 | 5 | 2 | 3 | 4 | 8 | 7 | 9 | 6 | | 1 | 5 | 2 | 3 | 4 | 8 | 9 | 7 | 6 | | 1 | 5 | 2 | 8 | 4 | 7 | 3 | 9 | 6 | | |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 1 | 5 | 2 | 8 | 4 | 7 | 9 | 3 | 6 | | 1 | 5 | 3 | 9 | 4 | 2 | 7 | 8 | 6 | | 1 | 5 | 3 | 9 | 4 | 2 | 8 | 7 | 6 | | 1 | 8 | 3 | 7 | 4 | 5 | 2 | 6 | 9 | | 1 | 8 | 3 | 7 | 4 | 5 | 6 | 2 | 9 | | 1 | 9 | 6 | 4 | 5 | 8 | 3 | 7 | 2 | | 1 | 9 | 6 | 4 | 5 | 8 | 7 | 3 | 2 | | 1 | 9 | 6 | 7 | 5 | 2 | 3 | 4 | 8 | | 1 | 9 | 6 | 7 | 5 | 2 | 4 | 3 | 8 | | 2 | 1 | 4 | 3 | 7 | 9 | 5 | 6 | 8 | | 2 | 1 | 4 | 3 | 7 | 9 | 6 | 5 | 8 | | 2 | 3 | 6 | 1 | 7 | 9 | 4 | 5 | 8 | | 2 | 3 | 6 | 1 | 7 | 9 | 5 | 4 | 8 | | 2 | 4 | 8 | 1 | 7 | 9 | 3 | 5 | 6 | | 2 | 4 | 8 | 1 | 7 | 9 | 5 | 3 | 6 | | 2 | 6 | 9 | 8 | 5 | 1 | 4 | 7 | 3 | | 2 | 6 | 9 | 8 | 5 | 1 | 7 | 4 | 3 | | |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 2 | 8 | 6 | 9 | 4 | 1 | 5 | 7 | 3 | | 2 | 8 | 6 | 9 | 4 | 1 | 7 | 5 | 3 | | 2 | 9 | 6 | 3 | 5 | 1 | 4 | 7 | 8 | | 2 | 9 | 6 | 3 | 5 | 1 | 7 | 4 | 8 | | 3 | 1 | 4 | 2 | 7 | 9 | 5 | 6 | 8 | | 3 | 1 | 4 | 2 | 7 | 9 | 6 | 5 | 8 | | 3 | 2 | 1 | 5 | 4 | 7 | 8 | 9 | 6 | | 3 | 2 | 1 | 5 | 4 | 7 | 9 | 8 | 6 | | 3 | 2 | 4 | 8 | 5 | 1 | 7 | 9 | 6 | | 3 | 2 | 4 | 8 | 5 | 1 | 9 | 7 | 6 | | 3 | 2 | 8 | 6 | 5 | 1 | 7 | 9 | 4 | | 3 | 2 | 8 | 6 | 5 | 1 | 9 | 7 | 4 | | 3 | 5 | 2 | 1 | 4 | 8 | 7 | 9 | 6 | | 3 | 5 | 2 | 1 | 4 | 8 | 9 | 7 | 6 | | 3 | 6 | 4 | 9 | 5 | 8 | 1 | 7 | 2 | | 3 | 6 | 4 | 9 | 5 | 8 | 7 | 1 | 2 | | 3 | 9 | 2 | 8 | 1 | 5 | 6 | 7 | 4 | | |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 3 | 9 | 2 | 8 | 1 | 5 | 7 | 6 | 4 | | 3 | 9 | 6 | 2 | 5 | 1 | 4 | 7 | 8 | | 3 | 9 | 6 | 2 | 5 | 1 | 7 | 4 | 8 | | 4 | 2 | 6 | 1 | 7 | 8 | 3 | 5 | 9 | | 4 | 2 | 6 | 1 | 7 | 8 | 5 | 3 | 9 | | 4 | 3 | 2 | 1 | 5 | 8 | 7 | 9 | 6 | | 4 | 3 | 2 | 1 | 5 | 8 | 9 | 7 | 6 | | 4 | 3 | 9 | 1 | 7 | 8 | 2 | 5 | 6 | | 4 | 3 | 9 | 1 | 7 | 8 | 5 | 2 | 6 | | 4 | 9 | 6 | 1 | 5 | 8 | 3 | 7 | 2 | | 4 | 9 | 6 | 1 | 5 | 8 | 7 | 3 | 2 | | 5 | 1 | 2 | 9 | 6 | 7 | 3 | 4 | 8 | | 5 | 1 | 2 | 9 | 6 | 7 | 4 | 3 | 8 | | 5 | 2 | 1 | 3 | 4 | 7 | 8 | 9 | 6 | | 5 | 2 | 1 | 3 | 4 | 7 | 9 | 8 | 6 | | 5 | 3 | 1 | 7 | 2 | 6 | 8 | 9 | 4 | | 5 | 3 | 1 | 7 | 2 | 6 | 9 | 8 | 4 | |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 5 | 4 | 1 | 9 | 2 | 7 | 3 | 8 | 6 | | 5 | 4 | 1 | 9 | 2 | 7 | 8 | 3 | 6 | | 5 | 4 | 8 | 9 | 6 | 7 | 1 | 3 | 2 | | 5 | 4 | 8 | 9 | 6 | 7 | 3 | 1 | 2 | | 5 | 7 | 2 | 8 | 3 | 9 | 1 | 6 | 4 | | 5 | 7 | 2 | 8 | 3 | 9 | 6 | 1 | 4 | | 5 | 9 | 3 | 6 | 2 | 1 | 7 | 8 | 4 | | 5 | 9 | 3 | 6 | 2 | 1 | 8 | 7 | 4 | | 6 | 2 | 8 | 3 | 5 | 1 | 7 | 9 | 4 | | 6 | 2 | 8 | 3 | 5 | 1 | 9 | 7 | 4 | | 6 | 3 | 1 | 9 | 2 | 5 | 7 | 8 | 4 | | 6 | 3 | 1 | 9 | 2 | 5 | 8 | 7 | 4 | | 6 | 9 | 3 | 5 | 2 | 1 | 7 | 8 | 4 | | 6 | 9 | 3 | 5 | 2 | 1 | 8 | 7 | 4 | | 7 | 1 | 4 | 9 | 6 | 5 | 2 | 3 | 8 | | 7 | 1 | 4 | 9 | 6 | 5 | 3 | 2 | 8 | | 7 | 2 | 8 | 9 | 6 | 5 | 1 | 3 | 4 | | |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 7 | 2 | 8 | 9 | 6 | 5 | 3 | 1 | 4 | | 7 | 3 | 1 | 5 | 2 | 6 | 8 | 9 | 4 | | 7 | 3 | 1 | 5 | 2 | 6 | 9 | 8 | 4 | | 7 | 3 | 2 | 8 | 5 | 9 | 1 | 6 | 4 | | 7 | 3 | 2 | 8 | 5 | 9 | 6 | 1 | 4 | | 7 | 3 | 4 | 1 | 6 | 5 | 2 | 9 | 8 | | 7 | 3 | 4 | 1 | 6 | 5 | 9 | 2 | 8 | | 7 | 5 | 2 | 8 | 4 | 9 | 1 | 3 | 6 | | 7 | 5 | 2 | 8 | 4 | 9 | 3 | 1 | 6 | | 7 | 6 | 4 | 8 | 5 | 9 | 1 | 3 | 2 | | 7 | 6 | 4 | 8 | 5 | 9 | 3 | 1 | 2 | | 7 | 8 | 3 | 1 | 4 | 5 | 2 | 6 | 9 | | 7 | 8 | 3 | 1 | 4 | 5 | 6 | 2 | 9 | | 7 | 9 | 6 | 1 | 5 | 2 | 3 | 4 | 8 | | 7 | 9 | 6 | 1 | 5 | 2 | 4 | 3 | 8 | | 8 | 2 | 4 | 3 | 5 | 1 | 7 | 9 | 6 | | 8 | 2 | 4 | 3 | 5 | 1 | 9 | 7 | 6 | | |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 8 | 3 | 2 | 7 | 5 | 9 | 1 | 6 | 4 | | 8 | 3 | 2 | 7 | 5 | 9 | 6 | 1 | 4 | | 8 | 5 | 2 | 1 | 4 | 7 | 3 | 9 | 6 | | 8 | 5 | 2 | 1 | 4 | 7 | 9 | 3 | 6 | | 8 | 5 | 2 | 7 | 4 | 9 | 1 | 3 | 6 | | 8 | 5 | 2 | 7 | 4 | 9 | 3 | 1 | 6 | | 8 | 6 | 4 | 7 | 5 | 9 | 1 | 3 | 2 | | 8 | 6 | 4 | 7 | 5 | 9 | 3 | 1 | 2 | | 8 | 6 | 9 | 2 | 5 | 1 | 4 | 7 | 3 | | 8 | 6 | 9 | 2 | 5 | 1 | 7 | 4 | 3 | | 8 | 7 | 2 | 5 | 3 | 9 | 1 | 6 | 4 | | 8 | 7 | 2 | 5 | 3 | 9 | 6 | 1 | 4 | | 8 | 9 | 2 | 3 | 1 | 5 | 6 | 7 | 4 | | 8 | 9 | 2 | 3 | 1 | 5 | 7 | 6 | 4 | | 9 | 1 | 2 | 5 | 6 | 7 | 3 | 4 | 8 | | 9 | 1 | 2 | 5 | 6 | 7 | 4 | 3 | 8 | | 9 | 1 | 4 | 7 | 6 | 5 | 2 | 3 | 8 | | |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 9 | 1 | 4 | 7 | 6 | 5 | 3 | 2 | 8 | | 9 | 2 | 8 | 7 | 6 | 5 | 1 | 3 | 4 | | 9 | 2 | 8 | 7 | 6 | 5 | 3 | 1 | 4 | | 9 | 3 | 1 | 6 | 2 | 5 | 7 | 8 | 4 | | 9 | 3 | 1 | 6 | 2 | 5 | 8 | 7 | 4 | | 9 | 3 | 2 | 1 | 5 | 6 | 4 | 7 | 8 | | 9 | 3 | 2 | 1 | 5 | 6 | 7 | 4 | 8 | | 9 | 4 | 1 | 5 | 2 | 7 | 3 | 8 | 6 | | 9 | 4 | 1 | 5 | 2 | 7 | 8 | 3 | 6 | | 9 | 4 | 8 | 5 | 6 | 7 | 1 | 3 | 2 | | 9 | 4 | 8 | 5 | 6 | 7 | 3 | 1 | 2 | | 9 | 5 | 3 | 1 | 4 | 2 | 7 | 8 | 6 | | 9 | 5 | 3 | 1 | 4 | 2 | 8 | 7 | 6 | | 9 | 6 | 4 | 3 | 5 | 8 | 1 | 7 | 2 | | 9 | 6 | 4 | 3 | 5 | 8 | 7 | 1 | 2 | | 9 | 8 | 6 | 2 | 4 | 1 | 5 | 7 | 3 | | 9 | 8 | 6 | 2 | 4 | 1 | 7 | 5 | 3 | |

## Second approach

Programming languages like Python provide ways to solve this problem with less code and faster than the way seen on the first approach. The following code, found on a Stack Exchange post [7] answering the same problem uses a different and efficient approach:

**import** itertools

p **=** itertools**.**permutations**([**1.0**,** 2.0**,** 3.0**,** 4.0**,** 5.0**,** 6.0**,** 7.0**,** 8.0**,** 9.0**])**

**def** is\_66**(**a**):**

result **=** **(**a**[**0**]** **+** 13 **\*** a**[**1**]** **/** a**[**2**]** **+** a**[**3**]** **+** 12 **\*** a**[**4**]** **-** a**[**5**]** **-** 11 **+** a**[**6**]** **\*** a**[**7**]** **/** a**[**8**]** **-** 10**)**

# handle the floats correctly, i.e. result == 66.00 will exclude some solutions

**return** **(**result **>** 65.99**)** **and** **(**result **<** 66.01**)**

solution\_counter **=** 0**;**

**for** a **in** p**:**

**if(**is\_66**(**a**)):**

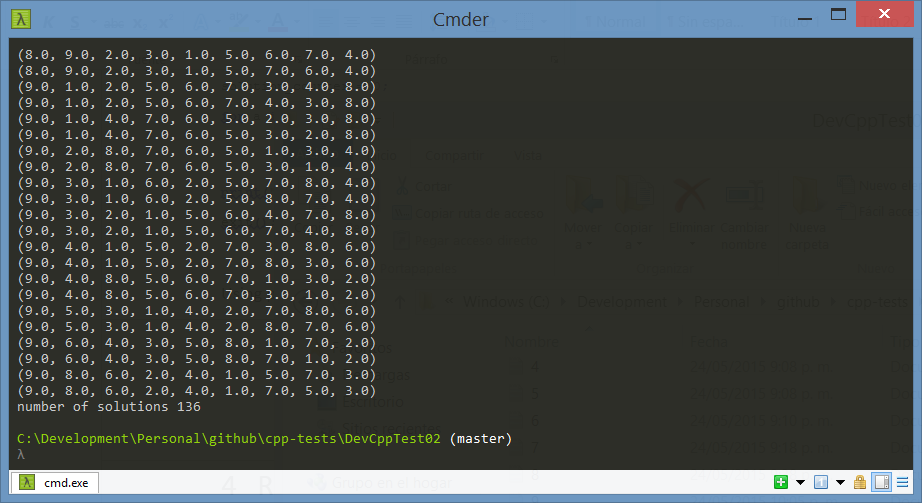
**print(**a**);**

solution\_counter **+=** 1**;**

**print(**"number of solutions %d" **%** solution\_counter**);**

exit**();**

Using python and the **itertools** library you can iterate through all the possible combinations very fast.



# Conclusions

* Using a recursive algorithm will increase the execution time of the process, and might not be a good idea if time is a problem. It might be helpful if memory consumption is an issue.
* Brute force attacks can help you find the solution of a system in a short amount of time with a high accuracy.

# Work to do

* Create an alternate C++ non recursive solution.
* Compare the time spent and the memory consumption with the first C++ approach and the pending C++ approach.
* Create a more detailed version of this document contemplating both the old approaches and the new ones.

# Sources

<https://github.com/jpruiz114/cpp-tests/tree/master/DevCppTest02>

# References

|  |  |
| --- | --- |
| [1] | «Tin nhanh VnExpress,» 18 5 2015. [En línea]. Available: http://vnexpress.net/tin-tuc/giao-duc/bai-toan-lop-3-lam-kho-ca-tien-si-3220186.html. [Último acceso: 22 5 2015]. |
| [2] | «tinmoi.vn,» 25 5 2015. [En línea]. Available: http://www.tinmoi.vn/nguoi-ra-de-bai-toan-lop-3-lam-kho-ca-tien-si-bi-de-xuat-nhac-nho-011359950.html. [Último acceso: 25 5 2015]. |
| [3] | «Tin nhanh VnExpress,» 22 5 2015. [En línea]. Available: http://vnexpress.net/tin-tuc/giao-duc/pgs-tran-dien-hien-bai-toan-lop-3-nam-ngoai-chuong-trinh-sach-giao-khoa-3222585.html. [Último acceso: 22 5 2015]. |
| [4] | A. Bellos, "The Guardian," 5 20 2015. [Online]. Available: http://www.theguardian.com/science/alexs-adventures-in-numberland/2015/may/20/can-you-do-the-maths-puzzle-for-vietnamese-eight-year-olds-that-has-stumped-parents-and-teachers. [Accessed 5 22 2015]. |
| [5] | J. Condliffe, "gizmodo," 5 20 2015. [Online]. Available: http://gizmodo.com/can-you-solve-this-vietnamese-math-puzzle-for-8-year-ol-1705734738. [Accessed 5 22 2015]. |
| [6] | "Piuha," [Online]. Available: http://www.piuha.fi/excel-function-name-translation/index.php?page=espanol-english.html. [Accessed 5 22 2015]. |
| [7] | «Mathematics Stack Exchange,» 18 5 2015. [En línea]. Available: http://math.stackexchange.com/questions/1288170/a-3rd-grade-math-problem-fill-in-blanks-with-numbers-to-obtain-a-valid-equation/. [Último acceso: 22 5 2015]. |