csce350 — Data Structures and Algorithms
Fall 2019 — Project 2

Assigned: October 3   **Due**: October 24, 11:54pm

For this assignment, you will implement an exhaustive search algorithm to solve a certain type of math puzzle. The purpose is to give you some practive designing and algorithm based on the brute force algorithm design strategy, along with some practice translating a non-trivial algorithmic idea into working code.

Background This project is about solving **cryptarithmetic** puzzles, which are math problems written with letters, in which each letter represents a digit. Here's an example of such a puzzle:

$$\text{send} + \text{more} = \text{money} \tag{1}$$

This one happens to have just one solution, shown in the following table:

Letter	Digit
d	7
e	5
m	1
n	6
o	0
r	8
s	9
y	2

If we use this table to subsitute letters for numbers, we get this addition equation:

$$9567 + 1085 = 10652 \tag{2}$$

The goal is to take something like Equation 1 and find all of its solutions like Equation 2. Some details:

- The puzzle consists of three words: two *addends* and a *sum*. Each word is a short sequence of lowercase letters. (We are only interested in cryptarithmetic puzzles with two numbers added to each other to form a third number. As you might imagine, there are similar problems out there that have more inputs, different arithmetic operations, etc. We'll keep it simple here, but after finishing this project, you might want to try expanding your program.)
- Each letter in each of the three words represents a single digit: 0, 1, 2, 3, 4, 5, 6, 7, 8, or 9. The goal is to determine which letters represent which digits, to make the addition work out correctly.
- Letters can appear more than once, but they represent the same digit each time they appear.
- Each letter represents a unique digit: Two different letters cannot represent the same digit.
- The words do not have leading zeroes: The first letter of each of the three words must represent a digit other than 0.

Your Task You should write a C++ program that does these things:

1. First, your program should read a crpytarithemic problem from standard input. The input will have a sequence of lowercase letters (the first addend), followed by a plus, followed by another sequence of lowercase letters (the second addend), followed by an equals sign, followed by another sequence of lowercase letters (the sum), followed by a newline. You may assume that the input obeys this format; there is no need to write error-checking code for this project.

2. Then, your program should solve that cryptarithmetic problem using *exhaustive search over all of the valid letter-to-digit mapping tables* like the one shown above. (You will probably need to think a bit about how to represent a candidate solution, how to enumerate all of the candidate solutions, and how to check each candidate to see if it's a solution.)

   Your program should produce one line of output for each correct equation that can be formed from the given puzzle. Each output line should match the input, but with digits replacing the letters. If there are no solutions, your program should output a single line that reads:

   ```
   no solution
   ```

   If there are multiple solutions, your program may list them in any order, but it should find and list all of them.

3. After printing all of the solutions (or the `no solution` message), your program should print a blank line.

4. After solving a puzzle, the program should return to Step 1 above to read and solve another puzzle. The program should terminate when standard input reaches end-of-file.

Sample Input

```
a+a=bc
ab+bc=bcb
ab+bb=aa
base+ball=games
```

Sample Output

```
5+5=10
6+6=12
7+7=14
8+8=16
9+9=18

91+10=101

no solution

7483+7455=14938
```

For calibration purposes, my solution has 97 lines of code, including comments; my program takes about 2 minutes solve all the examples above, most of it spent on `base+ball=games`.

What to Submit You should submit, using the department's dropbox website, a single C++ source file named containing all of the code for your program. I will compile this program using this command line:

```
g++ -Wall -std=c++11 yourfile.cpp
```