In a summation puzzle, you are given three strings of the form POT + PAN = BIB. Typically each is a word, often with a theme to the three chosen. Your goal is to assign a ​distinct​ digit to each letter in the equation in order to make the resulting true. For example, if the puzzle is POT + PAN = BIB, the mapping P:2, O:3, T:1, A:7, N:4, B:5, I:0 will solve this, as 231 + 274 = 505.

You are required to implement the function puzzleSolver in proj1.cpp. This function should return true if, and only if, the puzzle is solvable: that is, if there is a mapping of the letters that appear in the three strings to ​distinct​ digits such that the sum of the first two is the third. No string will have a value larger than 4,294,967,295 in its correct substitution, nor will the addition have any integer-overflow to check for. If you do not know what integer overflow is, you do not need to check during this assignment (although it’s worth knowing in general).

For this project, you have a few requirements:

● You must implement the function puzzleSolver in proj1.cpp​. ​You may assume it is called

with three valid non-empty strings as parameters and with an otherwise empty map. The

strings will always consist only of all-capital letters.

● Your solution must explicitly use recursion in a meaningful way towards solving the

problem. You may ​not ​solve this by using a function like std::next\_permutation (from <algorithm>) to enumerate possibilities.

○ The function ​puzzleSolver ​itself need not be recursive if you would prefer to have a helper function that is.

● The function must return a boolean indicating whether or not the puzzle ​has​ a solution.

○ If the puzzle ​does not​ have a solution, your function should return false.

○ If the puzzle ​does​ have a solution your function should return true ​and​ have the

map<char, unsigned> parameter containing the mapping needing to verify. That is, the four parameters to the ​puzzleSolver ​function need to be such that a correct solution to project 0 would return true with those parameters.

○ If there are multiple solutions, returning any of them is fine. You can think of my grading code as this:

■ I know if the test case has or hasn't a solution. I check that you return the right bool value.

■ If it has a solution, I also run a (correct) solution to proj0 on the three strings + the map's status at the end of your function.

○ If the previous point means you need to modify the given gtest code, feel free to do so -- my real grading code will run a verifier instead of checking for an explicitly expected mapping.

● Your program must run in under three minutes on a reasonably modern computer. Test cases that take longer than this to run may be deemed to be incorrect. Note that this means you will need to think a little about efficiency in your program.

You ​may​ use standard libraries as appropriate, unless there is one that makes solving this problem trivial. I am unaware of any such part of the library.

You are ​explicitly permitted​ to use std::set, std::list, std::queue, and std::stack if you so choose. You are pretty much required to use std::map