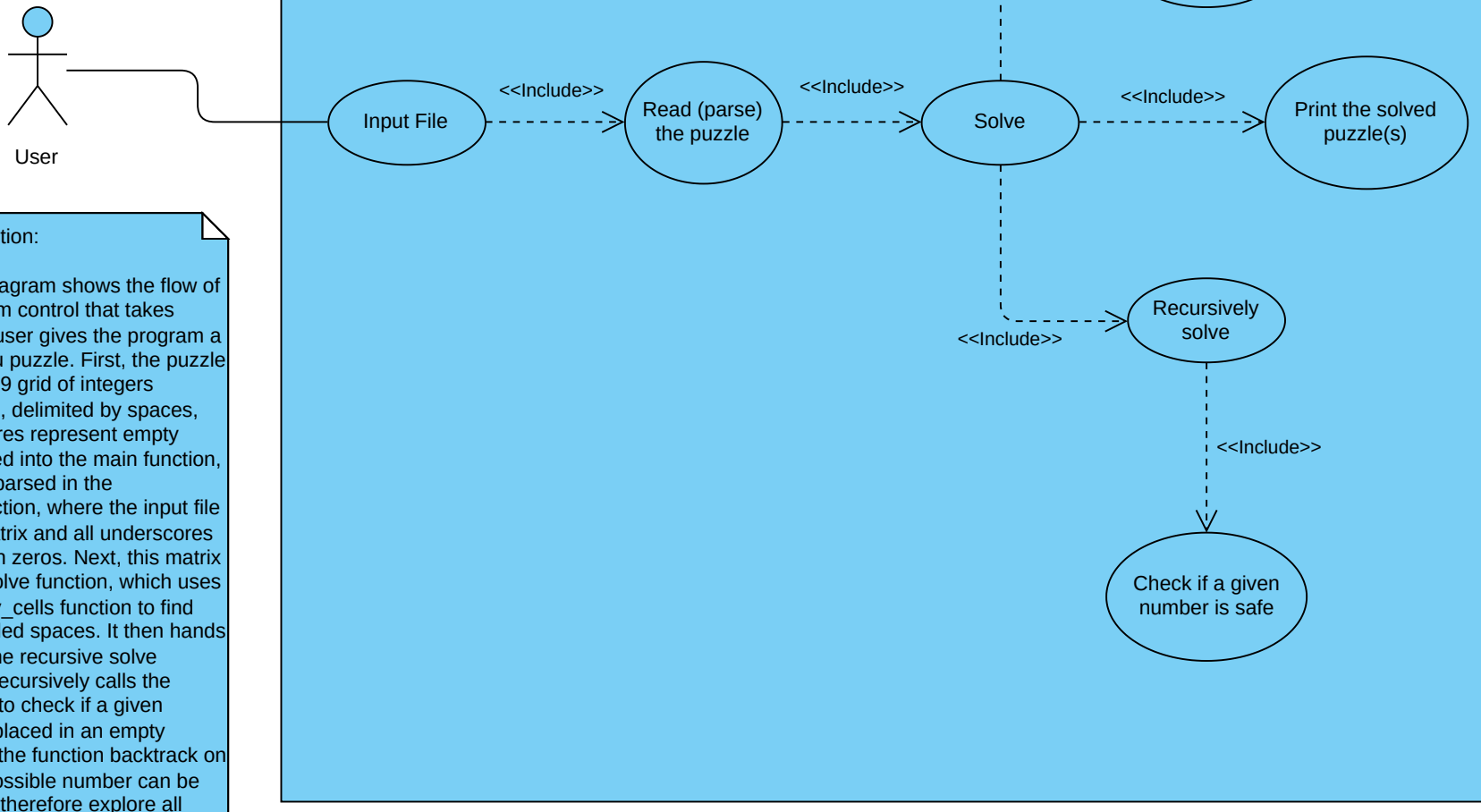
Made with VisualParadigm
For non-commercial use

Brief Description:

This diagram describes the interaction of the user with the sudoku solver program. The user only has one use case available to them, which is to input a unsolved sudoku puzzle into the Assingnment6.cpp and recieve all possible ways to solve the given puzzle.



Detailed Description:

This use case diagram shows the flow of data and program control that takes place when the user gives the program a unsolved sudoku puzzle. First, the puzzle (a .txt file of a 9x9 grid of integers ranging from 1-9, delimited by spaces, where underscores represent empty spaces) is passed into the main function, where it is then parsed in the read_matrix function, where the input file is stored in a matrix and all underscores are replaced with zeros. Next, this matrix is given to the solve function, which uses the count_empty_cells function to find the number of filled spaces. It then hands control over to the recursive solve function, which recursively calls the is_safe function to check if a given number can be placed in an empty space. This lets the function backtrack on itself when no possible number can be placed, and can therefore explore all possibilities for solving the input puzzle. Finally, all possible solutions are handed to the print_matrix function to be printed to the terminal.