Software Testing Project 1

R02921045 吳庭棻

SUT

Sudoku Solver

Recursively find a solution of input Sudoku problem. If the solution be found, output the fulfilled result. If not, output “no solution”

System Requirement

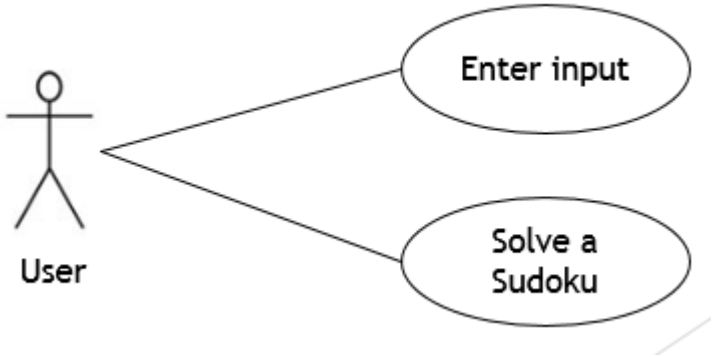
Input format:

0 in the input means blank to fill in, number in 1-9 means itself.

Input & output should match Sudoku constraints: 9x9 array, in each row, column and 9 sub-grids(3x3), same integer would not appear twice

User Requirement

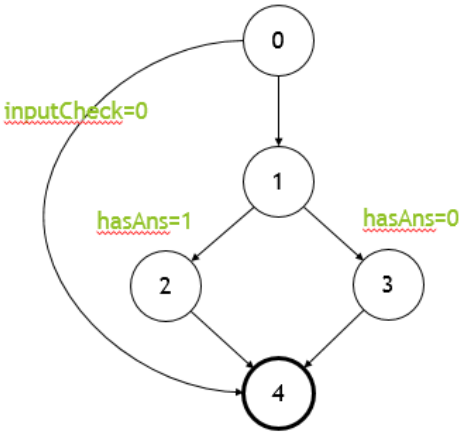
Given a Sudoku problem and find a solution or print “no solution”, and user may enter illegal input, check it before solving.



Test Criteria

Complete Path Coverage

[0, 1, 2, 4], [0, 1, 3, 4], [0, 4]



Test Requirement

Test the main function of sudoku.c by different input, I have tested about 10+ inputs and select 4 most significant of them for coverage.

Case 1: Legal input with solution

Case 2: Legal input but no solution

Case 3: Legal input with solution (bug)

Case 4: Illegal input

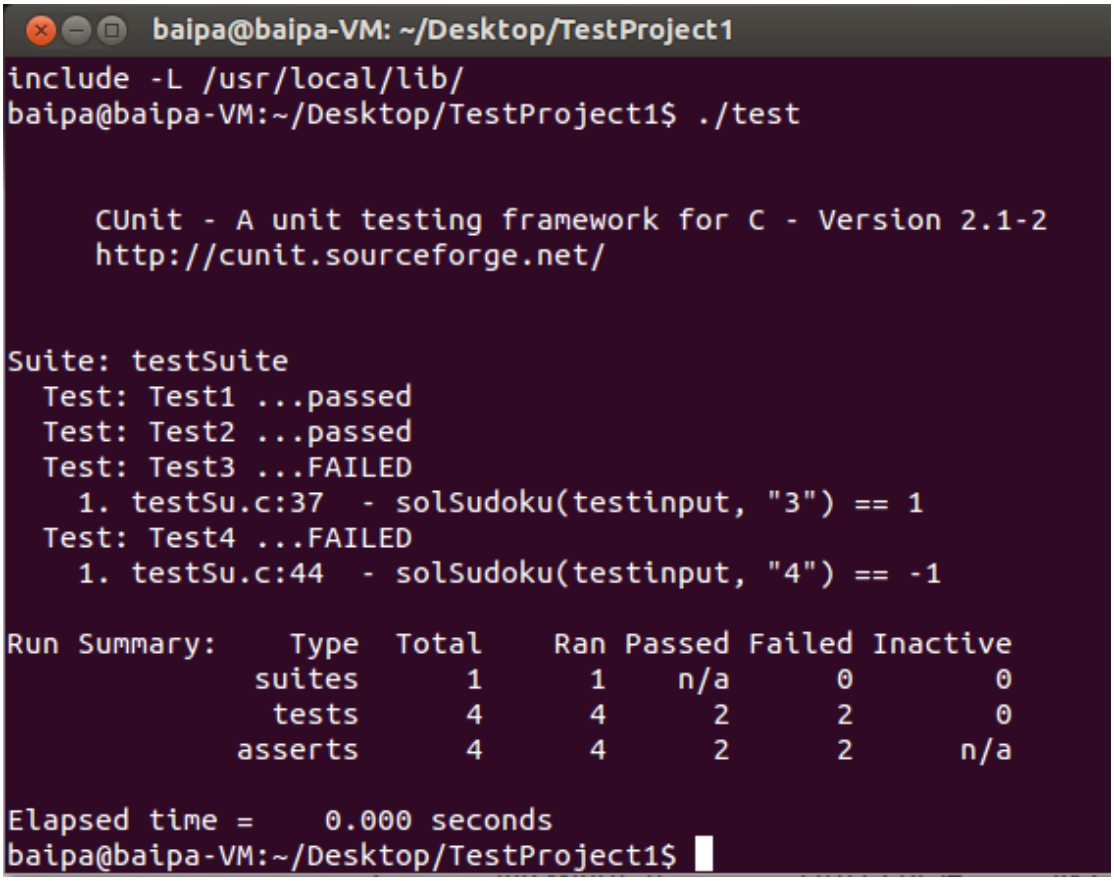
Implementation

Environment: Ubuntu 13.10 64-bit on VirtualBox

Use CUnit for testing

1. Modify the original program from standard I/O to file I/O
2. Write 4 test input into individual file
3. Write test program testSu.c for testing

Result Screenshot



The 3rd test case failed because the program has a bug.

The program doesn't handle with input checking, so the 4th test case failed.

Conclusion

Analyzing user requirement and determining which test criteria should be used are important before testing. CUnit can help us test the program systematically, but it doesn’t handle Segmentation fault