

Sudoku Solver – Report


Design Approach

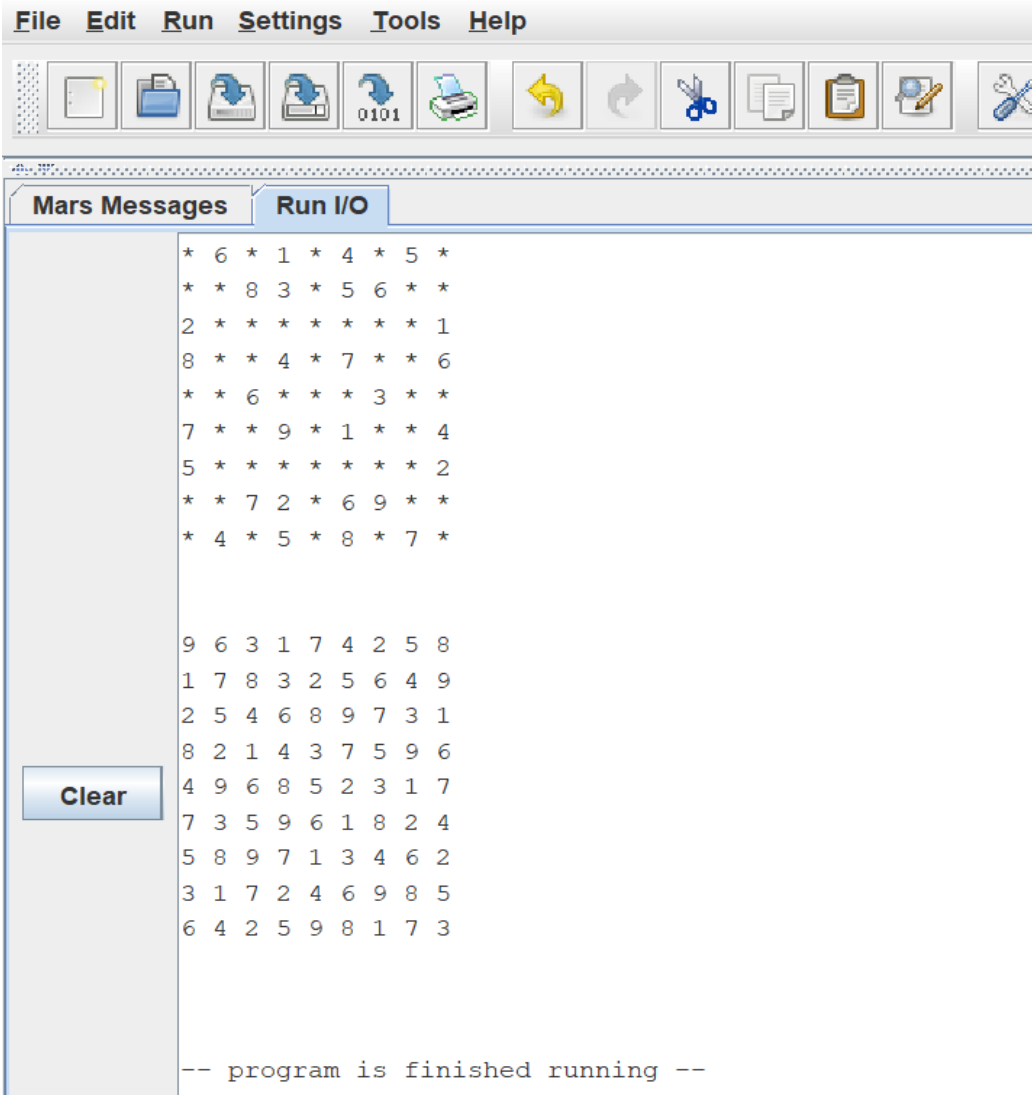
The program takes a partially filled 9x9 grid and solves the sudoku. The program starts by looking for an empty spot in the grid. Once it is found, the address of the first element in that same row and the first element in the column are calculated using the row and column index. The program loops in the loop named “assign” to try and place a value (1-9) in the empty spot. The row is checked first, then the column, then the 3x3 box. The address of the first element in the specific 3x3 box (containing the empty spot) is calculated using the row and column index. A bitwise operator is used to compare the value being inserted in the grid with the values in the row, column and 3x3 boxes, by using xor. When a value is chosen for the empty spot, all important parameters (address of empty spot, address of first element in row, address of first element in column, row index, column index and value placed in the spot) are all pushed to the stack. The function is then called recursively to continue solving the grid after placing that number. If the “assign” loop fails to place a number, it tries the next one (1-9). If all 9 options fail, the program backtracks. When backtracking, the number in the previous spot is removed (zero) and parameters are popped from the stack. Then, we continue looping in “assign” to try the rest of the options, since that one did not lead to a solution. The base case is when there are no more empty places in the grid, so the solved grid is printed.

The test case I have in the data section might take a while to run, as it includes a lot of backtracking. The screenshots of my running programs are below.


Screenshots

MIPS

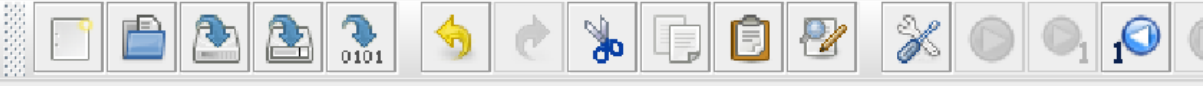
 C:\Users\Nada\Desktop\Semester 4\Assembly\assignment4\mips.asm - MARS 4.5



RISC-V

 C:\Users\Nada\Desktop\Semester 4\Assembly\assignment4\riscv.asm - RARS 1.5

File Edit Run Settings Tools Help



Messages Run I/O

Clear

```
* 6 * 1 * 4 * 5 *
* * 8 3 * 5 6 * *
2 * * * * * * * 1
8 * * 4 * 7 * * 6
* * 6 * * * 3 * *
7 * * 9 * 1 * * 4
5 * * * * * * * 2
* * 7 2 * 6 9 * *
* 4 * 5 * 8 * 7 *

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

-- program is finished running (0) --
```

x86

```
nadatamer@nadatamer-VirtualBox: /media/sf_csce-1102-swap-area/asm/assignment 4
File Edit View Search Terminal Help
nadatamer@nadatamer-VirtualBox:/media/sf_csce-1102-swap-area/asm/assignment 4$
nasm -f elf64 x86.asm -o obj.o
nadatamer@nadatamer-VirtualBox:/media/sf_csce-1102-swap-area/asm/assignment 4$
ld obj.o -o exe
nadatamer@nadatamer-VirtualBox:/media/sf_csce-1102-swap-area/asm/assignment 4$
./exe
* 6 * 1 * 4 * 5 *
* * 8 3 * 5 6 * *
2 * * * * * * * 1
8 * * 4 * 7 * * 6
* * 6 * * * 3 * *
7 * * 9 * 1 * * 4
5 * * * * * * * 2
* * 7 2 * 6 9 * *
* 4 * 5 * 8 * 7 *

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

nadatamer@nadatamer-VirtualBox:/media/sf_csce-1102-swap-area/asm/assignment 4$
```