



CS50 Walkthrough 4

Marta Bralic

Suduko by John Harvard

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

playing n00b #42

by John Harvard

[N]ew Game

[R]estart Game

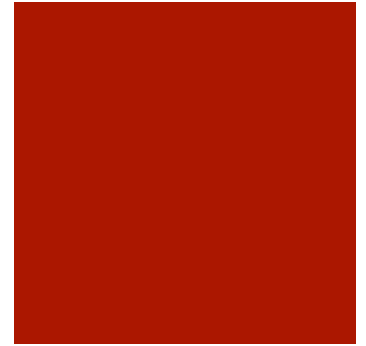
[Q]uit Game

To Do

- distribution code
- ncurses
- move cursor
- allow changing user-added numbers, but not original ones.
- allow replacement of blank with number
- invalid move?
- won?

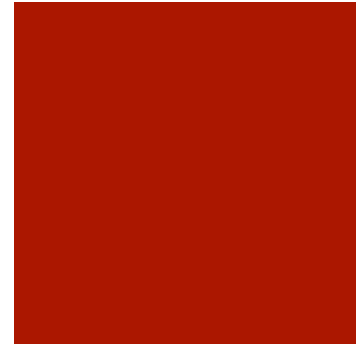


Distribution Code + Debugging



- sudoku.h
- sudoku.c
 - 600 lines!
- 2 window gdb debugging

ncurses



- sudoku.h
- Allows you to change colors, appearance of your program.
 - Always have foreground and background color.
- Allows you to have a cursor.
 - User interface
 - Updating board

Moving the cursor

- Switch statements!

```
switch (test)
{
    case x:
    case y:
        //Do this for cases x and y
    default:
        //Do this otherwise
}
```



How to refer to keys/cursor?

- Keys

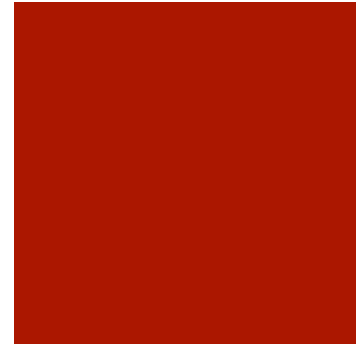
- KEY_UP
- KEY_DOWN
- KEY_LEFT
- KEY_RIGHT

- Cursor

- `g.board[g.y][g.x]` is spot on board where cursor is
 - `g.y` is cursor's y position
 - `g.x` is cursor's x position
- `showcursor()`

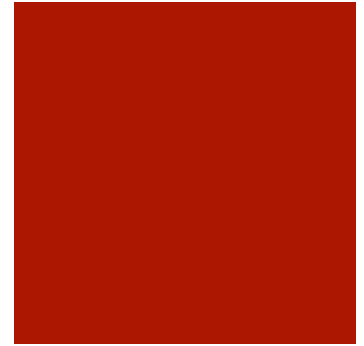


Don't replace original or move when won!



- Keep track of locations originally there.
- Before moving, ensure that it is not an original number and that game is not won
 - make a copy of the board at start.
 - If not a 0 in original board, don't change it!

Replace blanks/non-original numbers



- function, takes one argument `ch` (ascii)
 - if `ch` is 0, . , `KEY_BACKSPACE`, `KEY_DC`
 - set that spot in the board to 0
 - if `ch` is numerical between '1' and '9'
 - set that spot in the board to the values 1 through 9, not the ascii 1 through 9
 - like in Caesar, subtract '0'
- `draw_numbers()`

Invalid move!



- Check all the values in that row and column for the value in the tile.
- Check each box by starting top left, and moving 2 across, and 2 down looking for same value as `g.board[g.y][g.x]`, but “skip” `g.board[g.y][g.x]`

Won?



- Go to each box
 - Ensure no 0's
 - Check for errors
 - if no zero, and no errors, showbanner
- If not won, return to your box!