ComS 327 Project Part 4

Checkers: move viewer and editor

Fall 2019

1 Summary

For the last part of the project, you must read, display, and edit (for extra credit) a checkers exchange file. You will need to use a library (either `ncurses` or `termbox`) that supports displaying characters to specific locations in a UNIX terminal, and with desired color attributes. Your `Makefile` must now build the executables for all previous parts of the project, along with a new executable named `edit`. Your `edit` program will read (and possibly modify) an input file in exchange format, specified on the command line.

Implementation may be in C, C++, or a combination of the two.

2 Interface

The details of the user interface are not specified. However, your interface must contain the following elements.

- The name of the file being edited.
- Information about which hotkeys are active and what they do (for example, if the user is expected to press "Q" to quit, then this should be clear from the interface).
- A list of moves, arranged vertically and split out into red and black. The display should be organized so that the list of moves can take the available vertical space in the terminal.
- For each move in the list, something (color or some other attribute) to indicate if the move is valid (legal in its current position), invalid (illegal in its current position), or unknown (follows an invalid move).
- A cursor, that can be moved through the list of moves, and to the blank move after the last move in the list.
- Scrolling, when the list of moves is too long to fit in the terminal. There should be some indication as to the position in the file (examples include a scroll bar, or a counter).
- A checkerboard, that shows the game state *immediately before* the move containing the cursor. The game state following an invalid move is undefined.

Your code will be tested on the virtual machine, in a terminal with at least 80 columns and at least 25 rows. A screenshot of an example interface that satisfies the requirements is shown in Figure 1.

3 Grading

The project is worth 100 points for individuals, and 110 points for groups of 2. However, there is ample opportunity for extra credit.

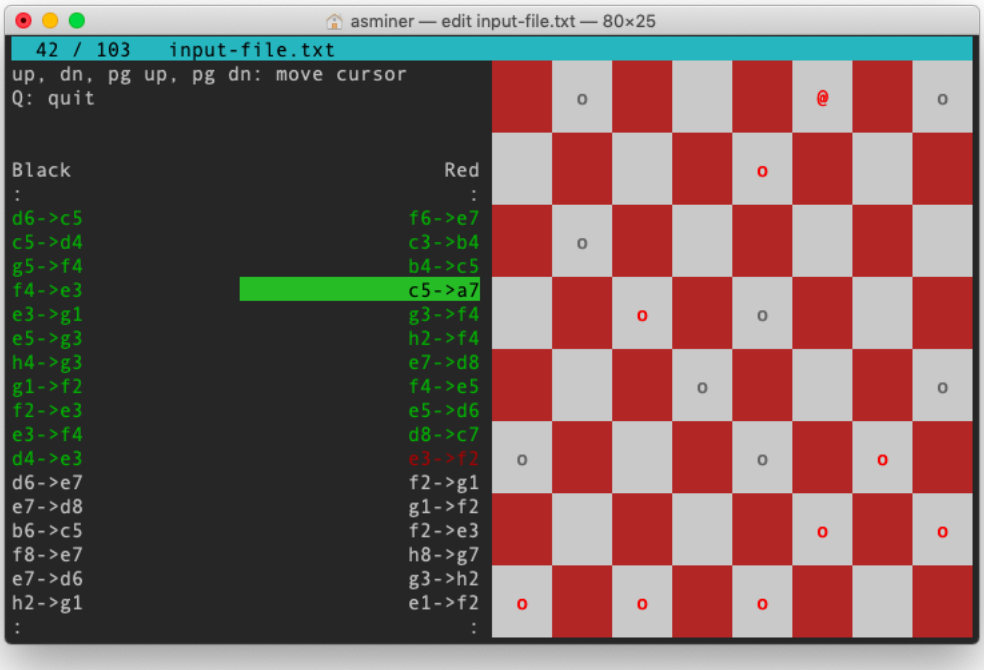

Figure 1: Screen shot of an example interface

6 pts	`README` file, that describes implemented features
7 pts	`DEVELOPERS` file
7 pts	Working `Makefile`
10 pts	Your `info` executable still works
10 pts	Your `change` executable still works
15 pts	Checkerboard display
15 pts	List of moves display
10 pts	Hotkeys display
10 pts	Moving cursor
10 pts	Scrolling
10 pts	Multiple jumps
10 pts	Forced capture
10 pts	Ability to change or append a move
5 pts	Ability to save edits

4 What to submit

Remember to submit the following in your git repository.

- Source code, in C and/or C++.
- A `Makefile`, so your executables can be built by simply typing "`make`".
- A `README` file, to indicate which features are implemented.
- A `DEVELOPERS` file, with necessary information about the source code for other developers. For group submissions, this file also should indicate which student(s) implemented which function(s).

- A `GRADEME` file, containing either

  > "Please grade my updated part 3 submission, for at most half credit. I understand that this could lower my grade for part 3."

  OR

  > " "

  An omitted file will be treated the same as an empty file.

Also, please turn in a short note in Canvas with the URL of your git repository and the members of your group.