Traffic Game: Assignment 01

By: Yoseph Campbell Student ID: 21552155 Date Created: 17/09/2023

How to run

The Makefile in the directory will contain the following code:

- `make` - create the executable file 'traffic'
- `make clean` - refreshes the folder, getting rid of object files in the folder.
- `make val` - launches valgrind to the game, you can specify the val flags through the makefile if you wish
- `make debug` - launches gdb for debugging

To start the game simply:

```
> ./traffic arg1 arg2
```

If you do not input any arguements it will just display the help.

Acceptable arguements

Arg1 - row of the map, needs to be odd and >= 3 Arg2 - col of the map, needs to be >= 5.

Anything else, the program will display an appropriate message.

Once in game

You have the following acceptable inputs:

`w` - moves you up if it is legal `a` - moves you left if it is legal `s` - moves you down if it is legal `d` - moves you right if it is legal `q` - to quit the program

Anything else is an invalid key and will do nothing.

How the program works

The is divided into the following: `main.c` - contains the CLI `board.c` - contains the Board related functions `rules.c` - contains the Rules of the game, defines what is legal and what isn't, the main "mind" of the game, this will act as the interface to the actual game to the CLI `players.c` - contains the player related function `cars.c` - contains the car related functions

Challenges

This was a really fun assignment, initially was setback with how to tackle the map generation but came down to the solution I have now:

- An empty board is the default map which just includes:
  - borders, empty spaces, roads, goals

Since these do not change positions at all this is our 'blank canvas'. Meanwhile, player position, car positions were the ones always change in every turn.

So upon setUp, the board is setup, player location is set up and also car positions and directions are set up aswell. This will allow future changes and updates, to change the player position, car position if needed making it more maintainable. Because the amount of cars changes based on the rows follows a simple sequence, so I used that idea to dynamically allocate the car positions in an array upon set up as well as initial directions using the provided random module.

Because I don't use C often, I found myself forgetting to free malloc'd arrays and it did frustrate me when doing this assignment as I am not used to freeing arrays just yet but after using valgrind, it was helpful in detecting the functions that seem to be causing the leaks.