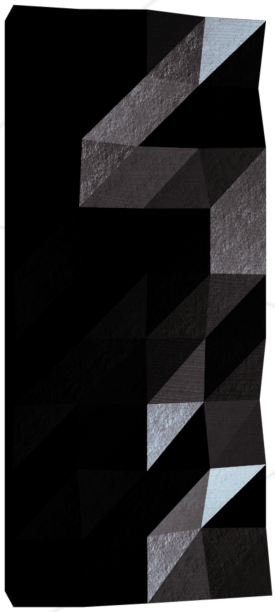

Connect Four

Summary:

Version: 1

Contents

Chapter I

Preamble

Joshua : Greetings, Professor Falken.

Stephen Falken : Hello, Joshua.

Joshua : A strange game. The only winning move is not to play. How about a nice game of chess?

Chapter II

Introduction

The goal of this rush is to write a Connect Four. That's not all, you will need to write an AI to play against you. Wonderful, isn't it?

Chapter III

General rules

- Your assignment has to be written in C.
- No norm.
- `cc` is used as compiler.
- You have to compile with the following flags `-Wall -Wextra -Werror`.
- You have to turn in a `Makefile` which will compile your source files.
- Your Makefile must at least contain the rules: `NAME, all, clean, fclean and re` and must not `relink`.
- Your program should not quite unexpectedly (segmentation fault, bus error, double free, and so forth) except for undefined behaviors.
- Within the mandatory part, you're allowed to use the following:
  - Libft.

    `malloc(3), free(3), rand(3), srand(3), time(3)`

Chapter IV

Instructions

- You need to implement the rules from here.
- The program name is `connect4`.
- The grid size must be taken as parameters to the program.
  Ex: ./connect4 8 10
- The minimum size is 6 lines and 7 columns.
  In case of invalid size, you must display an error and exit.
- The player and the AI play in turns. The first one to play is randomly choosen.
- At the start of the game and after each move, your program must display in the terminal
  the state of the grid in a clear format (except if the grid is too large to be displayed on
  a regular terminal). Pawns of both players and the positions must be easy to identify.
- The program should ask the player where to play on their turn.
  If it's invalid, it must ask again until a valid value is given.
- Your AI must always play on its turn and should not take forever to do so.
- And of course, your AI must try to win.
- At the end of the game, your program must display the winner or if it's a draw;
  and then exit properly.

Chapter V

Bonus Part

It is very cool to have a graphical interface to play, no?

- You can choose any lib you want (like sdl or ncurses).
- You must add an argument to activate or not the graphical interface.
- You must continue to also display all mandatory parts on the terminal but the position to play can be choosen on the interface.

The bonus part will only be accessed if the mandatory part is PERFECT. Perfect means the mandatory part has been integrally done and works without malfunctioning. If you have not passed ALL the mandatory requirements, your bonus part will be evaluated at all.

Chapter VI

Turn-in and peer-evualation

As usual, turn in your work on your repo GiT. Only the work included on your repo will be reviewed during the evaluation.

Good luck!