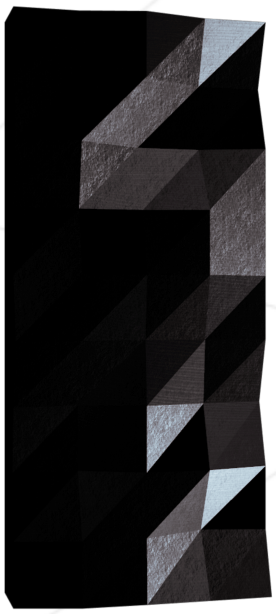

Chess

It might take some coding, but maybe you will be able to beat a human in chess.

Summary:

Version: 1

Contents

Chapter I

Introduction

This introduction is written by a human.
It's about the subject.
Definitely not a computer generated text.
Have fun!

Chapter II

Mandatory part

Program name	`Chess`
Turn in files	`chess.c`
Turn in Directory	`ex00/`
Makefile	`No`
Arguments	
External functs.	`write, malloc, free`
Libft authorized	`No`
Description	`WWrite a program that can read chess boards`

Write a program that takes rows of a chessboard as arguments and checks if your King is "in check".

The King is considered to be "in check" when an other enemy piece can capture it. When it's the case, you will print "Success" on the standard output followed by a newline, otherwise you will print "Fail" followed by a newline.

Quick reminder, chess is a game played on a chessboard, an 8 smaller-square long square board with specific pieces: King, Queen, Bishop, Knight, Rook and Pawns.

- Each moveset is detailled in the example.md file
- Your play from the bottom of the board
- The board can be of different sizes but will remain a square
- here's only one King and all other pieces are against it
- All the characters that are not used to refer to pieces are considered as empty squares
- If there are no arguments, the program will only print a newline.

Chess It might take some coding, but maybe you will be able to beat a human in chess.

Here are a few examples:

```
$> ./chessmate '..' '.K' | cat -e
Fail$
$> ./check_mate 'R...' '.K..' '..P.' '....' | cat -e
Success$
$> ./chessmate 'R...' 'iheK' '....' 'jeiR' | cat -e
Success$
$> ./chessmate | cat -e
$
$>
```

Chapter III

Submission and peer-evaluation

Turn in your assignment in your `Git` repository as usual. Only the moulinette will evaluate your work, so make sure everything works perfectly.