Python Projects

Projects

There are 2 projects for this course, you can choose one of them to work on. And you have 2 weeks from the day this document is published to submit your project.

Project 1: Build a simple chess game

In this project, you will build a simple chess game that can be played by 2 players. If you want you can add a GUI to the game, but it is not required. It is suggested to use the Object Oriented Programming paradigm to build this game but it's not required as long as your project is well structured.

Chess

The game of chess is played on an 8x8 board. Each player starts with 16 pieces: 1 king, 1 queen, 2 rooks, 2 knights, 2 bishops, and 8 pawns. The goal of the game is to checkmate the other player's king. A player's king is in check if it is under attack by the other player's pieces. A player can't make a move that puts their king in check. If a player's king is in check and there is no legal move that the player can make to get out of check, then the player is checkmated and the game is over.

Rules of the game

1. The game starts with the white player going first.
2. A player can move one of their pieces to a different square.
3. A player can't move a piece to a square that is occupied by one of their own pieces.
4. A player can't make a move that puts their king in check.
5. If a player's pawn reaches the other side of the board, the pawn is promoted to another piece except for a king or another pawn.
6. If a player's king is in check and there is no legal move that the player can make to get out of check, then the player is checkmated and the game is over.

7. If a player's king is not in check and the player can't make a legal move, then the game is a stalemate and the game is over.
8. Legal moves for each piece:
    - King: The king can move one square in any direction.
    - Queen: The queen can move any number of squares in any direction.
    - Rook: The rook can move any number of squares horizontally or vertically.
    - Bishop: The bishop can move any number of squares diagonally.
    - Knight: The knight can move in an L shape: 2 squares in one direction and 1 square in a direction perpendicular to the first direction.
    - Pawn: The pawn can move one square forward. If it is the pawn's first move, it can move two squares forward. The pawn can capture an opponent's piece by moving one square diagonally forward.
    - Pawn promotion: When a pawn reaches the other side of the board, the player can promote the pawn to any other piece except for a king or another pawn.
    - Castling: The king can move two squares towards a rook on the player's first rank and that rook moves to the square over which the king crossed. Castling is only allowed if the king and rook have not moved before and there are no pieces between the king and rook.
    - En passant: If a player moves a pawn two squares forward from its starting position and it lands next to an opponent's pawn, the opponent's pawn can capture the player's pawn as if it had only moved one square forward.
    - Check: A king is in check if it is under attack by the other player's pieces.
    - Checkmate: A player's king is in check and there is no legal move that the player can make to get out of check.
    - Stalemate: A player's king is not in check and the player can't make a legal move.
    - Draw: The game is a draw if there is no way for either player to win.
    - Resign: A player can resign at any time, which means they lose the game.
    - Draw by agreement: The players can agree to a draw at any time.
    - Draw by threefold repetition: The same position occurs three times with the same player to move.
    - Draw by the fifty-move rule: No pawn has moved and no piece has been captured in the last fifty moves.
    - Draw by insufficient material: There is no way for either player to win.
    - If a player's king is in check, the player must make a move that gets the king out of check, which is either moving the king, capturing the attacking piece, or blocking the attack.

Requirements

You can use any libraries you want to build this game. If you want to build a GUI for the game, you can use libraries like Pygame, Tkinter, or any other library you want. If you want to build a text-based game, you can use the built-in input() function to get input from the user.

Submission

You should submit your project as a single zip file that contains all the files needed to run your project. You should also include a README file that explains how to run your project including the needed libraries and how to play the game.

Example A simple text-based chess game board that uses the built-in input() function to get input from the user.

```
8 RB NB BB QB KB BB NB RB 8
7 PB PB PB PB PB PB PB PB 7
6 .. .. .. .. .. .. .. .. 6
5 .. .. .. .. .. .. .. .. 5
4 .. .. .. .. .. .. .. .. 4
3 .. .. .. .. .. .. .. .. 3
2 PW PW PW PW PW PW PW PW 2
1 RW NW BW QW KW BW NW RW 1
  a  b  c  d  e  f  g  h
```

Project 2: Liars Deck

Liar's Deck is a card game that focuses on playing cards and deception. Players can play cards face down and are allowed to make false statements about the cards they've played. The core of the game lies in the psychological warfare between players and judging whether opponents are lying.

Liar's Deck Operation Instructions:

- A and D keys: Switch cards
- Space: Select cards
- E key: Throw cards
- X key: Challenge (Call Liar!)
- Liar's Deck Rules in Liar's Bar

Basic Setup of Liar's Deck:

- The deck consists of 20 cards: 6 Aces, 6 Kings, 6 Queens, and 2 Jokers
- Jokers can substitute for any card
- 2-4 players participate, each starting with 5 cards

- Each player holds a revolver with 1 bullet randomly loaded in one of the 6 chambers
- Each player has a maximum of 30 seconds for thinking and decision-making

Game Process of Liar's Deck:

1. At the beginning of each round, the system designates the "liar's card" type for this round (e.g., "Ace" or "King").
2. Players take turns playing cards, 1-3 cards each time. For example, throw out 2 cards means the player claims to have played 2 "Aces".
3. The next player can choose to:
   - Believe the previous player's statement and play their own cards
   - Challenge the previous player's play (Call Liar!), indicating "I don't believe you just played 2 'Aces' ". Then the system reveals the pile to verify.
     - If the previous player didn't play 2 cards (e.g., 0 "Aces"), the challenge is successful, and the previous player undergoes a death roulette judgment;
     - If the previous player indeed played 2 "Aces" (including Jokers), the challenge fails, and the challenging player undergoes a death roulette judgment;Gun
4. Death roulette judgment means firing the gun at oneself. If it's an empty chamber, the game proceeds to the next round; if successful, the player is eliminated, and the game continues to the next round
5. The game continues until only one player remains, who becomes the winner.

Requirements

You can use any libraries you want to build this game. If you want to build a GUI for the game, you can use libraries like Pygame, Tkinter, or any other library you want. If you want to build a text-based game, you can use the built-in input() function to get input from the user.

Submission

You should submit your project as a single zip file that contains all the files needed to run your project. You should also include a README file that explains how to run your project including the needed libraries and how to play the game.