CPSC 2150
Checkers – Project Requirements

Liam Joyce

Requirements Analysis

Functional Requirements:

1. As a player, I want to see the current state of the board before each move so that I can make an informed decision.
2. As a player, I want to input the position and direction of my move so that I can move my piece on the board.
3. As a player, I want the program to check if my input is valid so that I don't make illegal moves.
4. As a player, I want the program to reject moves from non-playable positions so that gameplay rules are enforced.
5. As a player, I want to be notified if a move results in my piece being kinged so that I know my piece can move in all directions.
6. As a player, I want the program to detect when the game is over so that the winner can be declared.
7. As a player, I want to be prompted to play again after a game ends so that I can start a new match if desired.
8. As a player, I want the system to alternate turns automatically so that gameplay proceeds correctly.
9. As a player, I want to see errors for invalid positions or moves so that I can re-enter valid inputs.
10. As a player, I want to only be allowed one jump per turn so that the simplified game rules are followed.

Non-Functional Requirements:

1. The system must not crash on invalid input (error handling).
2. The game interface should respond promptly to player input (performance).
3. Only CheckersFE should handle input/output).
4. The codebase must follow best practices including no magic numbers, proper encapsulation, and information hiding.
5. The design must allow for future feature expansion (e.g., rule changes or UI updates).
6. The game must use consistent naming conventions and be easy to maintain.
7. UML diagrams must be electronically created using diagrams.net.
8. The code must be written in Java 17
9. The game must be runnable on Windows and Linux systems
10. The input/output must be done via the system terminal
11. Board size must be 8x8
12. Player 1 (x) must go first