Task:

You are required to develop a computer program in C++ that allows two different algorithms to compete against each other in a game of suicide checkers.

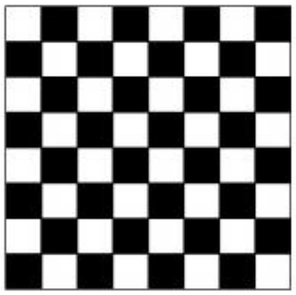
Background:

Suicide checkers, also known as Anti-Checkers, Giveaway Checkers and Losing draughts is a two player variant of the board game checkers where the goal outcome has been reversed - if you have no pieces left or if you cannot move anymore, you win. Players take alternating turns, moving their pieces according to the rules of checkers. The goal of the game is to have no pieces left.

More Info:

In the standard 8×8 version of the game, player 1 has 12 white pieces and player 2 has 12 black pieces. The starting configuration of the board has all pieces of each player lined up on the black spaces in the first three rows on either side of the board (see Figure 1b).

For checkers of varying even board size, we can have boards as shown in Figure 2 where we always have two blank rows in the center of the board. As can be seen the number of pieces changes for different board sizes. For a board size of 6×6, each player has 6 pieces. For a board size of 8×8, each player has 12 pieces. For a board size of 10×10, each player has 20 pieces. For a board size of 12×12, each player has 30 pieces.


(a) Checkers board.

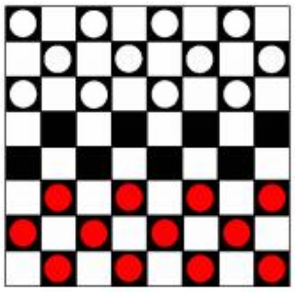
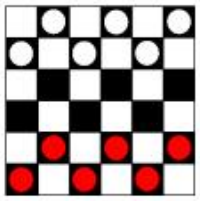
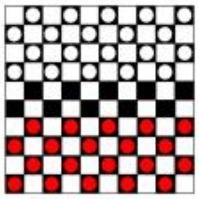

(b) Starting configuration.

Figure 1: Standard 8×8 checkers board.


(a) 6×6 checkers board.


(b) 10×10 checkers board.

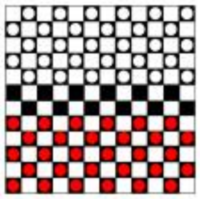

(c) 12×12 checkers board.

Figure 2: Starting configurations for various board sizes.

The easiest way to become accustomed with the rules of the game is to play it yourself. Visit `https://cardgames.io/checkers/` to play an online version.

A summary of the rules is provided below:

- Moves are only allowed on the black squares of the board, so pieces always move diagonally. Single pieces may only move forward toward the opponent.
- A piece making a non-capturing move (not involving a jump) may move only one square.
- A piece making a capturing move (a jump) jumps over one of the opponent's pieces, landing in a straight diagonal line on the other side of the opponents piece. Only one piece may be captured in a single jump; however, multiple jumps are allowed during a single turn.
- After a piece is captured, it must be removed from the board.
- If a player is able to make a capture, this move is compulsory; the jump must be made. If more than one capture is available, the player is free to choose whichever he or she prefers.
- Kings (bonus marks - not required):
  - When a piece reaches the furthest row from the player who controls that piece, it is crowned and becomes a king. One of the pieces which had been captured is placed on top of the king so that it is twice as high as a single piece.

    - Kings are limited to moving diagonally but may move both forward and backward. (Remember that single pieces, i.e. non-kings, are always limited to forward moves.)
    - Kings may combine jumps in several directions, forward and backward, on the same turn.

- If you choose not to implement the play of kings, pieces will be unable to move forwards once they reach the end of the board.
- Single pieces may shift direction diagonally during a multiple capture turn, but must always jump forward (toward the opponent).
- The game ends when one player has lost all their pieces and is crowned the winner.
- The game may also end when there is a stalemate (a draw) and both players have pieces remaining on the board but are unable to make any additional moves.

In this project you are required to

1. **Implement checkers for a board size of n x n where $6 \leq n \leq 12$ and n is always even.**
2. **Implement two different algorithms that will play suicide checkers against each other e.g. an algorithm that randomly chooses a valid move may be one of your algorithms.**
3. **Implementing the play of a king piece (see rules above) will be considered as a bonus and not a requirement. Pieces that reach the end of the board will therefore be unable to move as no forwards moves will be possible at this point.**

This project will expose you to many key aspects of engineering and software development. The following are two of the most important aspects:

- The (software) engineering lifecycle: problem identification, requirements analysis, design, implementation, testing and verification, documentation and maintenance.
- Project planning and time management.

Take time to research and find out more about the above.