Link - https://github.com/sagar-2000/softwareEngAssignment2/tree/stable

Software Engineering project
Assignment 2

By - Sagar Mahajan
Student id - 19204052

Part 1 initialisation -

First I initialised the bad for the game of focus by creating bunch of structs consisting of all the necessary information that will be needed while the game is played. I initialised the player and the struct elements were the name, colour chosen by the player, own pieces that will increase as the fallen pieces are collected by the player, acquired pieces which are the fallen pieces but not the same colour as the players. After that to initialise the board, I had set different cases for the squares of the board including invalid, empty, green and red and then used loops to set the board into the starting position.

Part 2 moves -

for the stacks to be moved from one position to the other I first ask the players the row and column of the stack they want to move then according to the number of pieces on that stack I ask the player whether they want to move the stack up, down, left or right. When they enter the moves new row and column is created which is where they want to move the stack. To push one stack to another I used pointers to the linked list . First I pointed to the last piece of the stack1 and the first piece of the stack2. Then I added them together and stored it in the new row and col. if the player has a own piece he/she has the voice to place it on top of any valid square. I did this using the push function. The player can select the row and col they want to place the piece on and then a pointer is made to point on top of that stack. Then a new space is created by the malloc statement where the new piece is stored.

Part3 5 pieces MAX in a stack

I Made a function to check if the pieces in the stack made is greater than 5. If there are more than 5 pieces, I remove the pieces after that. I do that using pointers and a new space of size equal to 5 pieces(as thats the max that can be removed from a stack). I made 2 pointers pointing to the top of the stack. Then I loop them to the fifth piece. Then I set the next piece in the first pointer to NULL. This became my final list after removing the extra pieces. Then the other pointer stores the pieces removed from the list to the new space created so that they can be checked and stored in the players own_pieces if they are of the same colour.

Part4 Winning situation

To check if one of the players has won the game I loop through the board before every move and check the top pieces of the stack. If they are of the same colour that means a player has won. I keep a track of this using the integers called red_won and green_won.