# Focus-Assignment 2

Sarah Duncan 19352911

Link https://github.com/sarah-duncan/SoftwareEngineeringAssignment2

**Game**

This program is a simulation of the board game Focus. The game consists of two players, red and green. They each have 18 pieces arranged on a 8X8 board with the 3 squares in each corner being blocked off. The aim of the game is to control the board by preventing the other player from making any moves. All stacks have a maximum of five pieces if another piece is put on top then the last piece of the stack is removed. If this piece is the same colour as the colour of the person controlling the stack then the piece can be put in reserve. This piece can later be placed on the board.

**Main**

Main is used mainly for calling additional functions, but it also is used to declare the players array and board . The players and board is initialised via function. Then it goes through a while loop representing a turn. This loop is only broken when the winning condition is met. Inside the while loop the user is told which player is to go next. The board is printed using a function. The coordinates array is declared and is filled with junk values. The players choice is asked using another function. The options are check, move, or place. Check is never used in the main as you can use it unlimited times therefore only move, or place are ever brought back to main. The function choose_coordinates allows users to choose their coordinates, then depending on which option the player chose the if /else statement either allows the player to move or to place pieces. Afterwards check_stack is used to see if the players actions have resulted in a stack greater than 5. If so assigns them to either a placeable piece or a lost piece. It then prints these numbers to the console. Then the function win_condition checks if the win condition have been met. If the win_condition has been reached then win is changed to true. Then bool turn is changed to !turn. If the win condition is met then the winners name is printed. Due to bool turn being changed the turn is inverted so if turn is true and then green wins if turn is false red wins. Then the program ends.

**game_init.h**
game_init.h was given however some changes were made to it. A pointer version of piece called *piece_node_ptr it is used to make connecting stacks easier. Name functionality was added to initialize_player. initialize_board, set_invalid, set_empty, set_green, set_red was not changed. connect is a new function which calls connecting_stacks and then print_stacks. connecting_stacks connects two stacks. It does this by going through the stack and taking note of the colour of a piece then when it reaches the end it goes back using recursion adding each piece on to the new stack. print_stack prints the stack.

**input_output.h**
input_output.h was given but it has been altered. print_board was altered to change the graphics slighty. An index was added and ┼ was used for intersections.The number of pieces in each square was also added. The function bool first_turn chooses which player goes first.

**movement.h**

movement.h is a library that was created to enable pieces to move and be placed. movement() explains wasd to the user and declares new_coordinates[2]. check_coordinates() which calculates where the piece should move as the piece is moved using wasd. The new coordinates are told to the user. The number of pieces from each square are added together. connect() is called to connect the two stacks. then set_empty is used to set the old square to empty. check_stack is called in the case that the stack gets above 5 and adds to the placeable and captured pieces. The board is printed, and the new coordinates becomes the coordinates. place () allows the player to place a piece anywhere on the board if there is a space. limits() is used in check_coordinates it checks that the coordinates chosen are actually on the board.

**pieces.h**

pieces.h contains check_stack() which pops pieces of the stack and reflects it in the player's statistics.

**turns.h**

turns.h contains all functiond used to create a 'turn'. choose_move_type() asks the user weather they want to move or place a piece. It also checks which actions can be preformed. choose_coordinates() is used in other functions to choose the coordinates it calles limits() to check that the coordinates are not out of bounds. check_colour is used to check who owns a particular stack.

**win.h**

win.h contains functions that check for the winning conditions. win_condition() checks if any of the winning conditions have been met.Iterates through all the top pieces in the stack and checks that there is at least one green piece and at least one red piece on the top. It returns either 1 or 0, 1 meaning there is a red and a green piece, 0 means that at least one is missing.