Caro Game

Caro is a popular turn-based two-player game that is played on a grid, typically 15x15 or 19x19. The objective of the game is to be the first player to align five of their symbols (either X or O) consecutively in a row, which can be horizontal, vertical, or diagonal.

In this assignment, your group will design and implement the classic board game Caro using the C programming language. This will enhance your understanding of fundamental programming concepts such as arrays, functions, loops, and user input handling while fostering teamwork and collaboration.

Your program should include the following key functions:

- Initialize the Board: The user will choose a different level Set up the game board as a 2D array. You can implement a console-based or graphic-based board.
- Play the Game: Manage the turn-taking process, allowing players to enter their moves, validating the input, and updating the board accordingly.
- Check for Win, Tie Conditions: Determine if the current player has achieved five consecutive symbols in any direction (horizontal, vertical, or diagonal) to declare a winner; verify if all cells on the board are filled without a winner, resulting in a tie.
- Save and Load Game State: Provide functionality to save the current game state, including the board configuration and player scores, allowing players to resume later; allow players to load a previously saved game state to continue from where they left off.
- Save Scores: Implement a feature to save player scores after each game, allowing for tracking of wins, losses, and ties over multiple sessions.
- Show Top Players: Show the top 10 players grouped by the board size.
- Display Instructions: Provide clear instructions on how to play the game, including how to enter moves and the rules of play.

caro_game_template.c

```c
#include <stdio.h>
#include <stdlib.h>

void print_menu();
int make_choice(int min, int max);
void play_game();
void load_game();
void show_top_players();


int main(void)
{
    void (*funs[4])(void) = {NULL, &play_game, &load_game, &show_top_players};
    int choice;

    do
    {
        choice = make_choice(0, 3);
        if(choice)
        {
            funs[choice]();
            printf("\nPress any key to return to the menu.");
            getchar();
        }
    }
    while(choice);

    return 0;
}


void print_menu()
{
    system("clear"); // for macos, linux
    // system("cls"); // for windows
    printf("CARO GAME\n");
    printf("-------------------\n");
    printf("1. Play game\n");
    printf("2. Load game\n");
    printf("3. Show top players\n");
    printf("0. Exit\n");
}


int make_choice(int min, int max)
{
    int not_valid = 1, num, scanf_ret;
    char c;

    print_menu();
    fflush(stdin);
    do
    {
        printf("\nEnter your choice: ");
        scanf_ret = scanf("%d%c", &num, &c);
        if(scanf_ret < 2 || c != '\n')
        {
            print_menu();
            printf("\nYour choice is not valid. Please try again!\n");
```

```
60              fflush(stdin);
61          }
62          else if(num < min || num > max)
63          {
64              print_menu();
65              printf("\nYour choice is not valid. Please try again!\n");
66          }
67          else
68              not_valid = 0;
69      } while(not_valid);
70
71      system("clear");
72
73      return num;
74  }
75
76
77  void play_game()
78  {
79      printf("Write your code here to implement the play_game() function.\n");
80  }
81
82
83  void load_game()
84  {
85      printf("Write your code here to implement the load_game() function.\n");
86  }
87
88
89  void show_top_players()
90  {
91      printf("Write your code here to implement the show_top_players() function.\n");
92  }
```