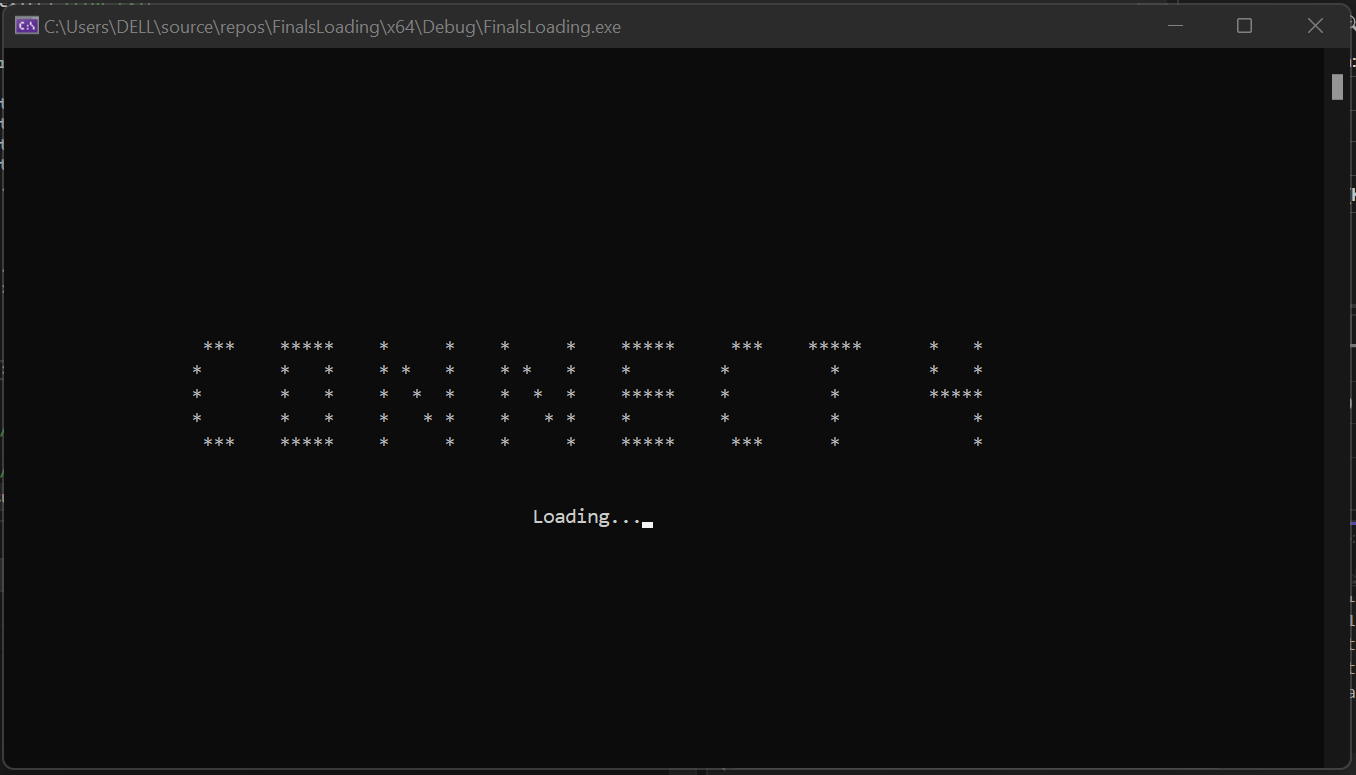
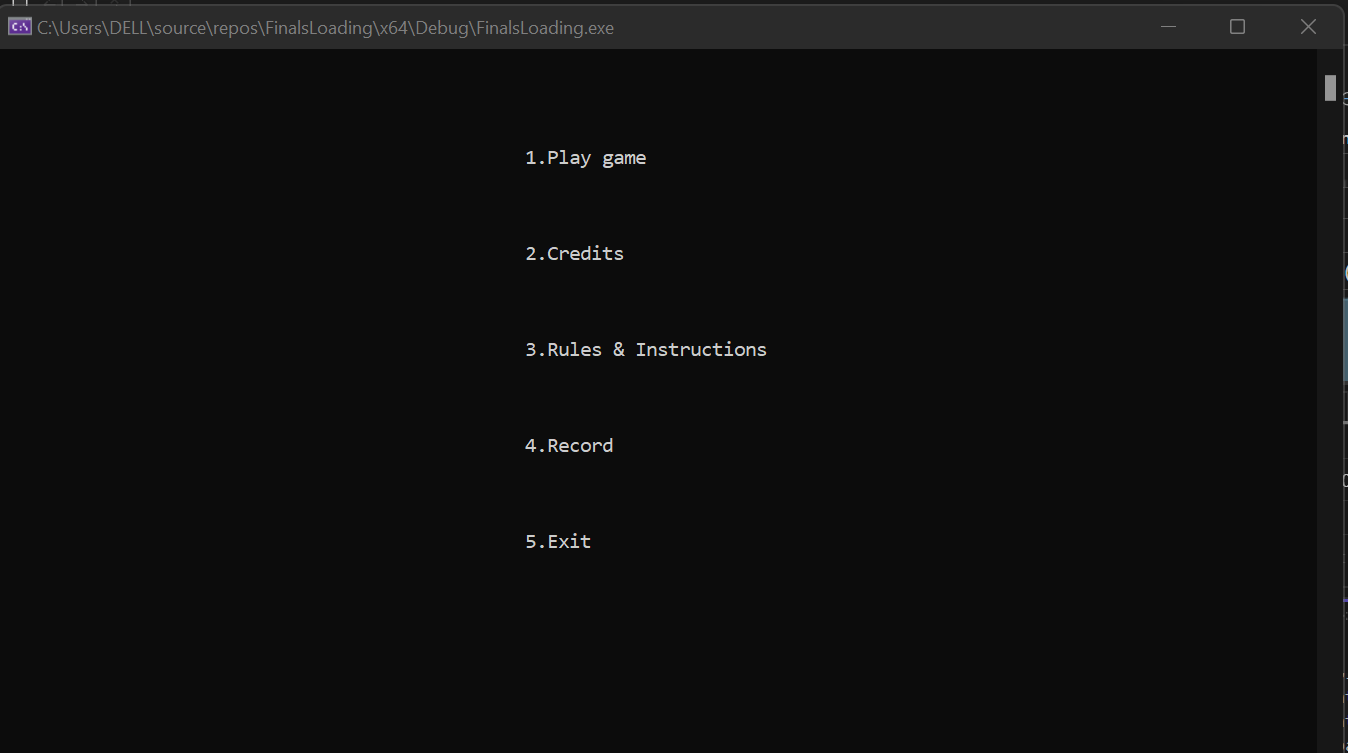
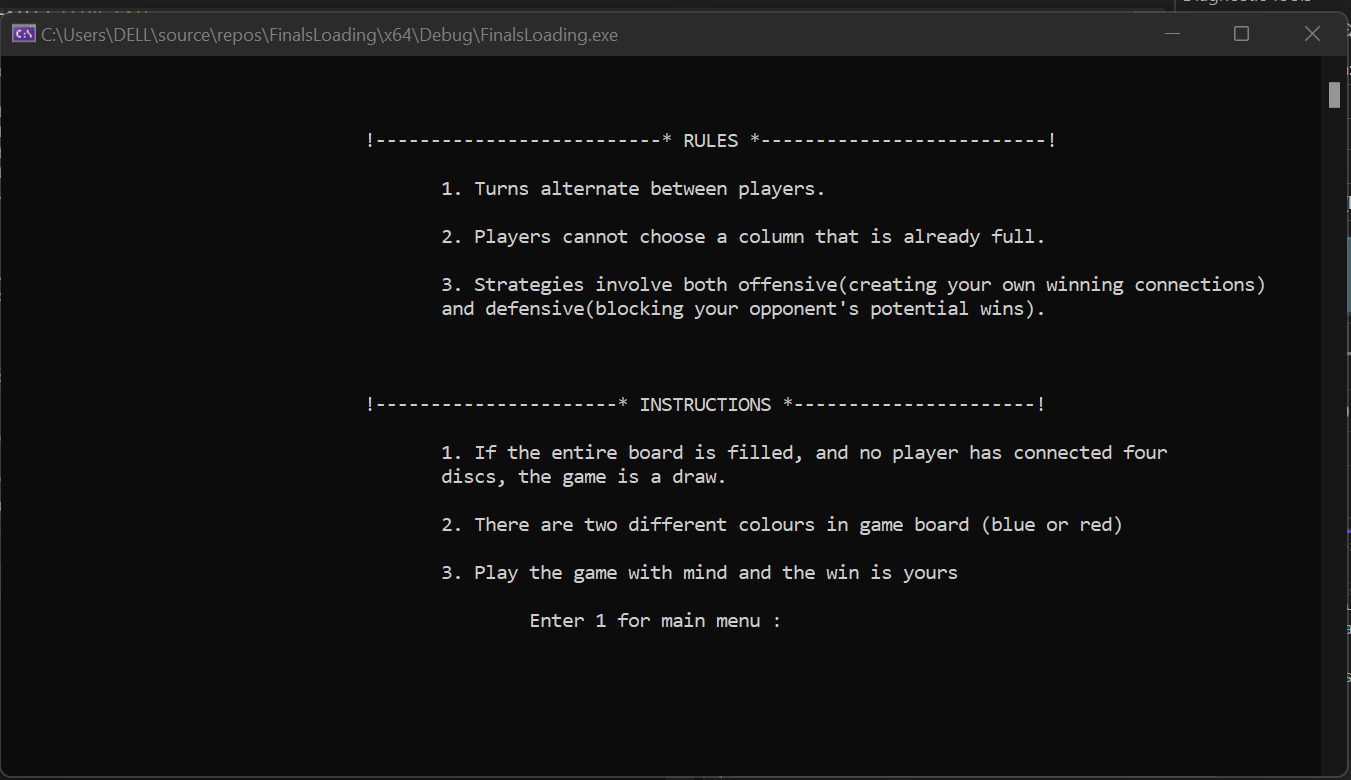
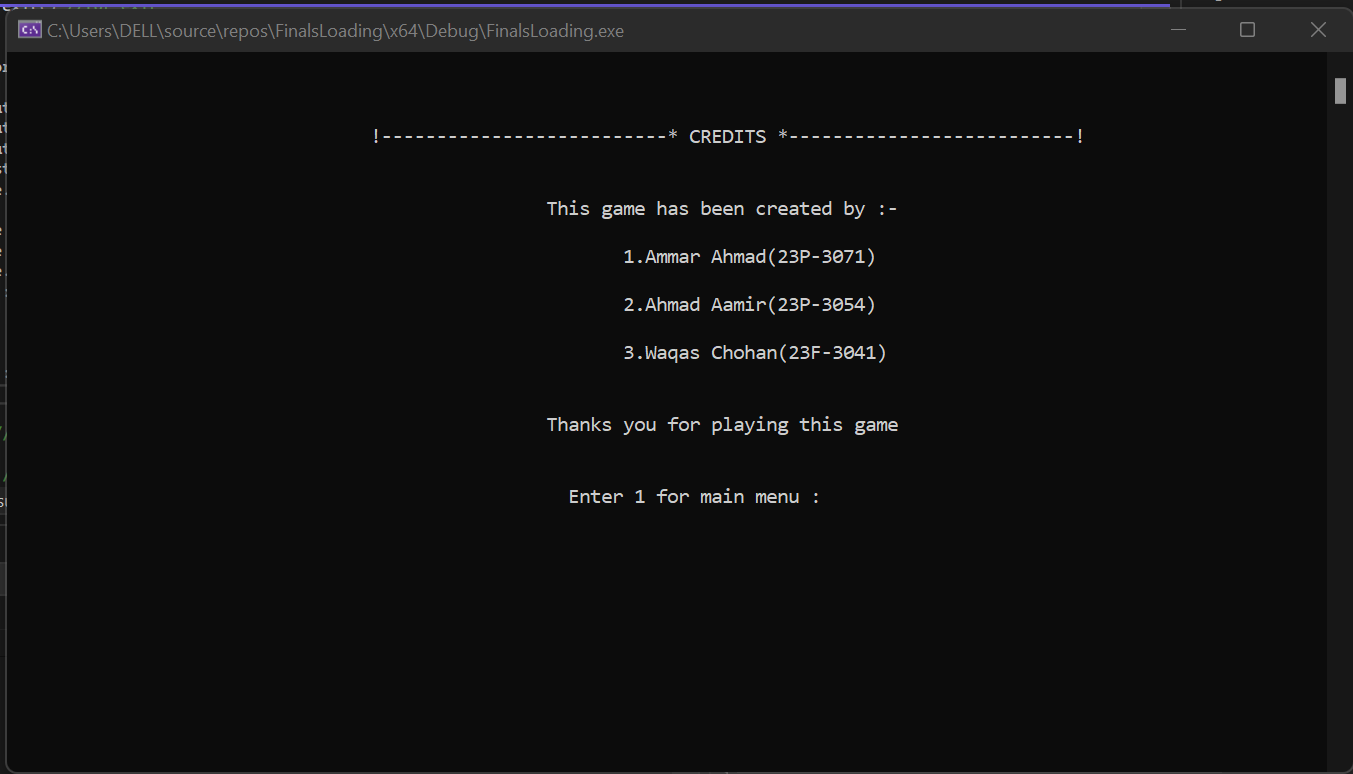
***OUTPUT SCREENSHOTS:***

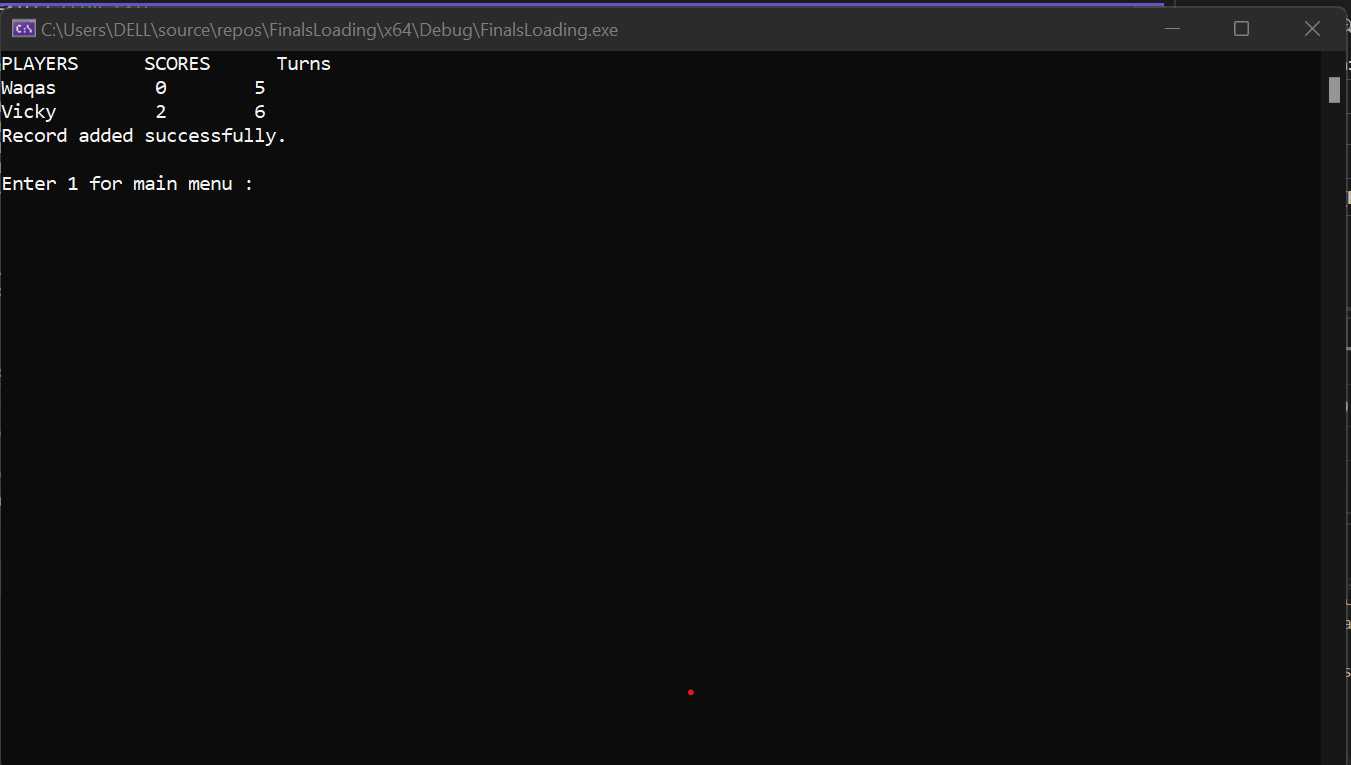
******

******

******

******

******

******

***GAME CODE:***

/\*Connect Four (also known as Four Up, Plot Four, Find Four, Four in a Row, Four in a Line, Drop

Four, and Gravitrips in the Soviet Union) is a two - player connection board game, in which the

players choose a color and then take turns dropping colored discs into a seven - column, six - row

vertically suspended grid.The pieces fall straight down, occupying the lowest available space

within the column.The objective of the game is to be the first to form a horizontal, vertical, or

diagonal line of four of one's own discs. Connect Four is a solved game. The first player can

always win by playing the right moves.\*/

#include<iostream>

#include<string>

#include<windows.h>

#include<conio.h>

#include<cstdlib>

#include<ctime>

#include<fstream>

void refresh();//using this function to refresh the screen after every move

void conditions(std::string, int, std::string, int, int);//using this function to check the conditions after every move of player 1

void conditions1(std::string, int, std::string, int, int);//using this function to check the conditions after every move of player 2

void player\_turn(int, std::string, std::string);//using this function for getting the player turn

void start(int);//using this function for asking the name and toss choice

void display\_board(std::string, std::string);//using this function to display the board

void exit();//using this function to exit the program

void fullboard(std::string, std::string);//using this function to check if it is draw

void cases(int, std::string, std::string, int, int, int, int);//using this function to set program according to user choice

void columadd(int);//using this function to add the turn in board of player 1

void columadd2(int);//using this function to add the turn in board of player 2

void menu(bool, std::string, std::string, int, int, int, int);//for asking the user choice

void reset();//for resetting the board

void addRecord(std::string, std::string, int, int, int, int);//using this function to show user the game record

char board[6][7] = { { ' ',' ',' ',' ',' ',' ',' ' },

{ ' ',' ',' ',' ',' ',' ',' ' } ,{ ' ',' ',' ',' ',' ',' ',' ' },

{ ' ',' ',' ',' ',' ',' ',' ' },{ ' ',' ',' ',' ',' ',' ',' ' },

{ ' ',' ',' ',' ',' ',' ',' ' } };//declaring an char type array

int r[7] = { 0 };

int count1 = 0;

int score = 0, score1 = 0;

int main()

{

bool h;

std::string b = "Player 1";

std::string c = "Player 2";

std::cout << "\n \n \n \n \n \n \n \n \n \n \n \n";

//displaying connect 4 logo

std::cout << "\t\t \*\*\* \*\*\*\*\* \* \* \* \* \*\*\*\*\* \*\*\* \*\*\*\*\* \* \* " << std::endl;

std::cout << "\t\t \* \* \* \* \* \* \* \* \* \* \* \* \* \* " << std::endl;

std::cout << "\t\t \* \* \* \* \* \* \* \* \* \*\*\*\*\* \* \* \*\*\*\*\* " << std::endl;

std::cout << "\t\t \* \* \* \* \* \* \* \* \* \* \* \* \* " << std::endl;

std::cout << "\t\t \*\*\* \*\*\*\*\* \* \* \* \* \*\*\*\*\* \*\*\* \* \* " << std::endl;

//using the system sleep function for loading

std::cout << "\n\n\t\t\t\t\t\tLoading";

std::cout << ".";

Sleep(850);

std::cout << ".";

Sleep(850);

std::cout << ".";

Sleep(900);

h = 1;

menu(h, b, c, 0, 0, 0, 0);//calling the menu function

system("pause");

return 0;

}

void menu(bool h, std::string b, std::string c, int scores, int scores1, int co, int co1)//function definition

{

int a;

if (h == 1)//checking if user wants to play

{

refresh();//for refreshing the screen

std::cout << "\n\n\n\n\t\t\t\t\t\t";

std::cout << "1.Play game\n";//showing the menu

std::cout << "\n\n\n\t\t\t\t\t\t";

std::cout << "2.Credits\n";

std::cout << "\n\n\n\t\t\t\t\t\t";

std::cout << "3.Rules & Instructions\n";

std::cout << "\n\n\n\t\t\t\t\t\t";

std::cout << "4.Record\n";

std::cout << "\n\n\n\t\t\t\t\t\t";

std::cout << "5.Exit\n";

std::cout << "\n\n";

std::cin >> a;

cases(a, b, c, scores, scores1, co, co1);//calling the case function

}

else

exit();//if any other input then exit

}

void cases(int t, std::string p1, std::string p2, int scores, int scores1, int co, int co1)//for showing according to user choice

{

int xy, yz, za;

switch (t)

{

case 1:

start(t);//for play game

break;

case 2:

system("cls");//adding credits portion to the game

std::cout << "\n\n\n\t\t\t\t !--------------------------\* CREDITS \*--------------------------!";

std::cout << "\n\n\n\t\t\t\t\t\t This game has been created by :-\n\n\t\t\t\t\t\t\t1.Ammar Ahmad(23P-3071)\n\n\t\t\t\t\t\t\t2.Ahmad Aamir(23P-3054)\n\n\t\t\t\t\t\t\t3.Waqas Chohan(23F-3041)";

std::cout << "\n\n\n\t\t\t\t\t\t Thanks you for playing this game";

std::cout << std::endl;

std::cout << "\n\n\t\t\t\t\t\t ";

std::cout << "Enter 1 for main menu : ";

std::cin >> xy;

if (xy == 1)

{

menu(1, p1, p2, scores, scores1, co1, co);

}

else

{

std::cout << "Exit";

exit(0);

}

break;

case 3:

system("cls");//adding rules and instruction portion

std::cout << "\n\n\n\t\t\t\t !--------------------------\* RULES \*--------------------------!";

std::cout << "\n\n\t\t\t\t\t1. Turns alternate between players.\n\n\t\t\t\t\t2. Players cannot choose a column that is already full.";

std::cout << "\n\n\t\t\t\t\t3. Strategies involve both offensive(creating your own winning connections) \n\t\t\t\t\tand defensive(blocking your opponent's potential wins).";

std::cout << std::endl;

std::cout << "\n\n\n\t\t\t\t !----------------------\* INSTRUCTIONS \*----------------------!";

std::cout << "\n\n\t\t\t\t\t1. If the entire board is filled, and no player has connected four \n\t\t\t\t\tdiscs, the game is a draw.";

std::cout << "\n\n\t\t\t\t\t2. There are two different colours in game board (blue or red)";

std::cout << "\n\n\t\t\t\t\t3. Play the game with mind and the win is yours";

std::cout << "\n\n\t\t\t\t\t\t";

std::cout << "Enter 1 for main menu : ";

std::cin >> yz;

if (yz == 1)

{

menu(1, p1, p2, scores, scores1, co1, co);//calling the menu function

}

else

{

std::cout << "Exit";

exit(0);

}

break;

case 4:

system("cls");

addRecord(p1, p2, scores, scores1, co1, co);//calling the addrecord function and passing the arguments

std::cout << std::endl;

std::cout << "Enter 1 for main menu : ";

std::cin >> za;

if (za == 1)

{

menu(1, p1, p2, scores, scores1, co1, co);

}

else

{

std::cout << "Exit";

exit(0);

}

break;

case 5:

exit();//for exit

break;

}

}

void refresh()//function definition

{

system("CLS");

}

void start(int w)//function definition

{

int toss;

std::string n1, n2;

if (w == 1)//if user wants to play the game

{

std::cout << "Enter first player name :";//getting the player name

std::cin >> n1;

std::cout << "Enter your toss choice by(0 or 1) :";//asking about toss choice

std::cin >> toss;

std::cout << "Enter second player name :";

std::cin >> n2;

display\_board(n1, n2);//displaying board

player\_turn(toss, n1, n2);//for player turn according to toss

}

}

void player\_turn(int q, std::string p1, std::string p2)//function definition

{

int count5 = 0, count6 = 0;

int y, x, x1;

srand((int)time(0));//for random toss

int r = rand() % 2;

std::cout << "\n\nToss result is :" << r << std::endl;

if (q == r)//according to toss

{

std::cout << p1 << " turn:In which column do you want to add :";//asking from player 1

std::cin >> y;

columadd(y);//adding in column

display\_board(p1, p2);

fullboard(p1, p2);

count5++;

conditions(p1, y, p2, count5, count6);//checking the conditions

for (int i = 0; i < 42; i++)//according to space in board

{

if (i % 2 != 0)//if remainder is not zero

{

std::cout << p1 << " turn:In which column do you want to add :";

std::cin >> x;

//calling the functions

columadd(x);

display\_board(p1, p2);

fullboard(p1, p2);

count5++;

conditions(p1, x, p2, count5, count6);

}

else//if remainder is zero

{

std::cout << p2 << " turn:In which column do you want to add :";

std::cin >> x1;

//calling the functions

columadd2(x1);

display\_board(p1, p2);

fullboard(p1, p2);

count6++;

conditions1(p2, x1, p1, count6, count5);

}

}

}

else//if second player wins the toss

{

std::cout << p2 << " turn:In which column do you want to add :";

std::cin >> y;

columadd2(y);

display\_board(p1, p2);

fullboard(p1, p2);

count6++;

conditions1(p2, y, p1, count6, count5);

for (int i = 0; i < 42; i++)//according to spaces in board

{

if (i % 2 == 0)//if remainder is zero

{

std::cout << p1 << " turn:In which column do you want to add :";

std::cin >> x;

//calling the functions

columadd(x);

display\_board(p1, p2);

fullboard(p1, p2);

count5++;

conditions(p1, x, p2, count5, count6);

}

else//if remainder is not zero

{

std::cout << p2 << " turn:In which column do you want to add :";

std::cin >> x1;

//calling the functions

columadd2(x1);

display\_board(p1, p2);

fullboard(p1, p2);

count6++;

conditions1(p2, x1, p1, count6, count5);

}

}

}

}//when player 1 enters

void columadd(int x)//function definition

{

int j;

j = x - 1;

for (int i = 6; i >= 0; i--)//when one user turns takes place one column will be filled

{

if (board[i - r[j]][j] == ' ')

{

board[i - r[j]][j] = 'x';

r[j]++;

break;

}

}

}//when player 2 enters

void columadd2(int u)//function definition

{

int j;

j = u - 1;

for (int i = 6; i >= 0; i--)//when one user turns takes place one column will be filled

{

if (board[i - r[j]][j] == ' ')

{

board[i - r[j]][j] = 'o';

r[j]++;

break;

}

}

}

//for displaying board along with player names and signs

void display\_board(std::string p1, std::string p2) {

// Displaying the board with color changes for 'x' and 'o'

system("cls");

HANDLE hConsole = GetStdHandle(STD\_OUTPUT\_HANDLE); // Get console handle

std::cout << "\t\t" << p1 << "[x]\n\t\t" << p2 << "[o]\n";

std::cout << "\t\t\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

std::cout << "\t\t Connect 4 \n";

std::cout << "\t\t\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

std::cout << "\t\t\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\n";

for (int i = 0; i < 6; i++) {

std::cout << "\t\t| | | | | | | |\n";

std::cout << "\t\t| ";

for (int j = 0; j < 7; j++) {

if (board[i][j] == 'x') {

SetConsoleTextAttribute(hConsole, FOREGROUND\_RED); // It changes the colour of x to red.

std::cout << board[i][j];

SetConsoleTextAttribute(hConsole, 15); // Reset color for later use.

}

else if (board[i][j] == 'o') {

SetConsoleTextAttribute(hConsole, FOREGROUND\_BLUE); // It changes the colour of o to blue.

std::cout << board[i][j];

SetConsoleTextAttribute(hConsole, 15); // Reset color

}

else {

std::cout << ' ';

}

std::cout << " | ";

}

std::cout << "\n\t\t|\_\_\_\_\_|\_\_\_\_\_|\_\_\_\_\_|\_\_\_\_\_|\_\_\_\_\_|\_\_\_\_\_|\_\_\_\_\_|\n";

}

}//for resetting the board

void reset()//function definition

{

for (int i = 0; i < 6; i++)

{

for (int j = 0; j < 7; j++)

{

board[i][j] = ' ';

}

}

for (int k = 0; k < 7; k++)

{

r[k] = { 0 };

}

}//for player 1 conditions check

void conditions(std::string p, int tu, std::string pl2, int c, int c1)//function definition

{

int i = 0, count = 0;

bool d;

for (int i = 0; i < 6; i++)//for checking the rows

{

for (int j = 0; j < 7; j++)

{

if (board[i][j] == 'x')

{

count++;

if (count == 4)

{

std::cout << p << " Wins\n";

score++;

std::cout << "Go back to main menu(Enter 1)(or any key to exit)\n";

std::cin >> d;//asking again the user choice

if (d == 1)//if user enters 1 then go to main menu

{

reset();

menu(1, p, pl2, score, score1, c, c1);//calling the function

}

else

exit();//for exit

}

}

else {

count = 0;

}

}

}

for (int i = 0; i < 6; i++)

{

for (int j = 0; j < 7; j++)//for checking the columns

{

if (board[j][i] == 'x')

{

count++;

if (count == 4)

{

std::cout << p << " Wins\n";

score++;

std::cout << "Go back to main menu(Enter 1) (or any key to exit)\n";

std::cin >> d;//again asking the user choice

if (d == 1)

{

reset();

menu(1, p, pl2, score, score1, c, c1);//calling the function

}

else

exit();//for exit

}

}

else {

count = 0;

}

}

}

i = tu - 1;

for (int j = 0; j < 6; j++)//for checking the forward diagonals

{

if (board[j][i] == 'x')

{

count++;

i++;

if (count == 4)

{

std::cout << p << " Wins the game\n";

score++;

std::cout << "Go back to main menu(Enter 1)(or any key to exit)\n";

std::cin >> d;//again asking the user choice

if (d == 1)

{

reset();

menu(1, p, pl2, score, score1, c, c1);//calling the function

}

else

exit();//for exit

}

}

else {

count = 0;

}

}

int k = tu - 1;

for (int j = 0; j < 7; j++)//for checking the backward diagonals

{

if (board[j][k] == 'x')

{

count++;

k--;

if (count == 4)

{

std::cout << p << " Wins the game\n";

score++;

std::cout << "Go back to main menu(Enter 1)(or any key to exit)\n";

std::cin >> d;//again asking the user choice

if (d == 1)

{

reset();

menu(1, p, pl2, score, score1, c, c1);//calling the function

}

else

exit();//for exit

}

}

else {

count = 0;

}

}

}//for player 2 conditions check

void conditions1(std::string p, int tu, std::string pl2, int c, int c1)//function definition

{

int i = 0, count = 0;

bool d;

for (int i = 0; i < 6; i++)//for checking the rows

{

for (int j = 0; j < 7; j++)

{

if (board[i][j] == 'o')

{

count++;

if (count == 4)

{

std::cout << p << " Wins\n";

score1++;

std::cout << "Go back to main menu(Enter 1)(or any key to exit)\n";

std::cin >> d;//again asking the user choice

if (d = 1)

{

reset();

menu(1, pl2, p, score, score1, c, c1);//calling the function

}

else

exit();//for exit

}

}

else {

count = 0;

}

}

}

for (int i = 0; i < 6; i++)//for checking the columns

{

for (int j = 0; j < 7; j++)

{

if (board[j][i] == 'o')

{

count++;

if (count == 4)

{

std::cout << p << " Wins\n";

score1++;

std::cout << "Go back to main menu(Enter 1)(or any key to exit)\n";

std::cin >> d;//again asking the user choice

if (d == 1)

{

reset();

menu(1, pl2, p, score, score1, c, c1);//calling the function

}

else

exit();//for exit

}

}

else {

count = 0;

}

}

}

i = tu - 1;

for (int j = 0; j < 6; j++)//for checking the forward diagonals

{

if (board[j][i] == 'o')

{

count++;

i++;

if (count == 4)

{

std::cout << p << " Wins the game\n";

score1++;

std::cout << "Go back to main menu(Enter 1)(or any key to exit)\n";

std::cin >> d;//again asking the user choice

if (d == 1)

{

reset();

menu(1, pl2, p, score, score1, c, c1);//calling the function

}

else

exit();//for exit

}

}

else {

count = 0;

}

}

int k = tu - 1;

for (int j = 0; j < 6; j++)//for checking the backward diagonals

{

if (board[j][k] == 'o')

{

count++;

k--;

if (count == 4)

{

std::cout << p << " Wins the game\n";

score1++;

std::cout << "Go back to main menu(Enter 1)(or any key to exit)\n";

std::cin >> d;//again asking the user choice

if (d == 1)

{

reset();

menu(1, pl2, p, score, score1, c, c1);//calling the function

}

else

exit();//for exit

}

}

else {

count = 0;

}

}

}//for draw condition

void fullboard(std::string p, std::string pl2)//function definition

{

int d;

count1++;

if (count1 == 42)

{

std::cout << "It is a draw\n";

std::cout << "Go back to main menu(Enter 1)(or any key to exit)\n";

std::cin >> d;//again asking the user choice

if (d == 1)

{

reset();

menu(1, p, pl2, 1, 1, 24, 24);//calling the function

}

else

exit();//for exit

}

}

void addRecord(std::string player1, std::string player2, int scores, int scores2, int co, int co1)//function definition

{

std::cout << "PLAYERS" << " " << " SCORES" << " " << " Turns" << std::endl;

std::cout << player1 << " " << scores << " " << co << std::endl;

std::cout << player2 << " " << scores2 << " " << co1 << std::endl;

std::ofstream file("connect4\_records.txt", std::ios::app); // Open the file in append mode

if (file.is\_open())

{

file << player1 << " " << scores << " " << co << std::endl;

file << player2 << " " << scores2 << " " << co1 << "\n";

file.close(); // Close the file

std::cout << "Record added successfully.\n";

}

else

{

std::cout << "Unable to open file.\n";

}

}

void exit()//function definition

{

exit(0);//for exit

}