CK Productions- Final Code

//Include necessary Proteus Libraries and Sequential Timing Libraries//

#include <FEHLCD.h>

#include <FEHIO.h>

#include <FEHUtility.h>

#include <time.h>

#include <stdlib.h>

//Create a class called disk//

class disk

{

//Include public member functions//

public:

void diskP();

void diskC();

//Inlude private data members for class//

private:

char color;

float x;

float y;

int column;

};

//Declare the 4 User-Defined Function Prototypes//

int offBoard(int x, int y);

int nextSpace(int x, int p[6][7]);

int getWinner(int p[6][7]);

void game(int \*wi, int \*l);

//Start main method//

int main(void)

{

//Create disk object//

disk p1;

//Include float xp and yp for when user touches screen//

float xp, yp;

//Include int variable opt for while loop counter//

int opt = 1;

//Include int wins and loss to keep track of score//

int wins = 0;

int loss = 0;

//Set Background and Font Color//

LCD.Clear();

LCD.SetBackgroundColor(WHITE);

LCD.SetFontColor(BLACK);

//Write title of Game to screen- The Introduction to the Game//

LCD.WriteAt("Connect Four!", 90, 10);

Sleep(2000);

LCD.Clear();

//Display Options Menu to User using while loop to able to Replay Game and menu options//

while (opt == 1)

{

LCD.WriteAt("Rules", 95, 40);

LCD.WriteAt("Play Connect 4!", 86, 70);

LCD.WriteAt("Statistics", 93, 100);

LCD.WriteAt("Credits", 93, 130);

LCD.WriteAt("Quit Game?", 94, 160);

//Get the Location of User's Touch if screen is touched//

if (LCD.Touch(&xp, &yp))

{

//Use if-else structure to display the appropriate screen to user based on user's touch//

if(yp>=30 && yp<=65)

{

LCD.Clear();

LCD.WriteLine("Rules for Connect 4:");

Sleep(1.0);

while(LCD.Touch(&xp, &yp) == 0)

{}

LCD.Clear();

//Write the Rules of the game to Screen//

LCD.WriteLine("The objective of the Game is to get a Row, Column, or Diagnal filled 4 in a row on the 6 by 7 matrix board before the computer does!"

"You will have the ability to move your red disks anywhere while the computer will place its blue disk after your turn. Play Strategically"

"and Good Luck! ");

Sleep(10.0);

LCD.Clear();

}

else if (yp>65&&yp<=90)

{

LCD.Clear();

//Invoke the game method to start the game!//

game(&wins,&loss);

LCD.Clear();

}

else if (yp>90 && yp<=120)

{

//Display the Statistics- Wins and Losses to the screen//

LCD.Clear();

Sleep(500);

LCD.WriteLine("Statistics");

LCD.Write("The number of Wins: ");

LCD.WriteLine(wins);

LCD.Write("The number of Losses: ");

LCD.WriteLine(loss);

while(LCD.Touch(&xp, &yp) == 0)

{}

Sleep(500);

LCD.Clear();

}

else if (yp>120 && yp <= 150)

{

//Display the Credits for the Game to screen//

LCD.Clear();

Sleep(500);

LCD.WriteLine("By CK Productions.");

while(LCD.Touch(&xp, &yp) == 0)

{}

Sleep(500);

LCD.Clear();

}

else if (yp>150 && yp <= 180)

{

//Tell user program is quitting and end while loop condition//

LCD.Clear();

Sleep(500);

LCD.WriteLine("Quitting Connect 4 by CK Inc. Thank You for Playing");

while(LCD.Touch(&xp, &yp) == 0)

{}

Sleep(500);

//End while loop condition opt//

opt = 0;

}

else

{

//Still waiting for Touch from user//

opt = 1;

}

}

}

return 0;

}

//Member function of class disk//

void disk::diskP()

{

LCD.SetFontColor(BLUE);

LCD.FillCircle(0,0,19);

}

//Member function of class disk//

void disk::diskC()

{

LCD.SetFontColor(RED);

LCD.FillCircle(0,0,19);

}

//Create int Function offBoard to check any invalid spaces outside board//

int offBoard(int x, int y)

{

int h = 6;

int w = 7;

//variable x represents the row and y represents the column//

if (y < 0||x<0||y>=w||x>=h)

{

return 0;

}

else

{

return 1;

}

}

//Function checks all rows, columns, and diagonals to find winner using Logical and Relational Operators//

int getWinner(int p[6][7])

{

//Let variable w represent the number of columns//

int w = 7;

//Let variable h represent the number of rows//

int h = 6;

//Checks rows//

for (int y= 0; y < h; y++)

{

for (int x = 0; x < w; x++)

{

//Checks if 4 consecutive rows are occupied by same player//

if(p[y][x] !=0 && p[y][x]==p[y][x+1] && p[y][x]==p[y][x+2] && p[y][x]==p[y][x+3])

{

//Returns which player won (1 represents Player 2 represents CPU)//

return p[y][x];

}

}

}

//Checks columns//

for (int y= 0; y < h; y++)

{

for (int x = 0; x < w; x++)

{

//Checks if 4 consecutive columns are occupied by same player//

if(p[y][x] !=0 &&p[y][x]==p[y+1][x] && p[y][x]==p[y+2][x] && p[y][x]==p[y+3][x])

{

//Returns which player won (1 represents Player 2 represents CPU)//

return p[y][x];

}

}

}

//Checks Diagnals in Both Directions//

for (int y= 0; y < h; y++)

{

for (int x = 0; x < w; x++)

{

for (int d =-1; d<=1;d+=2)

{

//Checks if 4 consecutive diagonals are occupied by same player//

if(p[y][x] !=0 &&p[y][x]==p[y+1\*d][x+1] && p[y][x]==p[y+2\*d][x+2] && p[y][x]==p[y+3\*d][x+3])

{

//Returns which player won (1 represents Player 2 represents CPU)//

return p[y][x];

}

}

}

}

//Continue game if space on board to continue//

for (int y= 0; y < h; y++)

{

for (int x = 0; x < w; x++)

{

if(p[y][x] == 0)

{

return 0;

}

}

}

//Return -1 if the game ends up in a tie//

return -1;

}

//Create function that checks if there is an empty space in a column to place disk from the bottom of board//

int nextSpace(int x, int p[6][7])

{

int h = 6;

for (int y = h-1;y>=0;y--)

{

if (board[y][x] == 0)

{

//Return the row which disk can be placed//

return y;

//Break loop once row is found//

break;

}

}

}

//Create main function game that displays the matrix board allows user to play CPU//

void game(int \*wi, int \*l)

{

//Include float x and y for when user touches screen//

int x, y;

float xp, yp;

int z,t;

float xd, yd;

//Include variable s for width of each rectangle in matrix//

int s = 38;

//Let variable w represent the number of columns//

int w = 7;

//Let variable h represent the number of rows//

int h = 6;

//Declar player as 1//

int player = 1;

int CPU = 2;

//Create 2d matrix board//

int board[6][7];

//Draw the matrix board for a Graphical Interface//

LCD.DrawRectangle(5,5,266,228);

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

{

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

{

//Draw the grid boxes for a complete matrix Connect 4 board//

LCD.DrawRectangle((j\*s)+5, (i\*s)+5, s, s);

}

}

//Initialize all spots on the matrix board to 0 initially//

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

{

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

{

board[i][j] = 0;

}

}

//Start while loop condition as long as no winner in game is found//

while (getWinner(board) == 0)

{

//Store the location of user touch//

while (!LCD.Touch(&xp,&yp))

{}

//Store touch location in dummy variable//

while (LCD.Touch(&xd,&yd))

{}

//Get the column that user wants to insert disk//

x = xp/(s+1);

//Invoke nextSpace function to get row in designated column//

y = nextSpace(x, board);

//Set that row and column location on board to player which equals 1//

board[y][x] = player;

//Use nested for loop to access position on board to fill circle//

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

{

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

{

//Fill the matrix board with Scarlet disk if it is player's position on board//

if (board[i][j] == 1)

{

LCD.SetFontColor(SCARLET);

LCD.FillCircle((j\*s)+24, (i\*s)+24, (s/2));

}

}

}

//Check after players turn if getWinner not 0//

if (getWinner(board) != 0)

{

//If 0 break out of while loop//

break;

}

//Wait for 1 second//

Sleep(1.0);

//Start the seeding srand function//

srand(TimeNow());

//Get a random number from 0 to 6 to represent CPU's column to insert disk-Single Player Capability agianst CPU//

z = rand() % 7;

//Invoke nextSpace function to get row in designated column//

t = nextSpace(z, board);

//Set that row and column location on board to CPU which equals 2//

board[t][z] = CPU;

//Use nested for loop to access position on board to fill circle//

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

{

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

{

//Fill the matrix board with Gray disk if it is CPU's position on board//

if (board[i][j] == 2)

{

LCD.SetFontColor(GRAY);

LCD.FillCircle((j\*s)+24, (i\*s)+24, (s/2));

}

}

}

}

Sleep(1.0);

//Once Get winner function finds a winner or tie game clear screen and display results of game (Additional Features of Game)//

if (getWinner(board) != 0)

{

//Change the background//

LCD.Clear();

LCD.SetBackgroundColor(BLACK);

LCD.SetFontColor(WHITE);

//Write to screen who wins//

if (getWinner(board)==1)

{

LCD.WriteLine("Winner is Player! You Won!");

//Increment wins pointer variable//

\*wi = \*wi + 1;

}

else if (getWinner(board)== 2)

{

LCD.WriteLine("Winner is CPU!");

//Increment losses pointer variable//

\*l = \*l + 1;

}

else

{

LCD.WriteLine("The Game resulted in a tie");

}

}

}