

|  |
| --- |
| PROGRAMMING FUNDAMENTALS |
|  |
| SAMI AHMED,  MARYAM TARIQ, |

# CONNECT 4:

|  |
| --- |
| With using basic C++… To get start this:  With Name of Allah, who is most beneficial and merciful.  Let discuss our project.  I created a game with C++ **“CONNECT 4”** which is totally based on C++. |
| *“Let Our Advance Worrying Become Advance Thinking And Planning.”* |

IN THIS CODE:

*BEFORE starting the code :*

*we are supposed to define this code briefly, we use different libraries for run the code successfully.*

*{*

*void draw\_board(); //function for drawing the pattern of the code o game...*

*void player\_movement(int player); //for both players ... player1 or player 2.*

*bool check\_for\_winner(int x, int y, int player); //either which player is won or not*

*bool check\_diagonal\_combo\_SW\_NE(int x, int y, int player); //for checking diagonal (/)*

*bool check\_diagonal\_combo\_NW\_SE(int x, int y, int player); //for checking diagonal(\)*

*bool check\_vertical(int x, int y, int player); //vertical line of 4 matches (|)*

*bool check\_horizontal(int x, int y, int player); //horizontal line of 4 matches(\_\_)*

*}*

*I used this function for defining our game. this game is named as connect 4. this game is played between 2 players. Both players played and put their stamp on their desired row to win. then I’ll win the game by connecting 4 stamps in a run column and diagonals too.*

*after one on them played successfully and game will end. We also add graphics in this game and Get help from YouTube* [*https://youtu.be/MvX4tVETjHk*](https://youtu.be/MvX4tVETjHk) *for graphics and code from seniors.*

*CODE:*

/\*

BEFORE starting the code :

we are supposed to define this code briefly, we use different libraries for run the code successfully.

{

void draw\_board(); //function for drawing the pattern of the code o game.

void player\_movement(int player); //for both players ... player1 or player 2.

bool check\_for\_winner(int x, int y, int player); //either which player is won or not

bool check\_diagonal\_combo\_SW\_NE(int x, int y, int player); //for checking diagonal (/)

bool check\_diagonal\_combo\_NW\_SE(int x, int y, int player); //for checking diagonal(\)

bool check\_vertical(int x, int y, int player); //vertical line of 4 matches (|)

bool check\_horizontal(int x, int y, int player); //horizontal line of 4 matches(\_\_)

}

I used this function for defining our game. this game is named as connect 4. this game is played between 2 players. Both players played and put their stamp on their desired row to win... they I’ll win the game by connecting 4 stamps in a run column and diagonals too after one on them played successfully game will ended.

we also add graphics in this game and get help from YouTube <https://youtu.be/MvX4tVETjHk> for graphics and code from seniors.

\*/

#include <iostream>

#include <windows.h>

//define commands are called directives, allow the user to give name to the constant value at the start of the code.

#define HEIGHT 6

#define WIDTH 7

using namespace std;

void draw\_board(); //function for drawing the pattern of the code o gamw...

void player\_movement(int player); //for both players ... player1 or player 2.

bool check\_for\_winner(int x, int y, int player); //either which player is won or not

bool check\_diagonal\_combo\_SW\_NE(int x, int y, int player); //for checking diagonal (/)

bool check\_diagonal\_combo\_NW\_SE(int x, int y, int player); //for checking diagonal(\)

bool check\_vertical(int x, int y, int player); //vertical line of 4 matches (|)

bool check\_horizontal(int x, int y, int player); //horizontal line of 4 matches(---)

int board\_info[HEIGHT][WIDTH] = { //declaring height and width as an array...

{ 0,0,0,0,0,0,0 },

{ 0,0,0,0,0,0,0 },

{ 0,0,0,0,0,0,0 },

{ 0,0,0,0,0,0,0 },

{ 0,0,0,0,0,0,0 },

{ 0,0,0,0,0,0,0 }

};

int lastmoveX, lastmoveY;

int main()

{

bool winner = false;

HANDLE h = GetStdHandle(STD\_OUTPUT\_HANDLE);

{

SetConsoleTextAttribute(h, 16); //gives pink colour

cout << "\n\t\t\t\t\tWELCOME TO CONNECT FOUR GAME. " << endl;

}

{

SetConsoleTextAttribute(h, 6); //gives red colour

cout << "\nWHERE DO YOU WANT TO PLACE YOUR STAMP...!\n" << endl;

}

SetConsoleTextAttribute(h, 16); //gives colour for the whole game prints

draw\_board();

for (int i = 0; i < 21; i++) //maximum 21 try for each players because , 6x7=42/2=21.

{

player\_movement(1); //for the winner of player 1

draw\_board();

winner = check\_for\_winner(lastmoveX, lastmoveY, 1);

if (winner)

{

{ SetConsoleTextAttribute(h, 6); //gives blue colour

cout << "\nPlayer 1,YOU WIN....!";

}

break; //terminate from the comiler

}

player\_movement(2);//for the winner of player 2

draw\_board();

winner = check\_for\_winner(lastmoveX, lastmoveY, 2);

if (winner)

{

{

SetConsoleTextAttribute(h, 6); //gives blue colur

cout << "\nplayer 2, YOU WIN....!" << endl;

}

break;

}

}

system("pause");

return 0;

}

void draw\_board() //drawn the proper pattern on the complier

{

HANDLE h = GetStdHandle(STD\_OUTPUT\_HANDLE);

cout << "| 1| 2| 3| 4| 5| 6| 7|" << endl;

for (int y = 0; y < HEIGHT; y++) //for rows

{

for (int x = 0; x < WIDTH; x++) //for columns

{

cout << "| ";

if (board\_info[y][x] == 0) cout << " ";

else if (board\_info[y][x] == 1) cout << "X"; //stamp of player 1.

else if (board\_info[y][x] == 2) cout << "O"; //stamp of player 2.

}

cout << "\n\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_" << endl;

}

}

void player\_movement(int player)

{

int choice;//variable for seescting the column

int counter = 0; //

HANDLE h = GetStdHandle(STD\_OUTPUT\_HANDLE);

{

SetConsoleTextAttribute(h, 6); //gives yellow colour

cout << "\nPLAYER" << player << ",SELECT NUMBER BETWEEN(1-7): ";

}

cin >> choice; //selescted line entered

while (choice > WIDTH || choice <= 0) //enEtered number should be in the range of between (1-7)

{

{

SetConsoleTextAttribute(h, 12); //gives blue colour

cout << "\nYOUR ENTERED NUMBER IS NOT IN THE RANGE(1-7), PRESS AGAIN...! : ";

}

cin >> choice;

}

while (board\_info[(HEIGHT - 1) - counter][(choice - 1)] != 0)

{

counter++;

if (counter > (HEIGHT - 1)) //for checking the completed column

{

{

SetConsoleTextAttribute(h, 10); //gives blue colour

cout << "\nCOLUMN IS COMPLETED, SELECT ANOTHER COLUMN: ";

}

cin >> choice;

counter = 0; //You must reset the number else it will start at the beginning of the array

}

}

player=board\_info[(HEIGHT - 1) - counter][choice - 1] = player;

lastmoveY = (HEIGHT - 1) - counter;

lastmoveX = choice - 1;

}

bool check\_for\_winner(int x, int y, int player)

{

bool winner;

if (check\_diagonal\_combo\_SW\_NE(x, y, player))

{

return true;

}

else if (check\_diagonal\_combo\_NW\_SE(x, y, player))

{

return true;

}

else if (check\_vertical(x, y, player))

{

return true;

}

else if (check\_horizontal(x, y, player))

{

return true;

}

else

{

return false;

}

}

bool check\_diagonal\_combo\_SW\_NE(int x, int y, int player)// (/)

{

int score = 1;

int count = 1;

while ((y - count >= 0) && (x + count < WIDTH))

{

if (board\_info[y - count][x + count] == player) //Check SW to NE

{

score++;

count++;

}

else break; //If no combo is detected break from the loop

}

count = 1;

while ((y + count < HEIGHT) && (x - count >= 0))

{

if (board\_info[y + count][x - count] == player) //Check NE to SW

{

score++;

count++;

}

else break; //If no combo is detected break from the loop

}

if (score == 4)

{

return true;

}

else

{

return false;

}

}

bool check\_diagonal\_combo\_NW\_SE(int x, int y, int player)//(\)

{

int score = 1;

int count1 = 1;

while ((y + count1 >= 0) && (x + count1 < WIDTH))

{

if (board\_info[y + count1][x + count1] == player) //Check NW to SE

{

score++;

count1++;

}

else

{

break; //If no combo is detected break from the loop

}

}

count1 = 1;

while ((y - count1 < HEIGHT) && (x - count1 >= 0))

{

if (board\_info[y - count1][x - count1] == player) //Check SE to NW

{

score++;

count1++;

}

else

{

break; //If no combo is detected break from the loop

}

}

if (score == 4)

{

return true;

}

else

{

return false;

}

}

bool check\_vertical(int x, int y, int player) //for vertical (|)

{

int score = 1;

int count2 = 1;

while (y + count2 >= 0 && y + count2 < HEIGHT)

{

if (board\_info[y + count2][x] == player) //Check dowm verticals

{

score++;

count2++;

}

else break; //If no combo is detected break from the loop

}

int count = 1;

while (y - count2 >= 0 && y - count2 < HEIGHT)

{

if (board\_info[y - count2][x] == player) //Check dowm verticals

{

score++;

count2++;

}

else break; //If no combo is detected break from the loop

}

if (score == 4)

{

return true;

}

else

{

return false;

}

}

bool check\_horizontal(int x, int y, int player) //for horizontal (\_\_)

{

int score = 1;

int count3 = 1;

while ((x + count3 >= 0) && (x + count3 < WIDTH))

{

if (board\_info[y][x + count3] == player) //Check Left most hoorizontal

{

score++;

count3++;

}

else

{

break; //If no combo is detected break from the loop

}

}

count3 = 1;

while ((x - count3 < WIDTH) && (x - count3 >= 0))

{

if (board\_info[y][x - count3] == player) //Check Rightmost horizontal

{

score++;

count3++;

}

else

{

break; //If no combo is detected break from the loop

}

}

if (score == 4) //for connect four

{

return true;

}

else

{

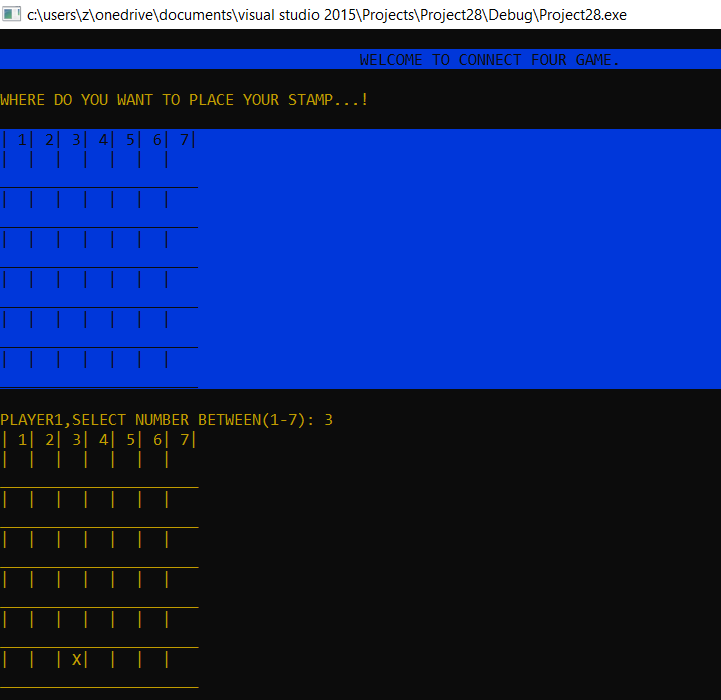
return false;

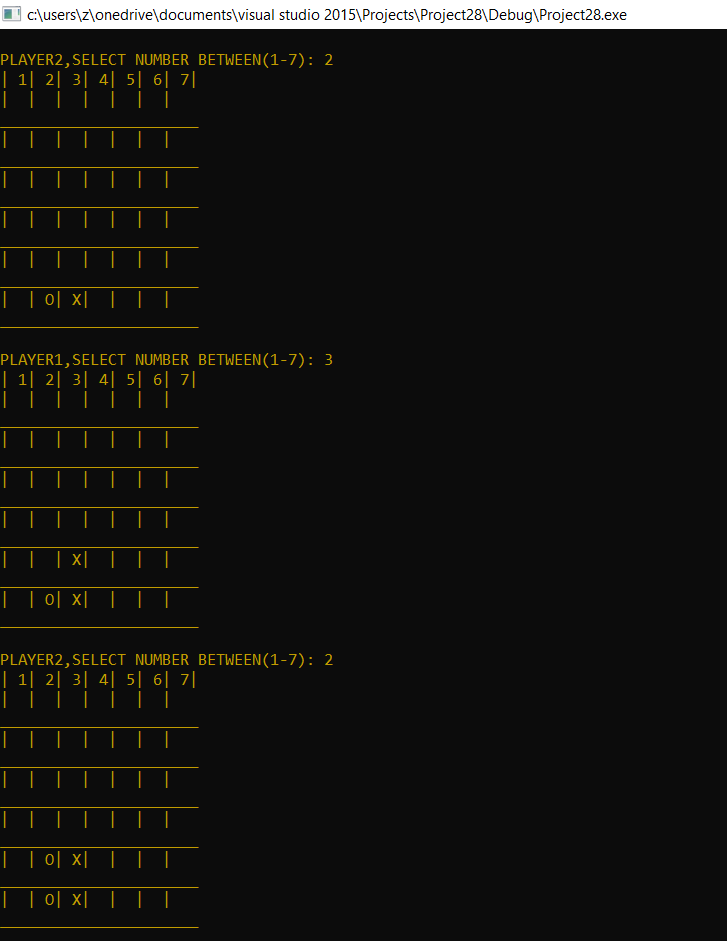
}

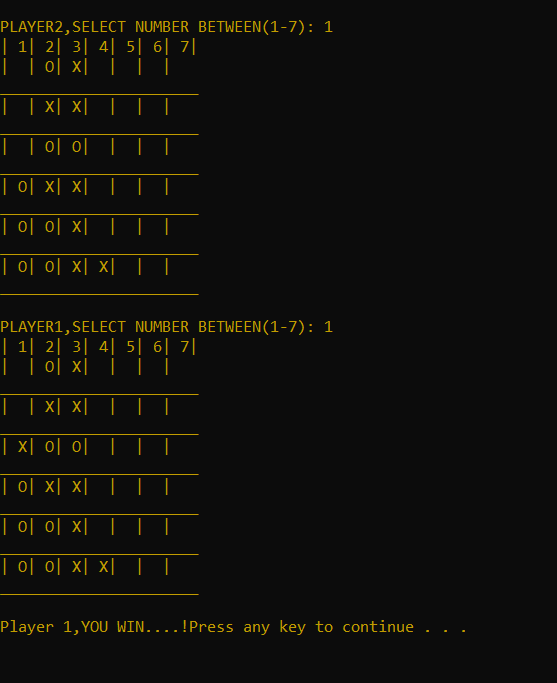
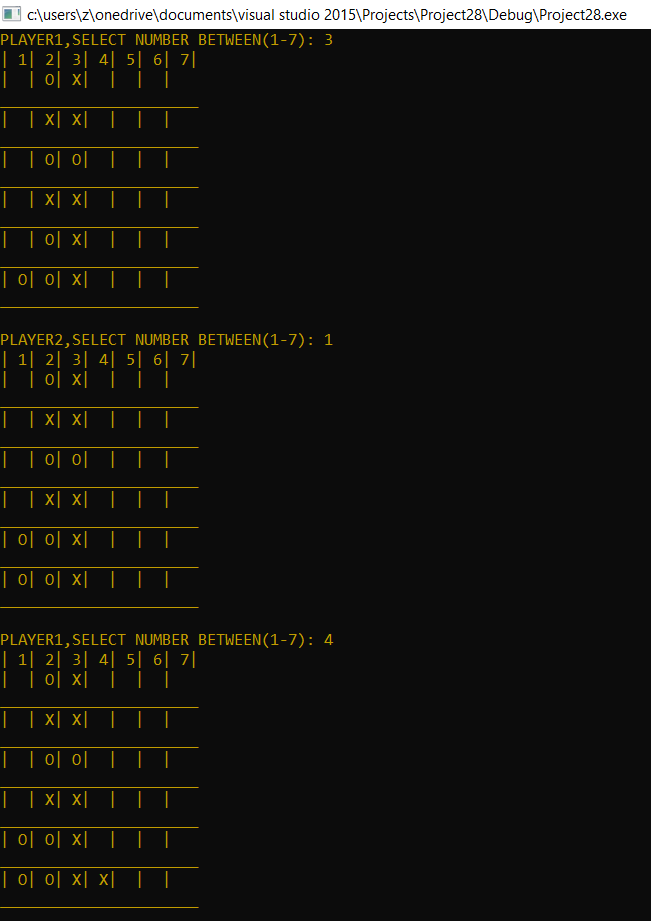
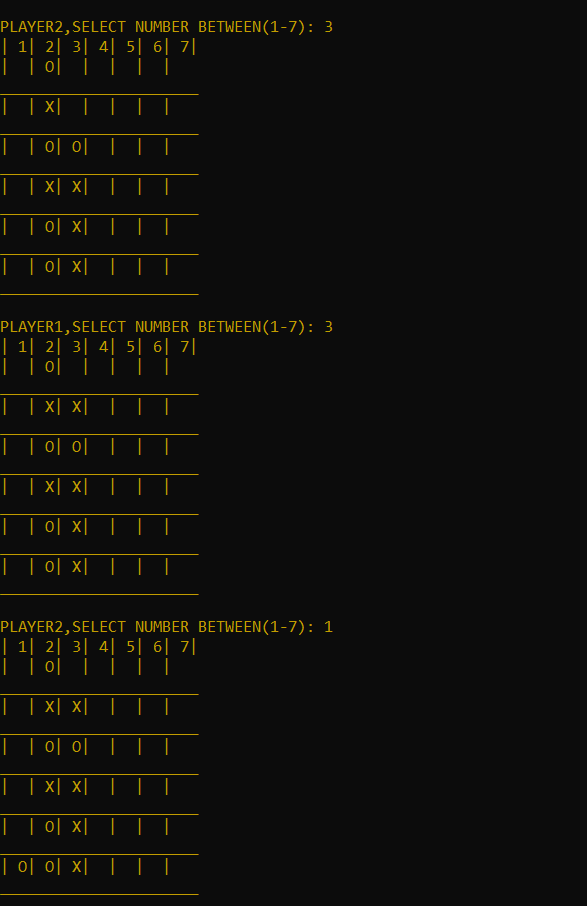
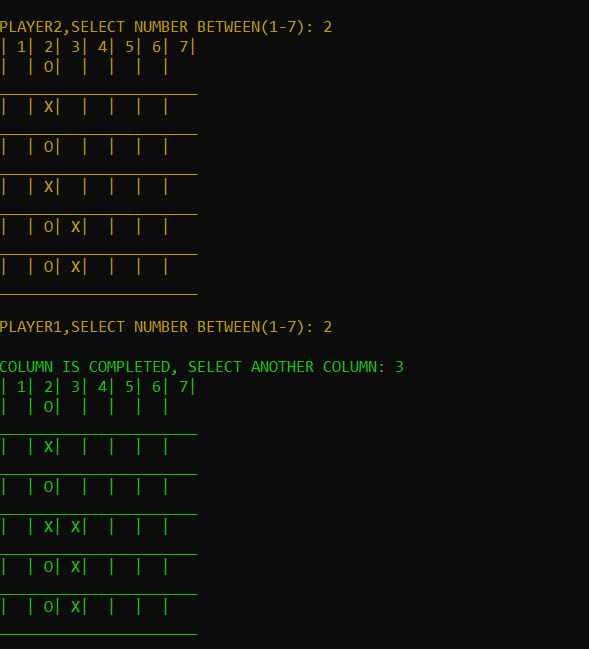
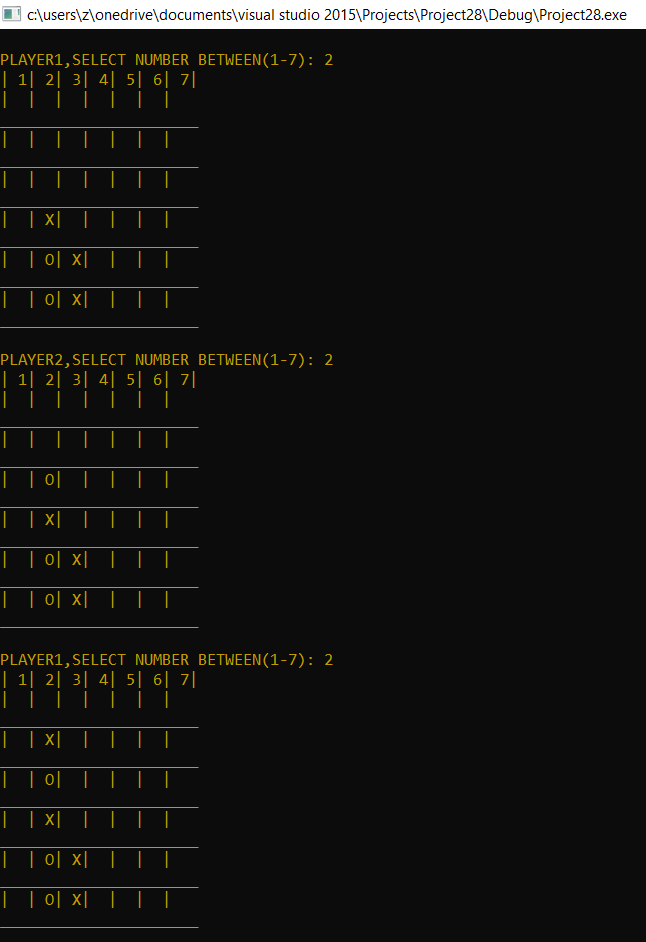
}

/\* end...! \*/

*Output:*







*The end…!*