```cpp
/*
    Zach Hofmeister     1/28/19
    ticTacToe           A game of tic-tac-toe made for two players.
*/

#include "pch.h"
#include <iostream>

using namespace std;

void printTable(char table[3][3]);  //Prints out the game board in it's current state.
char detectWin(char table[3][3]);   //Detects if any winning combination has been played, if the game results in a draw, or if the game can continue.

int main() {
    int turn = 0; //Keeps track of the number of turns played, and which player's turn it is.
    char table[3][3] = { {' ', ' ', ' '}, {' ', ' ', ' '}, {' ', ' ', ' '} }; //Empty tic-tac-toe array
    cout << "Welcome to Tic-Tac-Toe!" << endl;
    cout << "=======================" << endl;

    do { //do-while loop for turns
        cout << endl;
        int row, column; //holds player input
        cout << "Player " << (turn % 2 == 0 ? '1' : '2') << "\'s turn \'" << (turn % 2 == 0 ? 'X' : 'O') << "\'" << endl;
        printTable (table);
        cout << "What row and column would you like? Seperate them with a space: ";
        cin >> row >> column; //Accepts player input for row/column

        if (table[row][column] == ' ') { //If the space is empty, populate it and record the turn.
            table[row][column] = (turn % 2 == 0 ? 'X' : 'O');
            turn++;
        } else { //Loops again if the chosen space is occupied or the player selects a space that is out-of-bounds.
            cout << "Invalid move. Please choose another space." << endl;
        }
    } while (detectWin(table) == ' '); //Continue to loop as long as there is no winner and empty spaces.

    cout << endl <<  "Game over!" << endl;
    printTable(table);
    if (detectWin(table) == 'C') {
        cout << "Cat's game! Nobody wins! Better luck next time!" << endl;
    } else {
```

```
41          cout << "Player " << (detectWin(table) == 'X' ? '1' : '2') << " wins!"
              << endl;
42      }
43
44      return 0;
45  }
46
47  void printTable(char table[3][3]) { //Prints out the tic-tac-toe chart, takes
      the tic-tac-toe array as an input.
48      cout << "\t   0 1 2" << endl;
49      cout << "\t 0 " << table[0][0] << '|' << table[0][1] << '|' << table[0][2]
        << endl;
50      cout << "\t   " << "-----" << endl;
51      cout << "\t 1 " << table[1][0] << '|' << table[1][1] << '|' << table[1][2]
        << endl;
52      cout << "\t   " << "-----" << endl;
53      cout << "\t 2 " << table[2][0] << '|' << table[2][1] << '|' << table[2][2]
        << endl;
54  }
55
56  /*Detects if any winning combination has been played, if the game results in a
      draw, or if the game can continue,
57      and returns that result in the form of a char.*/
58  char detectWin(char table[3][3]) {
59      if (table[0][0] != ' ' && table[0][0] == table[0][1] && table[0][1] ==
        table[0][2]) {
60          return table[0][0];
61      } else if (table[1][0] != ' ' && table[1][0] == table[1][1] && table[1][1]
        == table[1][2]) {
62          return table[1][0];
63      } else if (table[2][0] != ' ' && table[2][0] == table[2][1] && table[2][1]
        == table[2][2]) {
64          return table[2][0];
65      } else if (table[0][0] != ' ' && table[0][0] == table[1][0] && table[1][0]
        == table[2][0]) {
66          return table[0][0];
67      } else if (table[0][1] != ' ' && table[0][1] == table[1][1] && table[1][1]
        == table[2][1]) {
68          return table[0][1];
69      } else if (table[0][2] != ' ' && table[0][2] == table[1][2] && table[1][2]
        == table[2][2]) {
70          return table[0][2];
71      } else if (table[0][0] != ' ' && table[0][0] == table[1][1] && table[1][1]
        == table[2][2]) {
72          return table[0][0];
73      } else if (table[0][2] != ' ' && table[0][2] == table[1][1] && table[1][1]
        == table[2][0]) {
74          return table[0][2];
75      } else { //Winning combo not found.
```

```
76          for (int i = 0; i < 3; i++) {
77              for (int j = 0; j < 3; j++) {
78                  if (table[i][j] == ' ') {
79                      return ' '; //Searches for an empty space, which if found
                        means that the game can continue.
80                  }
81              }
82          }
83          return 'C'; //If an empty space is not found in the for loop above,
              returns 'C' for "cat's game" (draw).
84      }
85  }
86
87  /* EXAMPLE OUTPUT:
88  Welcome to Tic-Tac-Toe!
89  ======================
90
91  Player 1's turn 'X'
92        0 1 2
93      0  | |
94        -----
95      1  | |
96        -----
97      2  | |
98  What row and column would you like ? Seperate them with a space : 1 1
99
100 Player 2's turn 'O'
101       0 1 2
102     0  | |
103       -----
104     1  |X|
105       -----
106     2  | |
107 What row and column would you like ? Seperate them with a space : 0 1
108
109 Player 1's turn 'X'
110       0 1 2
111     0  |O|
112       -----
113     1  |X|
114       -----
115     2  | |
116 What row and column would you like ? Seperate them with a space : 1 0
117
118 Player 2's turn 'O'
119       0 1 2
120     0  |O|
121       -----
122     1  |X|
```

```
123         -----
124      2  |O|
125  What row and column would you like ? Seperate them with a space : 2 1
126
127  Player 1's turn 'X'
128        0 1 2
129      0  |O|
130         -----
131      1 X|X|
132         -----
133      2  |O|
134  What row and column would you like ? Seperate them with a space : 1 2
135
136  Game over!
137        0 1 2
138      0  |O|
139         -----
140      1 X|X|X
141         -----
142      2  |O|
143  Player 1 wins!
144  */
```