Activity No. 5.1	
Hands-on Activity 5.1: Multidimensional Arrays	
Course Code: CPE007	Program: Computer Engineering
Course Title: Programming Logic and Design	Date Performed: September 29, 2025
Section: CPE11S1	Date Submitted: September 30, 2025
Name(s): Tobias, Lawrence C.	Instructor: Engr. Jimlord M. Quejado

6. Output

1.

tic tac keme.cpp shiminet.cpp

```cpp
#include <iostream>
using namespace std;

int main() {
    int table[10][10];

    for (int a=0; a<10; a++) {
        for (int b=0; b<10; b++) {
            table[a][b] = (a+1) * (b+1);
        }
    }

    cout << "Multiplication Table\n";
    for (int a=0; a<10; a++) {
        for (int b=0; b<10; b++) {
            cout << table[a][b] << "\t ";
        }
        cout << endl;
    }

    return 0;
}
```

C:\Dev-Cpp\shiminet.exe

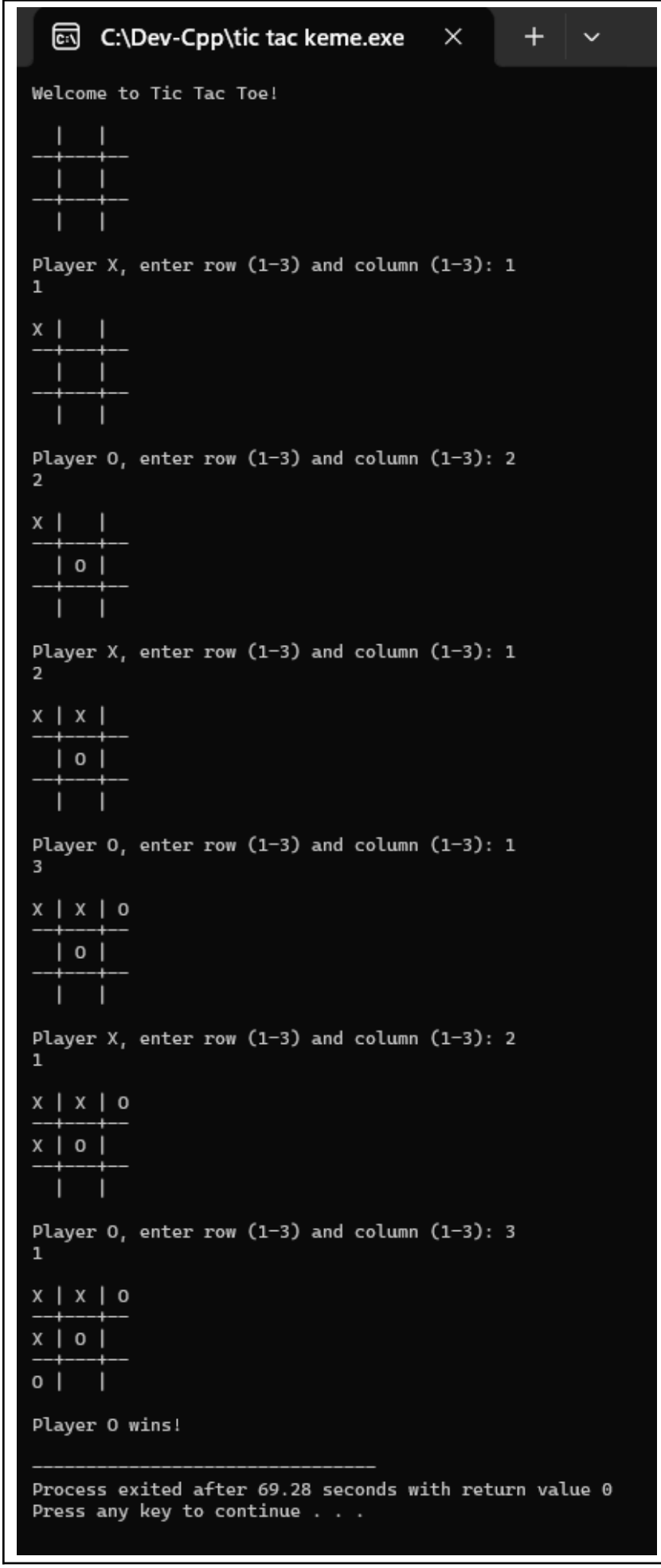
```
Multiplication Table
1       2       3       4       5       6       7       8       9       10
2       4       6       8       10      12      14      16      18      20
3       6       9       12      15      18      21      24      27      30
4       8       12      16      20      24      28      32      36      40
5       10      15      20      25      30      35      40      45      50
6       12      18      24      30      36      42      48      54      60
7       14      21      28      35      42      49      56      63      70
8       16      24      32      40      48      56      64      72      80
9       18      27      36      45      54      63      72      81      90
10      20      30      40      50      60      70      80      90      100

--------------------------------
Process exited after 0.279 seconds with return value 0
Press any key to continue . . .
```

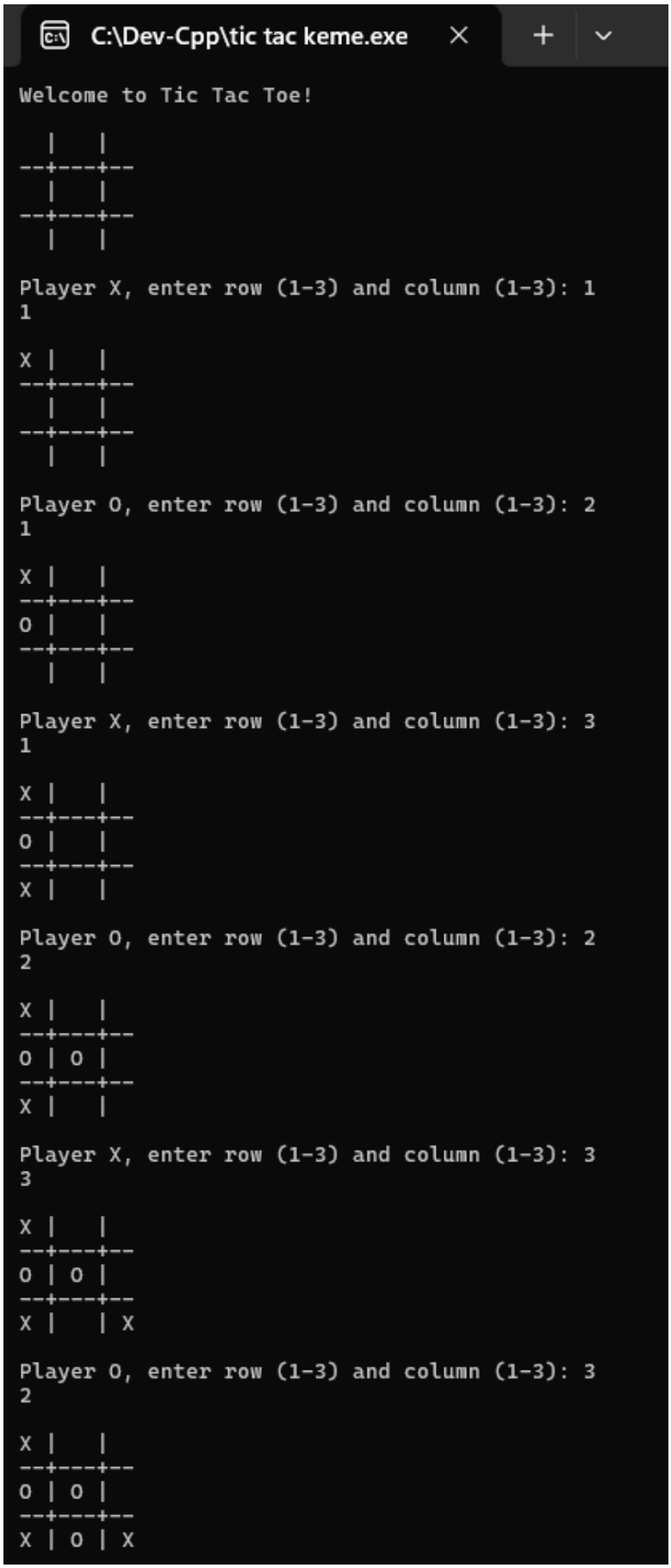
2.

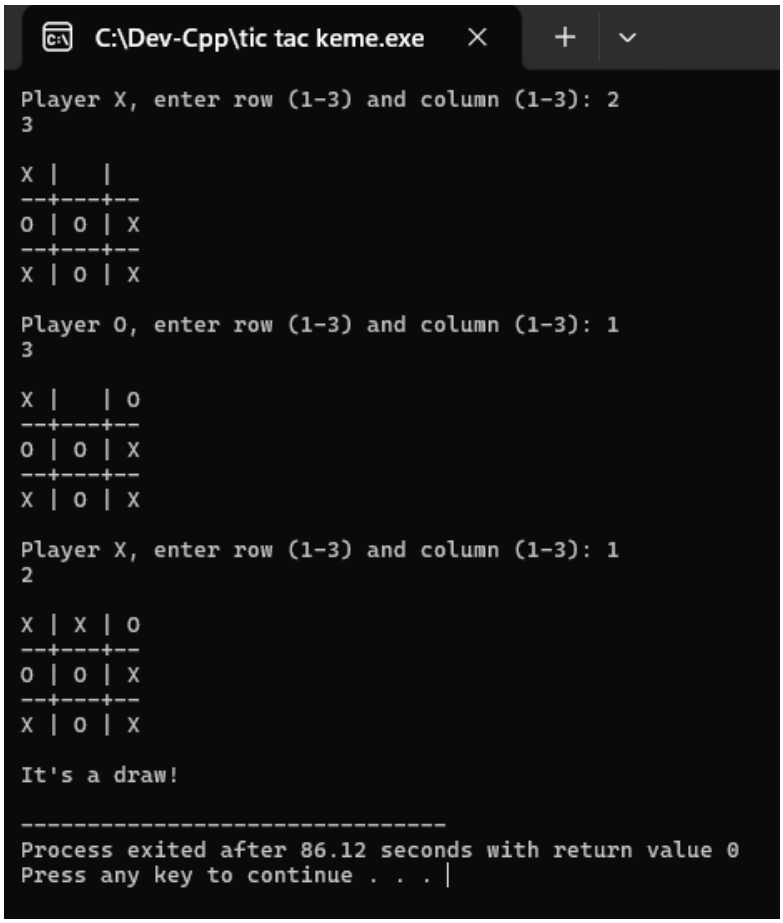
tic tac keme.cpp | shiminet.cpp

```cpp
#include <iostream>
using namespace std;

char board[3][3] = {
    {' ', ' ', ' '},
    {' ', ' ', ' '},
    {' ', ' ', ' '}
};

void printBoard() {
    cout << endl;
    for (int i = 0; i < 3; i++) {
        for (int j = 0; j < 3; j++) {
            cout << board[i][j];
            if (j < 2) cout << " | ";
        }
        cout << endl;
        if (i < 2) cout << "--+---+--" << endl;
    }
    cout << endl;
}

bool checkWin(char player) {

    for (int i = 0; i < 3; i++) {
        if ((board[i][0] == player && board[i][1] == player && board[i][2] == player) ||
            (board[0][i] == player && board[1][i] == player && board[2][i] == player)) {
            return true;
        }
    }

    if ((board[0][0] == player && board[1][1] == player && board[2][2] == player) ||
        (board[0][2] == player && board[1][1] == player && board[2][0] == player)) {
        return true;
    }
    return false;
}
```

```cpp

int main() {
    int row, col;
    char player = 'X';
    int moves = 0;

    cout << "Welcome to Tic Tac Toe!" << endl;
    printBoard();

    while (true) {
        cout << "Player " << player << ", enter row (1-3) and column (1-3): ";
        cin >> row >> col;

        row--; col--;

        if (row < 0 || row > 2 || col < 0 || col > 2 || board[row][col] != ' ') {
            cout << "Invalid move, try again." << endl;
            continue;
        }

        board[row][col] = player;
        moves++;
        printBoard();

        if (checkWin(player)) {
            cout << "Player " << player << " wins!" << endl;
            break;
        }

        if (moves == 9) {
            cout << "It's a draw!" << endl;
            break;
        }

        player = (player == 'X') ? 'O' : 'X';
    }

    return 0;
}

```


C:\Dev-Cpp\tic tac keme.exe
Welcome to Tic Tac Toe!
| |
--+---+--
| |
--+---+--
| |
Player X, enter row (1-3) and column (1-3): 1
1
X | |
--+---+--
| |
--+---+--
| |
Player O, enter row (1-3) and column (1-3): 2
2
X | |
--+---+--
| O |
--+---+--
| |
Player X, enter row (1-3) and column (1-3): 1
2
X | X |
--+---+--
| O |
--+---+--
| |
Player O, enter row (1-3) and column (1-3): 1
3
X | X | O
--+---+--
| O |
--+---+--
| |
Player X, enter row (1-3) and column (1-3): 2
1
X | X | O
--+---+--
X | O |
--+---+--
| |
Player O, enter row (1-3) and column (1-3): 3
1
X | X | O
--+---+--
X | O |
--+---+--
O | |
Player O wins!
Process exited after 69.28 seconds with return value 0
Press any key to continue . . .


C:\Dev-Cpp\tic tac keme.exe
Welcome to Tic Tac Toe!
| |
--+---+--
| |
--+---+--
| |
Player X, enter row (1-3) and column (1-3): 1
1
X | |
--+---+--
| |
--+---+--
| |
Player O, enter row (1-3) and column (1-3): 2
1
X | |
--+---+--
O | |
--+---+--
| |
Player X, enter row (1-3) and column (1-3): 3
1
X | |
--+---+--
O | |
--+---+--
X | |
Player O, enter row (1-3) and column (1-3): 2
2
X | |
--+---+--
O | O |
--+---+--
X | |
Player X, enter row (1-3) and column (1-3): 3
3
X | |
--+---+--
O | O |
--+---+--
X | | X
Player O, enter row (1-3) and column (1-3): 3
2
X | |
--+---+--
O | O |
--+---+--
X | O | X



7. Supplementary Activity

Analysis for 1:

- So this program just makes a multiplication table. It starts in the main function where everything runs. I made a 10 by 10 table to keep all the answers. Then I used two loops, one to go through the rows and another to go through the columns, to fill it up. Each spot in the table is just the row number times the column number, but it starts at 1 not 0. After that, it prints “Multiplication Table” on the screen. Another two loops are used to print all the numbers neatly, row by row. When one row is done, it moves to the next line so it looks like a real table. At the end, it shows the whole multiplication table from 1×1 up to 10×10.

Analysis for 2:

- This code lets us play a simple Tic Tac Toe game. It sets up a 3×3 board using a 2D array where the players’ moves will be stored. The printboard() function shows the board after every turn, while the checkwin() function checks the rows, columns, and diagonals to see if someone has three in a row. In the main() function, the game begins with a welcome message, shows the empty board, and then asks players to type the row and column for their move. The game keeps going until a player wins or all nine turns are used, and if no one wins, it ends in a draw.

8. Conclusion

- In this activity, I learned how 2D arrays can store values in rows and columns, which makes it easier to create things like a multiplication table. The activity also showed me how nested loops work to fill the arrays and print them neatly without typing each value one by one. In the supplementary activity, I learned how arrays and conditions can be used to make a Tic Tac Toe game, where the program checks moves after every turn and decides if someone wins or if it's a draw. About my performance in this activity, I think I did okay because I struggled with some errors that took me a long time to fix, but I was able to solve them in the end even though new ones kept showing up. For me, I need to improve on finding errors faster because it still takes me hours to spot a single mistake.
-