Activity No. 5.1	
Multidimensional Arrays	
Course Code: CPE 007	**Program:** Computer Engineering
Course Title: Programming Logic and Design	**Date Performed: 09/25/2025**
Section: CPE11S1	**Date Submitted: 09/30/2025**
Name(s): Canoy Hail B.	**Instructor: Engr. Jimlord M. Quejado**
6. Output	
7. Supplementary Activity	

1. Write a program that creates a multiplication table using multidimensional array.

Code :

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

    int main (){
        const int size = 10;
        int table[size][size];

        for (int i = 0; i < size; i++){
            for (int j = 0; j < size; j++){
                cout << (i+1) * (j+1) << "\t";

                if (j == size -1){
                    cout << endl;
                }
            }
        }


    return 0;
}
```

Output :

```
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 0.03395 seconds with return value 0
Press any key to continue . . .
```

Explanation :

- So basically this code starts with **const int size = 10;** to represent size which is 10 x 10 then **int table[size][size];** to create a 2d array named table then **for (int i = 0; i < size; i++){** to read 0 to 10 in row then increase increment by 1 **for (int j = 0; j < size; j++){** to read from 0 to 10 in column then increase increment by 1 then**cout << (i+1) * (j+1) << "\t";** to print the i+1 multiplied j+1 basically row multiplied by column then **if (j == size -1){ cout << endl;** for it to print a newline when j is size - 1 which is 9 then **return 0;** to finish the code.

2. Write a program that creates a board with a tic-tac-toe moves.

Code :

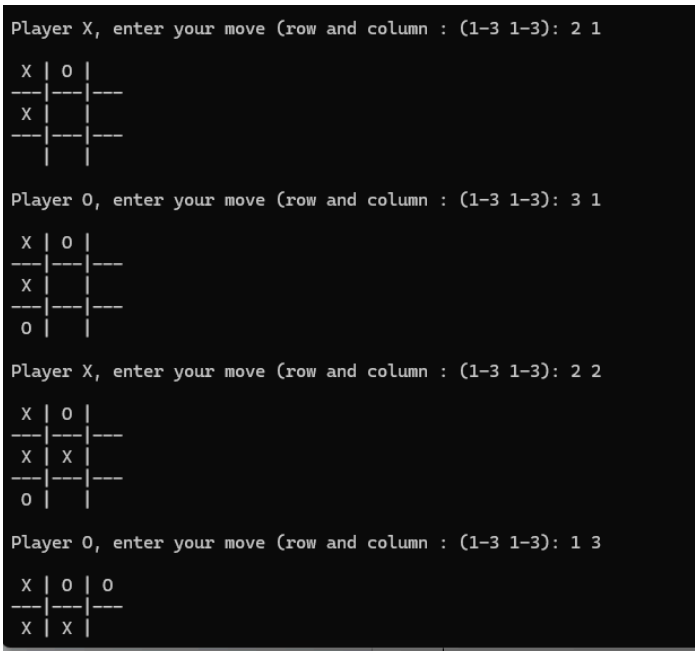
```cpp
#include <iostream>
using namespace std;

char board[3][3];

void initializeBoard() {
    for (int i = 0; i < 3; i++)
        for (int j = 0; j < 3; j++)
            board[i][j] = ' ';
}

void printBoard() {
    cout << "\n";
    for (int i = 0; i < 3; i++) {
        cout << " " << board[i][0] << " | " << board[i][1] << " | " << board[i][2] << "\n";
        if (i < 2) cout << "---|---|---\n";
    }
    cout << "\n";
}

char checkWinner() {
    for (int i = 0; i < 3; i++) {
        if (board[i][0] != ' ' &&
            board[i][0] == board[i][1] &&
            board[i][1] == board[i][2])
            return board[i][0];
        if (board[0][i] != ' ' &&
            board[0][i] == board[1][i] &&
            board[1][i] == board[2][i])
            return board[0][i];
    }

    if (board[0][0] != ' ' &&
        board[0][0] == board[1][1] &&
```

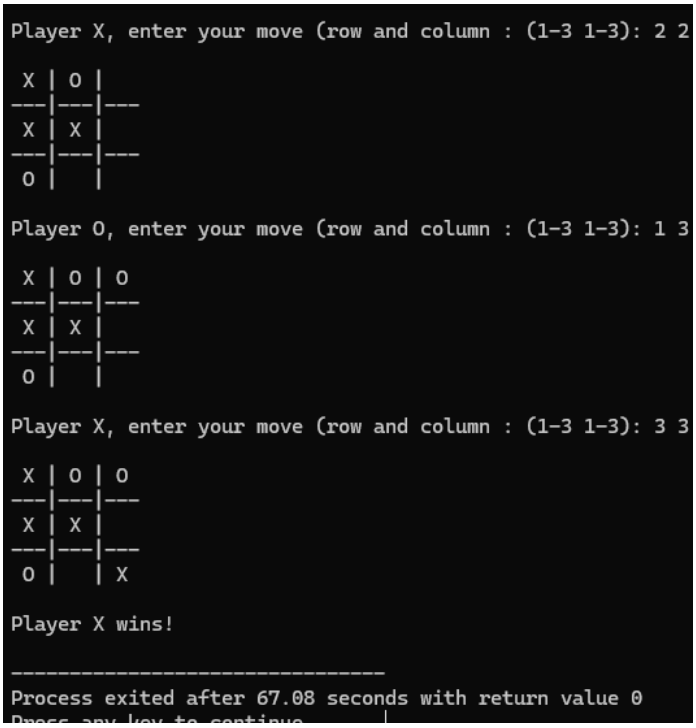
```
        board[1][1] == board[2][2])
        return board[0][0];

    if (board[0][2] != ' ' &&
        board[0][2] == board[1][1] &&
        board[1][1] == board[2][0])
        return board[0][2];

    return ' ';
}

bool isDraw() {
    for (int i = 0; i < 3; i++)
        for (int j = 0; j < 3; j++)
            if (board[i][j] == ' ')
                return false;
    return true;
}

bool makeMove(char player) {
    int row, col;
    cout << "Player " << player << ", enter your move (row and column : (1-3 1-3)): ";
    cin >> row >> col;

    row--;
    col--;

    if (row >= 0 && row < 3 && col >= 0 && col < 3 && board[row][col] == ' ') {
        board[row][col] = player;
        return true;
    } else {
        cout << "Invalid move. Try again.\n";
        return false;
    }
```

```
    if (row >= 0 && row < 3 && col >= 0 && col < 3 && board[row][col] == ' ') {
        board[row][col] = player;
        return true;
    } else {
        cout << "Invalid move. Try again.\n";
        return false;
    }
}

int main() {
    initializeBoard();
    char currentPlayer = 'X';
    char winner = ' ';

    while (true) {
        printBoard();
        if (makeMove(currentPlayer)) {
            winner = checkWinner();
            if (winner != ' ') {
                printBoard();
                cout << "Player " << winner << " wins!\n";
                break;
            } else if (isDraw()) {
                printBoard();
                cout << "It's a draw!\n";
                break;
            }
            currentPlayer = (currentPlayer == 'X') ? 'O' : 'X';
        }
    }

    return 0;
```

Output :

```
   |   |
---|---|---
   |   |
---|---|---
   |   |

Player X, enter your move (row and column : (1-3 1-3): 1 1

 X |   |
---|---|---
   |   |
---|---|---
   |   |

Player O, enter your move (row and column : (1-3 1-3): 1 2

 X | O |
---|---|---
   |   |
---|---|---
   |   |

Player X, enter your move (row and column : (1-3 1-3): 2 1

 X | O |
---|---|---
 X |   |
---|---|---
   |   |
```





Explanation :

char board[3][3]; declared a 2d array of 3x3 then **void initializeBoard()** void is for printing text without result back **{for (int i = 0; i < 3; i++) for (int j = 0; j < 3; j++) board[i][j] = ' ';** for setting up lines **void printBoard() { cout << "\n"; for (int i = 0; i < 3; i++) { cout << " " << board[i][0] << " | " << board[i][1] << " | " << board[i][2] << "\n"; if (i < 2) cout << "---|---|---\n";** is to print the table with lines and spaces **char checkWinner() { for (int i = 0; i < 3; i++) { if (board[i][0] != ' ' && board[i][0] == board[i][1] && board[i][1] == board[i][2]) return board[i][0]; if (board[0][i] != ' ' && board[0][i] == board[1][i] && board[1][i] == board[2][i]) return board[0][i]; if (board[0][0] != ' ' && board[0][0] == board[1][1] && oard[1][1] == board[2][2]) eturn board[0][0]; if (board[0][2] != ' ' && board[0][2] == board[1][1] && board[1][1] == board[2][0]) return board[0][2]; return ' ';** basically checks if x wins or O wins in a row, columns, or even diagonal and tells ‘ ‘ if no winner yet **bool is**

Draw() { for (int i = 0; i < 3; i++) for (int j = 0; j < 3; j++) if (board[i][j] == ' ') return false; return true; if all spaces are filled and there is no 3 matching x or o the program will print draw **bool makeMove(char player) {int row, col; cout << "Player " << player << ", enter your move (row and column : (1-3 1-3): "; cin >> row >> col; row--; col--; if (row >= 0 && row < 3 && col >= 0 && col < 3 && board[row][col] == ' ') {board[row][col] = player; return true;} else { cout << "Invalid move. Try again.\n"; return false;** prints instruction for the player to input if its valid it will print the x or o in the free spaces, if you put an invalid number the program will print invalid and you will try again

int main() { initializeBoard(); char currentPlayer = 'X'; char winner = ' '; while (true) { printBoard(); if (makeMove(currentPlayer)) { winner = checkWinner(); if (winner != ' ') { printBoard(); cout << "Player " << winner << " wins!\n"; break; } else if (isDraw()) { printBoard(); cout << "It's a draw!\n"; break; currentPlayer = (currentPlayer == 'X') ? 'O' : 'X'; this is basically to print the board then lets the current player input a number it will display the move then it will check for a win or draw then it alternates the move or x and o the program will end if the spaces are filled

8. Conclusion

- In this activity I was really stressed, it was really not easy, the first one is kinda easy but is still a little complicated to me. I spent days researching and watching tutorials on youtube . It was hard for me to understand number 2, but with this I learned to organize a very long code, and how to understand and analyze to make a code. This activity made me practice and drained me so I can be a better coder. In number 1 it was easy for me to do, with the help of old codes I referenced it and finished it in 1 day but the second one really took time. I understand the structure of the code thoroughly in number 1 and 2 but the problem is I don't quite get number 2 even though it's already done. It's still so complicated for me. But overall this activity was really helpful. It pressured me how much I still need to learn but like they say pressure makes diamond, I'am still learning and adapting though this and hoping I would be a great coder someday, I want to learn so I can feel more confident about my coding.