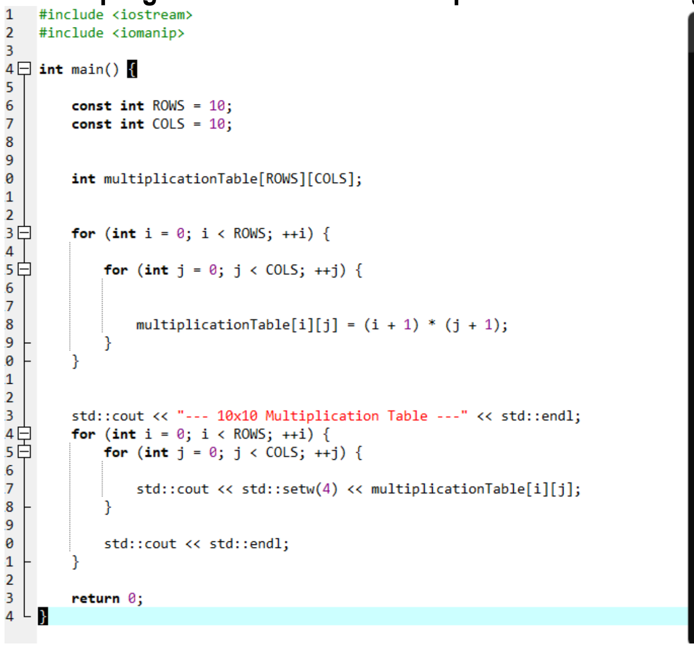
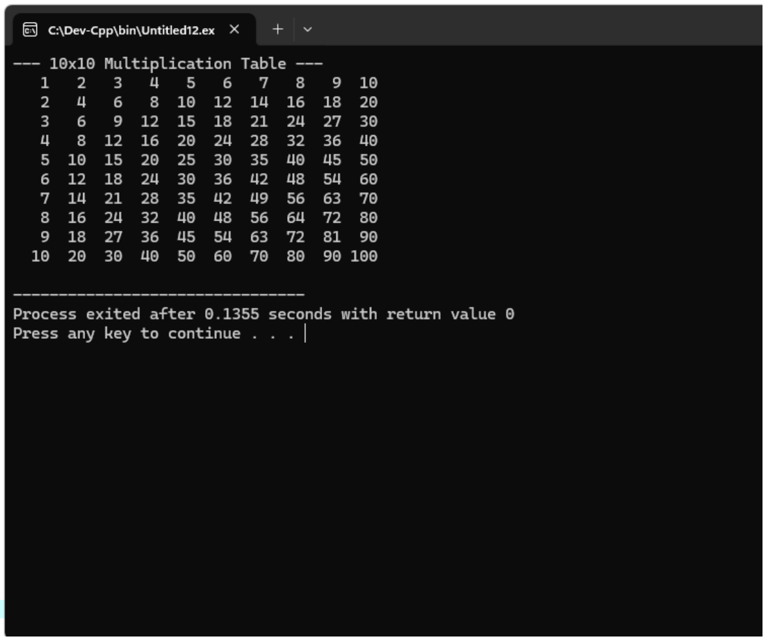
Activity No. <5.1>	
<Multidimensional Arrays >	
Course Code: CPE007	**Program:** Computer Engineering
Course Title: CPE 007-CPE11S1 - Programming Logic and Design	**Date Performed:9/2525**
Section:CPE11S1	**Date Submitted:9/25/25**
Name(s):Will Stuart D. Ponce Jr.	**Instructor: Engr. Jimlord Quejado**

6. Output

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

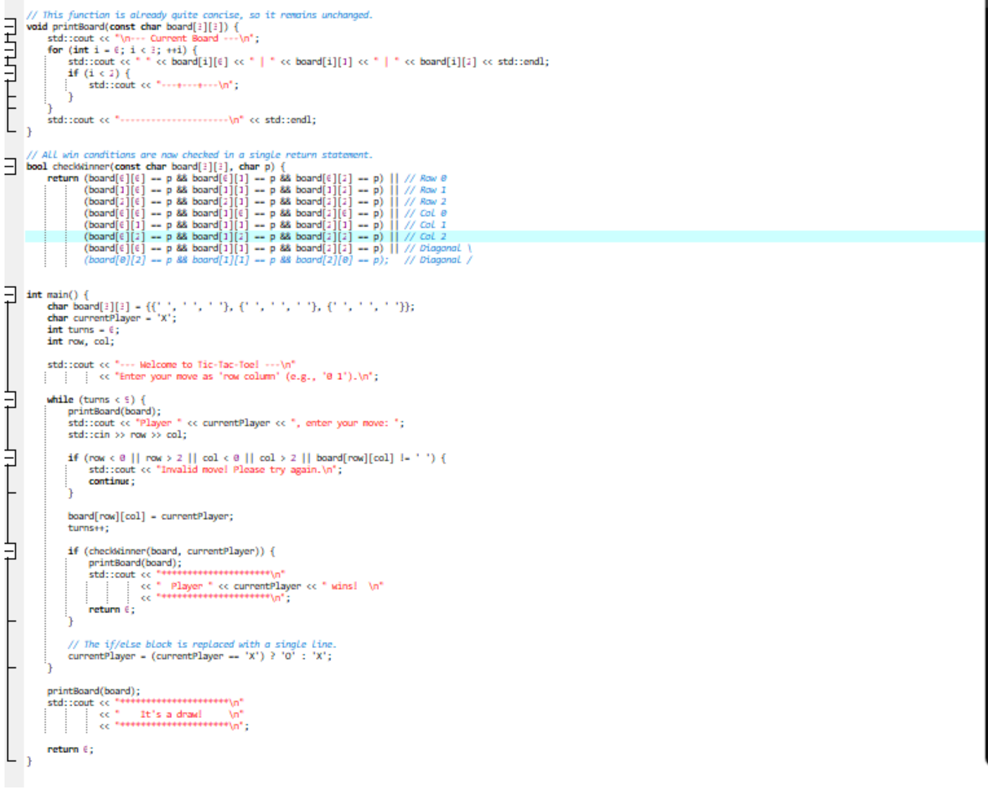


```cpp
#include <iostream>
#include <iomanip>

int main() {

    const int ROWS = 10;
    const int COLS = 10;


    int multiplicationTable[ROWS][COLS];


    for (int i = 0; i < ROWS; ++i) {

        for (int j = 0; j < COLS; ++j) {


            multiplicationTable[i][j] = (i + 1) * (j + 1);
        }
    }


    std::cout << "--- 10x10 Multiplication Table ---" << std::endl;
    for (int i = 0; i < ROWS; ++i) {
        for (int j = 0; j < COLS; ++j) {

            std::cout << std::setw(4) << multiplicationTable[i][j];
        }

        std::cout << std::endl;
    }

    return 0;
}
```

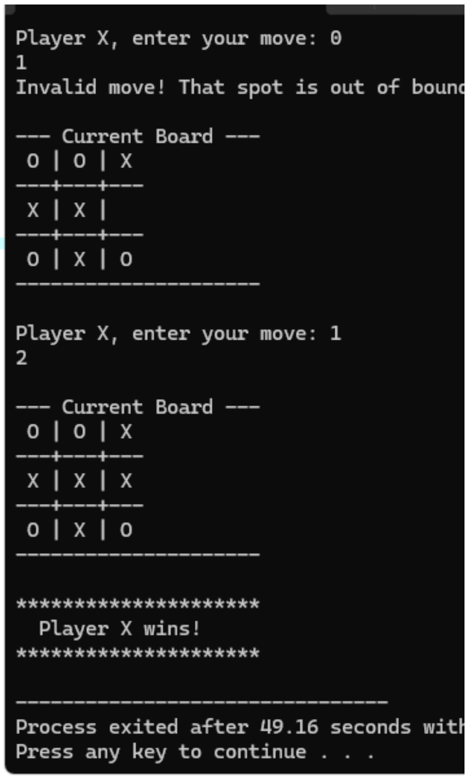


```
C:\Dev-Cpp\bin\Untitled12.ex

--- 10x10 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 0.1355 seconds with return value 0
Press any key to continue . . .
```

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



```cpp
// This function is already quite concise, so it remains unchanged.
void printBoard(const char board[3][3]) {
    std::cout << "\n--- Current Board ---\n";
    for (int i = 0; i < 3; ++i) {
        std::cout << " " << board[i][0] << " | " << board[i][1] << " | " << board[i][2] << std::endl;
        if (i < 2) {
            std::cout << "---+---+---\n";
        }
    }
    std::cout << "---------------------\n" << std::endl;
}

// All win conditions are now checked in a single return statement.
bool checkWinner(const char board[3][3], char p) {
    return (board[0][0] == p && board[0][1] == p && board[0][2] == p) || // Row 0
           (board[1][0] == p && board[1][1] == p && board[1][2] == p) || // Row 1
           (board[2][0] == p && board[2][1] == p && board[2][2] == p) || // Row 2
           (board[0][0] == p && board[1][0] == p && board[2][0] == p) || // Col 0
           (board[0][1] == p && board[1][1] == p && board[2][1] == p) || // Col 1
           (board[0][2] == p && board[1][2] == p && board[2][2] == p) || // Col 2
           (board[0][0] == p && board[1][1] == p && board[2][2] == p) || // Diagonal \
           (board[0][2] == p && board[1][1] == p && board[2][0] == p);   // Diagonal /


int main() {
    char board[3][3] = {{' ', ' ', ' '}, {' ', ' ', ' '}, {' ', ' ', ' '}};
    char currentPlayer = 'X';
    int turns = 0;
    int row, col;

    std::cout << "--- Welcome to Tic-Tac-Toe! ---\n"
              << "Enter your move as 'row column' (e.g., '0 1').\n";

    while (turns < 9) {
        printBoard(board);
        std::cout << "Player " << currentPlayer << ", enter your move: ";
        std::cin >> row >> col;

        if (row < 0 || row > 2 || col < 0 || col > 2 || board[row][col] != ' ') {
            std::cout << "Invalid move! Please try again.\n";
            continue;
        }

        board[row][col] = currentPlayer;
        turns++;

        if (checkWinner(board, currentPlayer)) {
            printBoard(board);
            std::cout << "*********************\n"
                      << "  Player " << currentPlayer << " wins!  \n"
                      << "*********************\n";
            return 0;
        }

        // The if/else block is replaced with a single line.
        currentPlayer = (currentPlayer == 'X') ? 'O' : 'X';
    }

    printBoard(board);
    std::cout << "*********************\n"
              << "    It's a draw!     \n"
              << "*********************\n";

    return 0;
}
```



```
Player X, enter your move: 0
1
Invalid move! That spot is out of bound

--- Current Board ---
 O | O | X
---+---+---
 X | X |
---+---+---
 O | X | O
---------------------


Player X, enter your move: 1
2

--- Current Board ---
 O | O | X
---+---+---
 X | X | X
---+---+---
 O | X | O
---------------------

*********************
  Player X wins!
*********************


--------------------------------
Process exited after 49.16 seconds with
Press any key to continue . . .
```

7. Supplementary Activity
Analysis
With the use of a two-dimensional integer array named multiplication Table, this C++ program is able to create and show a 10x10 multiplication table, in which all the multiplied values are stored in a neatly organized grid structure At the very beginning, the program uses two nested for loops to fill in the array and the main calculation is (i + 1) * (j + 1) for each cell to insert the correct product.

This C++ code is a full, command-line Tic-Tac-Toe game, emphasizing the conciseness of the code. A 3x3 character array is used to efficiently manage the game's state, along with a turn counter and a current player variable to track progress. The check Winner function is very brief, consisting of a single return statement with logical OR (||) operators, which, in one step, check all eight winning situations at the same time. The user input is wrapped in a while loop that also checks the moves for validity so that no square can be overwritten and no coordinates out of range can be used.

8.Conclusion
Three snippets in C++ you provided are great examples of fundamental programming concepts, each with a different emphasis. The first Tic-Tac-Toe program exhibited a straightforward, readable, and well-structured approach to creating a basic game, isolating the printing the board and checking the winner functionalities as separate functions.The multiplication table program was a showcase for 2-dimensional arrays as a practical solution for storing tabular data, and it also demonstrated the significance of output formatting by <iomanip> for neat, aligned displays.At last, the shortened version of the Tic-Tac-Toe code gave insight into the same logic that can be refactored for brevity and efficiency by merging conditional checks and utilizing contemporary C++ operators such as the ternary operator.