Activity No. 4.4	
Hands-on Activity 4.4: Characters and Strings	
Course Code: CPE007	**Program:** Computer Engineering
Course Title: Programming Logic and Design	**Date Performed: 9 – 30 – 25**
Section: CPE11S1	**Date Submitted: 9 – 30 – 25**
Name(s): Paula Esguerra	**Instructor: Engr. Jimlord Quejado**
6. Output	

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

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

int main (){
    const int SIZE = 10;
    int multiplicationTable[SIZE][SIZE];

    for (int i = 0; i < SIZE; ++i) {
        for (int j = 0; j < SIZE; ++j) {
            multiplicationTable[i][j] = (i +1 ) * (j + 1);
        }
    }

    std::cout << "Multiplication Table (1x1 to  " << SIZE << "x" << SIZE << "):\n";

    for (int j = 0; j < SIZE; ++j) {
        std::cout << "----";
    }


    std::cout << std::endl;

    for (int i = 0; i < SIZE; ++i) {
        std::cout << std::setw(4) << (i + 1);
    }
    std::cout << std::endl;

    std::cout << "----";
    for (int j = 0; j < SIZE; ++j) {
        std::cout << "----";
    }
    std::cout << std::endl;

    for (int i = 0; i < SIZE; ++i){
        std::cout << std::setw(3) << (i + 1) << "|";
        for (int j = 0; j < SIZE; ++j) {
            std::cout << std::setw(4) << multiplicationTable[i][j];
        }
        std::cout << std::endl;

    }

    return 0;



}
```

```
Multiplication Table (1x1 to  10x10:
----------------------------------------
   1   2   3   4   5   6   7   8   9  10
----------------------------------------
  1|   1   2   3   4   5   6   7   8   9  10
  2|   2   4   6   8  10  12  14  16  18  20
  3|   3   6   9  12  15  18  21  24  27  30
  4|   4   8  12  16  20  24  28  32  36  40
  5|   5  10  15  20  25  30  35  40  45  50
  6|   6  12  18  24  30  36  42  48  54  60
  7|   7  14  21  28  35  42  49  56  63  70
  8|   8  16  24  32  40  48  56  64  72  80
  9|   9  18  27  36  45  54  63  72  81  90
 10|  10  20  30  40  50  60  70  80  90 100

--------------------------------
Process exited after 0.1752 seconds with return value 0
Press any key to continue . . .
```

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

```cpp
#include <iostream>

class Player {
private:
    char symbol;
    std::string name;

public:
    Player(char sym = 'X', std::string n = "Player X") : symbol(sym), name(n) {}

    char getSymbol() const { return symbol; }
    std::string getName() const { return name; }
};


class Board {
private:
    char grid[3][3];
    int filledCells;

public:

    Board() : filledCells(0) {
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++) {
                grid[i][j] = ' ';
            }
        }
    }

```

```
31
32      void drawBoard() const {
33          std::cout << "-------------" << std::endl;
34          for (int i = 0; i < 3; i++) {
35              std::cout << "| ";
36              for (int j = 0; j < 3; j++) {
37                  std::cout << grid[i][j] << " | ";
38              }
39              std::cout << std::endl << "-------------" << std::endl;
40          }
41      }
42
43
44      bool isValidMove(int row, int col) const {
45          return (row >= 0 && row < 3 && col >= 0 && col < 3 && grid[row][col] == ' ');
46      }
47
48
49      void makeMove(int row, int col, char symbol) {
50          if (isValidMove(row, col)) {
51              grid[row][col] = symbol;
52              filledCells++; // Increment counter when a cell is filled
53          }
54      }
55
56      bool checkWin(char symbol) const {
57
58          for (int i = 0; i < 3; i++) {
59              if (grid[i][0] == symbol && grid[i][1] == symbol && grid[i][2] == symbol) {
60                  return true;
61              }
62          }
63
64          for (int i = 0; i < 3; i++) {
65              if (grid[0][i] == symbol && grid[1][i] == symbol && grid[2][i] == symbol) {
66                  return true;
67              }
68          }
69
70          if (grid[0][0] == symbol && grid[1][1] == symbol && grid[2][2] == symbol) {
71              return true;
72          }
73          if (grid[0][2] == symbol && grid[1][1] == symbol && grid[2][0] == symbol) {
74              return true;
75          }
76
77          return false;
78      }
79
80
81      bool isFull() const {
82          return filledCells == 9;
83      }
84
85
86      int getFilledCellsCount() const {
87          return filledCells;
88      }
89  };
90
```

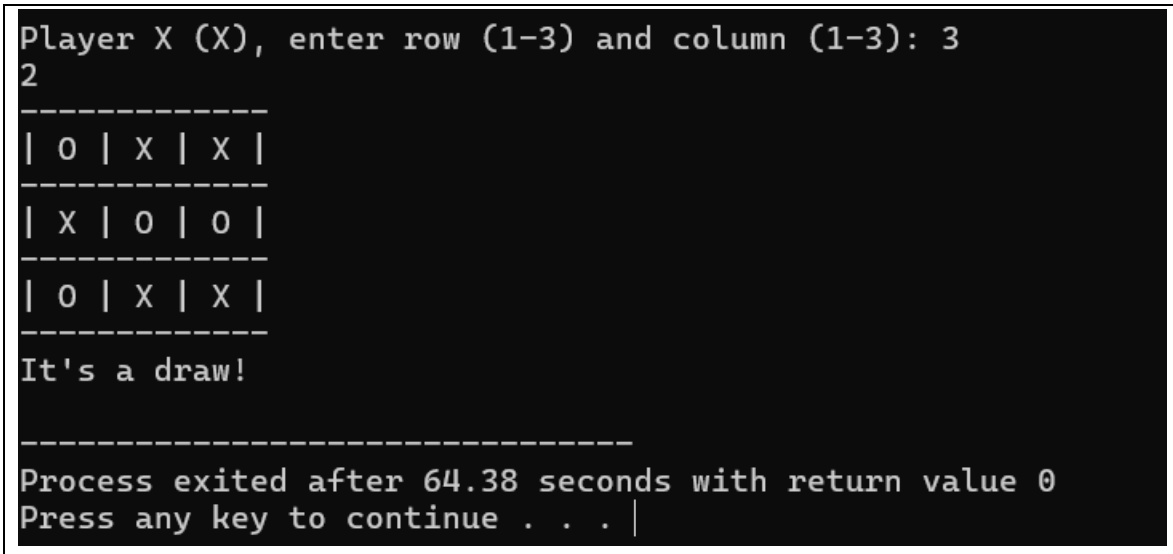
```cpp
class TicTacToe {
private:
    Board board;
    Player players[2];
    int currentPlayerIndex;

public:
    TicTacToe() : currentPlayerIndex(0) {
        players[0] = Player('X', "Player X");
        players[1] = Player('O', "Player O");
    }

    Player& getCurrentPlayer() {
        return players[currentPlayerIndex];
    }

    void switchTurn() {
        currentPlayerIndex = (currentPlayerIndex + 1) % 2;
    }

    void play() {
        int row, col;
        std::cout << "Welcome to Tic-Tac-Toe!" << std::endl;

        while (!board.isFull()) {

            board.drawBoard();

            Player& currentPlayer = getCurrentPlayer();

                while (true) {
                std::cout << currentPlayer.getName() << " (" << currentPlayer.getSymbol()
                    << "), enter row (1-3) and column (1-3): ";
                std::cin >> row >> col;
                row--; col--;

                if (board.isValidMove(row, col)) {
                    break;
                } else {
                    std::cout << "Invalid move. Try again." << std::endl;
                }
            }


            board.makeMove(row, col, currentPlayer.getSymbol());


            if (board.checkWin(currentPlayer.getSymbol())) {
                board.drawBoard();
                std::cout << currentPlayer.getName() << " wins!" << std::endl;
                return;
            }


            switchTurn();
        }


        board.drawBoard();
        std::cout << "It's a draw!" << std::endl;
```

```
    }
};

int main() {

    TicTacToe game;

    game.play();
    return 0;
}

```

```
Welcome to Tic-Tac-Toe!
-------------
|   |   |   |
-------------
|   |   |   |
-------------
|   |   |   |
-------------
Player X (X), enter row (1-3) and column (1-3): 1
2
-------------
|   | X |   |
-------------
|   |   |   |
-------------
|   |   |   |
-------------
Player O (O), enter row (1-3) and column (1-3): 2
2
-------------
|   | X |   |
-------------
|   | O |   |
-------------
|   |   |   |
-------------
Player X (X), enter row (1-3) and column (1-3): 1
3
-------------
|   | X | X |
```

```
-------------
|   | O |   |
-------------
|   |   |   |
-------------
Player O (O), enter row (1-3) and column (1-3): 1
1
-------------
| O | X | X |
-------------
|   | O |   |
-------------
|   |   |   |
-------------
Player X (X), enter row (1-3) and column (1-3): 3
3
-------------
| O | X | X |
-------------
|   | O |   |
-------------
```

```
Player O (O), enter row (1-3) and column (1-3): 2
3
-------------
| O | X | X |
-------------
|   | O | O |
-------------
|   |   | X |
-------------
Player X (X), enter row (1-3) and column (1-3): 2
1
-------------
| O | X | X |
-------------
| X | O | O |
-------------
|   |   | X |
-------------
Player O (O), enter row (1-3) and column (1-3): 3
1
-------------
| O | X | X |
-------------
| X | O | O |
-------------
| O |   | X |
```



7. Supplementary Activity

1. Analysis

First I include iostream and iomanip. Iomanip is to manipulate input or outputs, to gain access to the format manipulators. Then used for loop to display all of the numbers between 0 to 9. So the body of the loop is executed and update expression is evaluated. The loop variable "i" and "j" goes from 0-9 for a total of 10 columns, 0 is part of the count. Then cout to print.

2. Analysis

Bool is a function to check if a player's intended move is valid, if it is true then the specific cell is represented by the character made by the player. If it is false then it will show the intended print for it. It would made to be invalid. Char stores the character representing the player of the board. String stores the name of the player. While loop is to prompts the currentplayer to enter a row and column for their move.

8. Conclusion

In this code, I have practice how to use the columns and rows in the code. Though I still have to practice combining for loop, while loop, bool and void, I still have to get the basic out of it, even though I understood the code, I still have trouble using them. I learned that you could use a code to shortcut the method, but some of the doesn't work on devc+ and I still have to figure out the error so I stick to what I have and learned.

9. Assessment Rubric