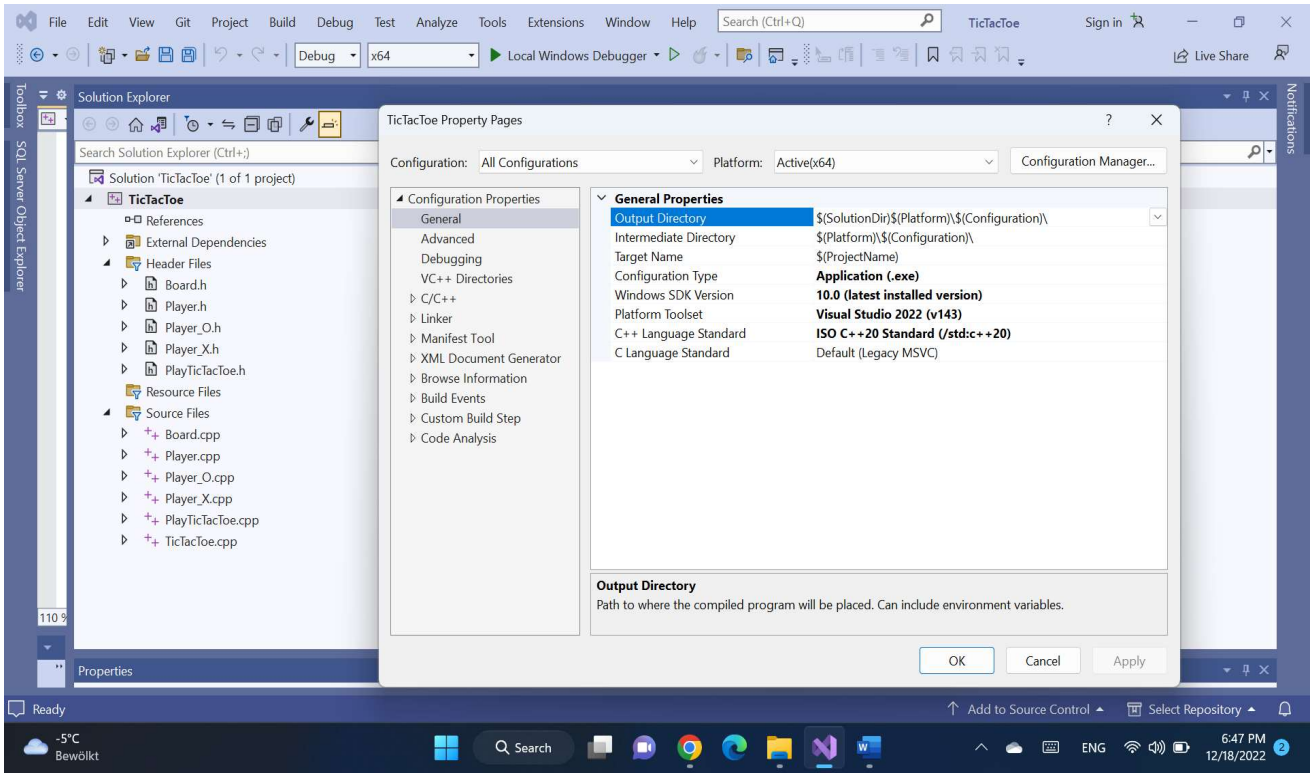
Overview Project TicTacToe Game

Programming Language: C++20
IDE: Microsoft Visual Studio 2022, Community Edition

Major Folders:

1. **TicTacToe. Contains the files that run the program.**
2. **TestTicTacToe. Contains Google tests.**

1. TicTacToe Folder.



class Board: Board.h / Board.cpp
algorithms about the board.

class Player: Player.h / Player.cpp, a base class for Player_X and Player_0
class Player_X: Player_X.h / Player_X.cpp, a derived class from Player
class Player_O: Player_O.h / Player_O.cpp, a derived class from Player
algorithms about the players.

class PlayTicTacToe: PlayTicTacToe.h / PlayTicTacToe.cpp
combines all pieces together and contains the method that starts the program:
void start();

File TicTacToe.cpp: contains method 'int main()' with initialized class PlayTicTacToe:

```
int main ()
{
    PlayTicTacToe play;
    play.start();
}
```

Short instructions accompany the players at every step: all that is needed is to start the game and follow instructions. The console is cleared after each move, thus the displayed board is the most recent one, after the latest move of one of the players.

By default, the board is initialized with a square of 3 rows and 3 columns. But it is easily extendable for a different size of square, either for a new game or for each separate round within a game. Input from players is in the form “digit:digit”, which stands for “row:column”. Valid input range for row/column for a player is from 1 to 3 for the default size of the board. If the board is initialized with different square size, then the valid input range for row/column for a player is from 1 to the side of square. Maximum size of the board is 9 rows and 9 columns.

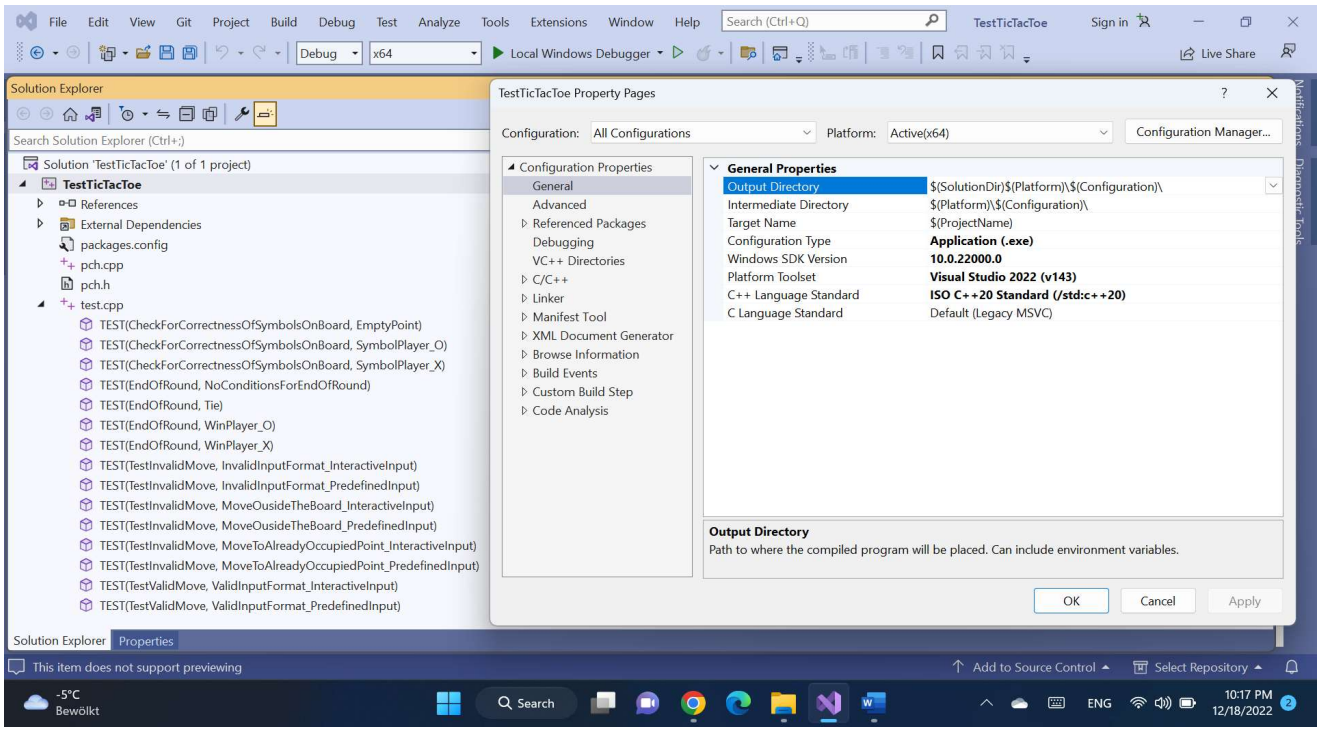
Each game always starts the first round with player X making the first move. After that, the player making the last move that sets end of round conditions – i.e., making the winning move, or a move that results in a draw – is the player that starts second in the following round. In other words, if there is a losing player, then this player always makes the first move in the following round.

After each round, a summary of statistics about the previous rounds is displayed: wins per player, ties, total rounds played. Next round can be started by pressing ‘enter’ and the game can be exited by pressing any other key.

Winning combination is not calculated by directly comparing symbols of rows/columns/diagonals. Rather, after each move, there is an update for the ASCII sum of the symbols of the row/column/diagonal (if it is on such a point) of this move.

Then, if this sum is equal to totalRows*static_cast<int>(‘X’) or totalRows*static_cast<int>(‘O’) or totalColumns*static_cast<int>(‘X’) or totalColumns*static_cast<int>(‘O’), there is a winner. Actually, in this context, the characters will automatically convert to their corresponding ASCII integers. However, I consider it a good practice to write explicit code and leave no room for vagueness.

2. TestTicTacToe Folder.



In order to facilitate testing, some of the private fields in certain classes are made public. In such cases, these are appropriately marked with comments.

Since this is an interactive application, constantly taking an input from the players, some of the methods are overloaded to take a predefined input during the tests. However, for the sake of demonstration, there are tests that check both the original method that requires an input during the test and the specifically overloaded method that takes a predefined input. Again, these are appropriately marked with comments.

File test.cpp starts all the tests.