Cairo University

Faculty of Computers and Artificial Intelligence

CS213: Object Oriented Programming

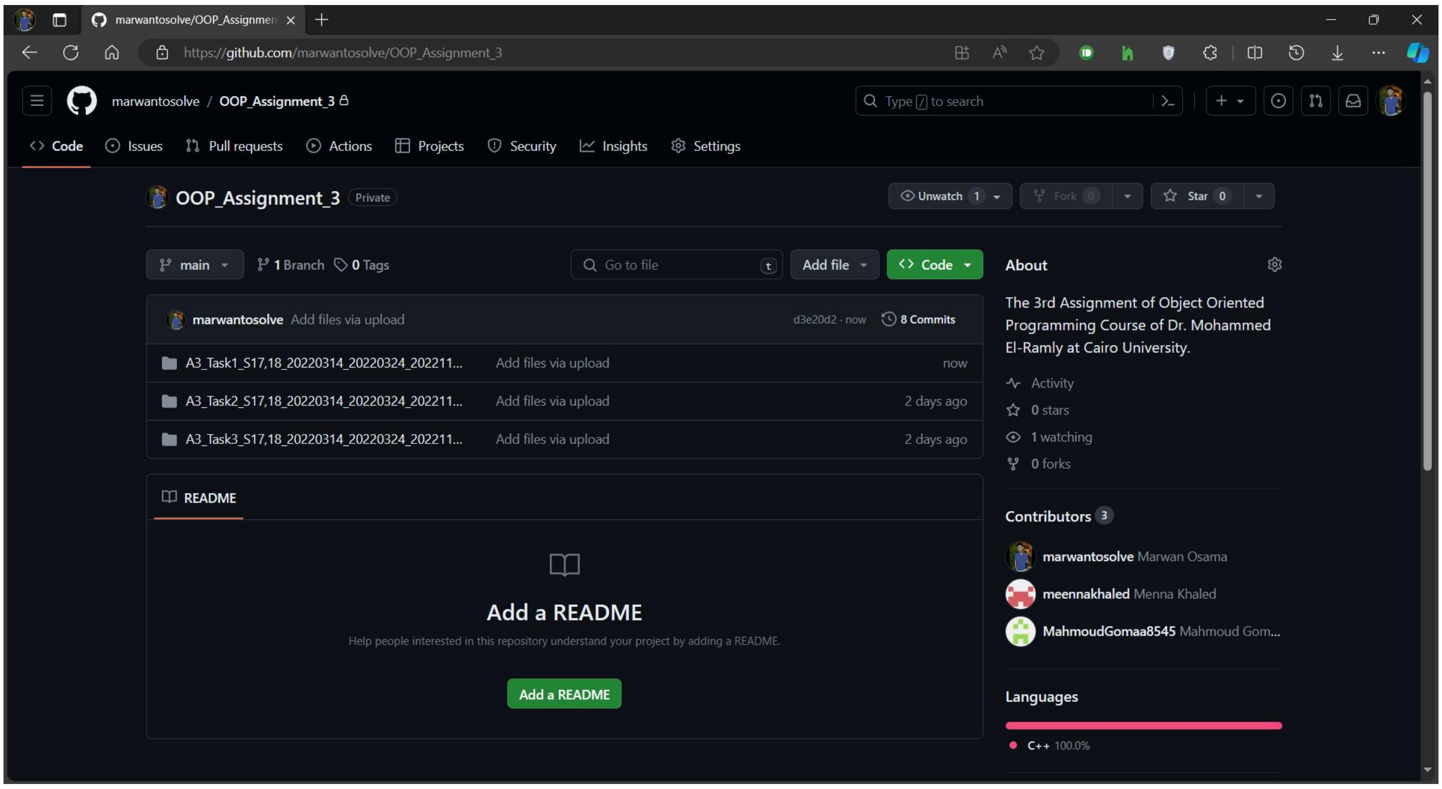
Assignment 3

PDF Report of Task 2, 3 and 4

Name	**ID**
Marwan Osama Abd El-Azem	20220324
Menna Khaled Gamal	20221166
Mahmoud Gomaa Rabee	20220314

• First: Proofs of using GitHub

There’s a screenshot showing our contributions and commits on the Repository.



marwantosolve/OOP_Assignment_3

https://github.com/marwantosolve/OOP_Assignment_3

marwantosolve / OOP_Assignment_3

Code | Issues | Pull requests | Actions | Projects | Security | Insights | Settings

OOP_Assignment_3 Private

Unwatch 1 | Fork 0 | Star 0

main | 1 Branch | 0 Tags | Go to file | Add file | Code

marwantosolve Add files via upload d3e20d2 · now 8 Commits

A3_Task1_S17,18_20220314_20220324_202211...	Add files via upload	now
A3_Task2_S17,18_20220314_20220324_202211...	Add files via upload	2 days ago
A3_Task3_S17,18_20220314_20220324_202211...	Add files via upload	2 days ago

README

Add a README

Help people interested in this repository understand your project by adding a README.

Add a README

About

The 3rd Assignment of Object Oriented Programming Course of Dr. Mohammed El-Ramly at Cairo University.

Activity
0 stars
1 watching
0 forks

Contributors 3

marwantosolve Marwan Osama
meennakhaled Menna Khaled
MahmoudGomaa8545 Mahmoud Gom...

Languages

C++ 100.0%

Link: https://github.com/marwantosolve/OOP_Assignment_3

• Second: A detailed program description and the idea behind each function in the inherited Board class we wrote for our games

• “fourInRow” class for the board of four in a row game:

Constructor fourInRow::fourInRow()

This constructor initializes the game board with empty spaces. It sets the number of rows (n_rows) to 6 and the number of columns (n_cols) to 7. The game board is a 2D array (board) of characters, where each element is initially set to '.'.

Method fourInRow::update_board(int x, int y, char mark)

This method is used to update the game board with a player's move. It takes three parameters: x (which is not used), y (the column where the player wants to make a move), and mark (the player's marker, either 'X' or 'O').

It checks if the selected column is valid (between 1 and 7). If not, it outputs an error message and returns false.

It then iterates through the selected column to find the first available row (starting from the bottom) and updates the board with the player's marker.

It increments the n_moves counter.

If the move is successful, it returns true. Otherwise, if the column is full, it outputs an error message and returns false.

Method fourInRow::is_winner()

This method checks if there is a winning combination on the game board. It iterates through each cell on the board and checks for horizontal, vertical, and diagonal sequences of four identical markers.

Method fourInRow::is_draw()

This method checks if the game is a draw. It returns true if the number of moves (n_moves) is equal to 42 (indicating that the board is full) and there is no winner.

Method fourInRow::game_is_over()

This method returns true if the number of moves (n_moves) is greater than or equal to 42, indicating that the game is over.

Method fourInRow::display_board()

This method displays the current state of the game board in a readable format. It prints the column numbers at the top and then prints each row, including the player markers or empty spaces.

• “Pyramic_X_O” class for the board of four in a row game:

Constructor (Pyramic_X_O_Board::Pyramic_X_O_Board()):

Initializes the board with a size of 5x5 (n_rows = n_cols = 5).

Allocates memory for a 2D array (board) to represent the game board.

Initializes each element of the board to 0.

bool Pyramic_X_O_Board::update_board(int x, int y, char mark):

Takes three parameters: x and y are the coordinates of the move, and mark is the character ('X' or 'O') to be placed on the board.

Checks if the specified move is valid based on certain conditions (specific coordinates where moves are allowed), and if the cell is not already occupied.

If the move is valid, updates the board with the specified mark, increments the move count (n_moves), and returns true. Otherwise, returns false.

bool Pyramic_X_O_Board::is_winner():

Checks for winning conditions in the game.

Examines various combinations of cells on the board to determine if there is a winner.

Returns true if any of the winning conditions are met, indicating that a player has won. Otherwise, returns false.

bool Pyramic_X_O_Board::is_draw():

Checks if the game is a draw (a tie).

Returns true if the number of moves (n_moves) is 9 (indicating a fully filled board) and no player has won (using is_winner()). Otherwise, returns false.

bool Pyramic_X_O_Board::game_is_over():

Checks if the game is over.

Returns true if the number of moves is greater than or equal to 9 (indicating that the board is fully filled). Otherwise, returns false.

void Pyramic_X_O_Board::display_board():

Presumably, this function would contain code to display the current state of the game board. However, the implementation is not provided in the code snippet.

• "5x5_X_O_board" class for the board of four in a row game:

Constructor (X_O_5x5_board::X_O_5x5_board()):

Initializes the board with a size of 5x5 (n_rows = n_cols = 5).

Allocates memory for a 2D array (board) to represent the game board.

Initializes each element of the board to 0.

bool X_O_5x5_board::update_board(int x, int y, char mark):

Takes three parameters: x and y are the coordinates of the move, and mark is the character ('X' or 'O') to be placed on the board.

Checks if the specified move is within the valid board range and if the cell is not already occupied.

If the move is valid, updates the board with the specified mark, increments the move count (n_moves), and returns true. Otherwise, returns false.

bool X_O_5x5_board::is_winner():

Checks for winning conditions in the game.

Utilizes helper functions (check_row, check_column, check_diagonal_1, check_diagonal_2) to check for matches in rows, columns, and diagonals.

Keeps track of the number of winning positions for each player (p1 for 'X' and p2 for 'O').

Returns true if any player has more winning positions than the other, indicating a winner. Otherwise, returns false.

bool X_O_5x5_board::is_draw():

Checks if the game is a draw (a tie).

Returns true if the number of moves (n_moves) is 24 (indicating a fully filled board) and no player has won (using is_winner()). Otherwise, returns false.

bool X_O_5x5_board::game_is_over():

Checks if the game is over.

Returns true if the number of moves is greater than or equal to 24 (indicating that the board is fully filled). Otherwise, returns false.

bool X_O_5x5_board::is_valid(int x, int y):

Checks if the given coordinates (x, y) are within the valid board range.

void X_O_5x5_board::display_board():

Displays the current state of the game board, including cell coordinates and marks ('X', 'O').

bool X_O_5x5_board::check_row(int x, int y)

bool X_O_5x5_board::check_column(int x, int y)

bool X_O_5x5_board::check_diagonal_1(int x, int y)

bool X_O_5x5_board::check_diagonal_2(int x, int y):

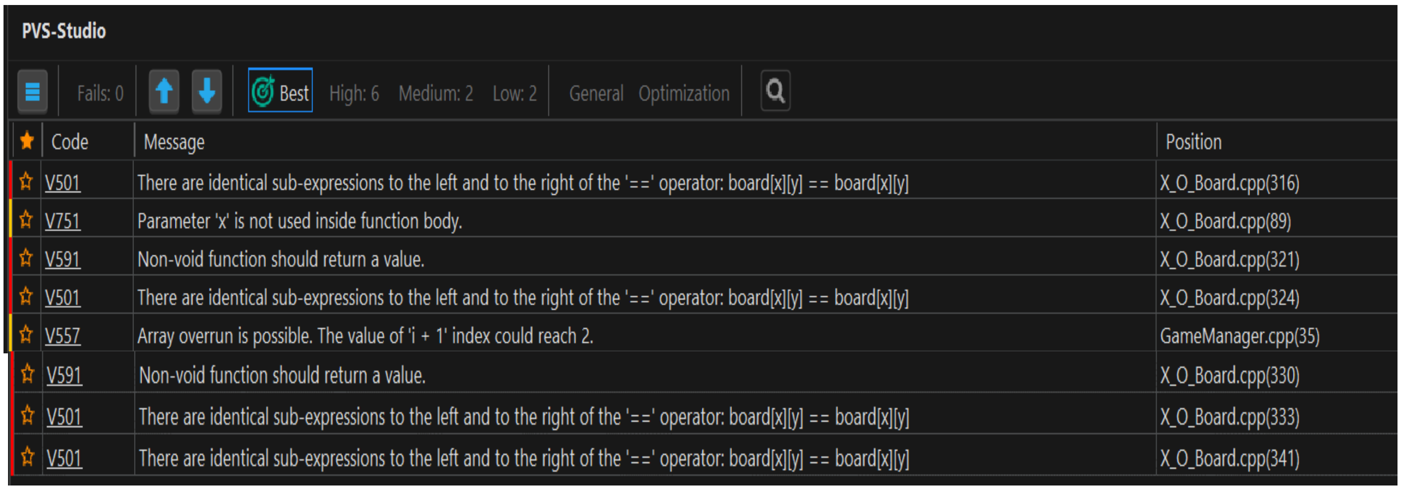
All of them are helper functions to check for winning conditions in rows, columns, and diagonals starting from the specified coordinates.

• Third: Who did what table

Name	ID	Part
Marwan Osama	20220324	- Four In A Row Game - Integrated Games App - GitHub Repository - The pdf Report - Code Quality Report - **AI** for Four In A Row Game - **GUI** for Tic-Tac-Toe Games (connect 4 only)
Mahmoud Gomaa	20220314	- Pyramic X,O Game
Menna Khaled	20221166	- 5x5 X,O Game

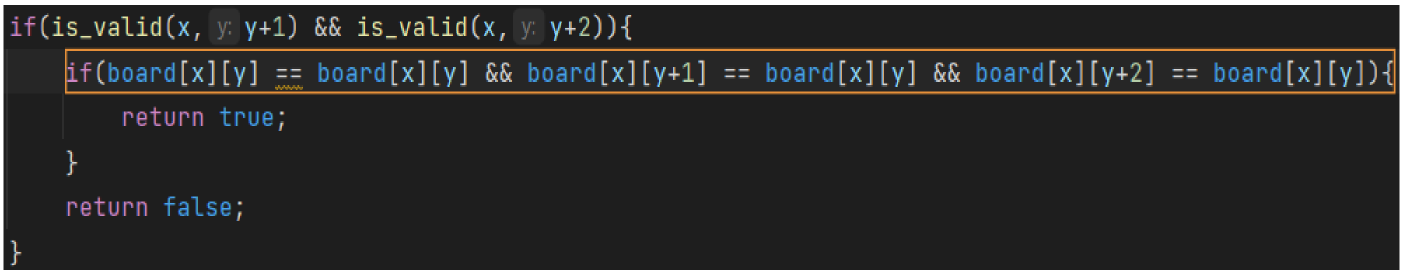
• Fourth: Code Quality Report

After we checked our codes and found no mistakes prevents the program from running perfectly, So we used (PVS Studio Code Analyser) to review our codes and it found few problems we will show the screenshots and explain how we solved it.



Code	Message	Position
V501	There are identical sub-expressions to the left and to the right of the '==' operator: board[x][y] == board[x][y]	X_O_Board.cpp(316)
V751	Parameter 'x' is not used inside function body.	X_O_Board.cpp(89)
V591	Non-void function should return a value.	X_O_Board.cpp(321)
V501	There are identical sub-expressions to the left and to the right of the '==' operator: board[x][y] == board[x][y]	X_O_Board.cpp(324)
V557	Array overrun is possible. The value of 'i + 1' index could reach 2.	GameManager.cpp(35)
V591	Non-void function should return a value.	X_O_Board.cpp(330)
V501	There are identical sub-expressions to the left and to the right of the '==' operator: board[x][y] == board[x][y]	X_O_Board.cpp(333)
V501	There are identical sub-expressions to the left and to the right of the '==' operator: board[x][y] == board[x][y]	X_O_Board.cpp(341)

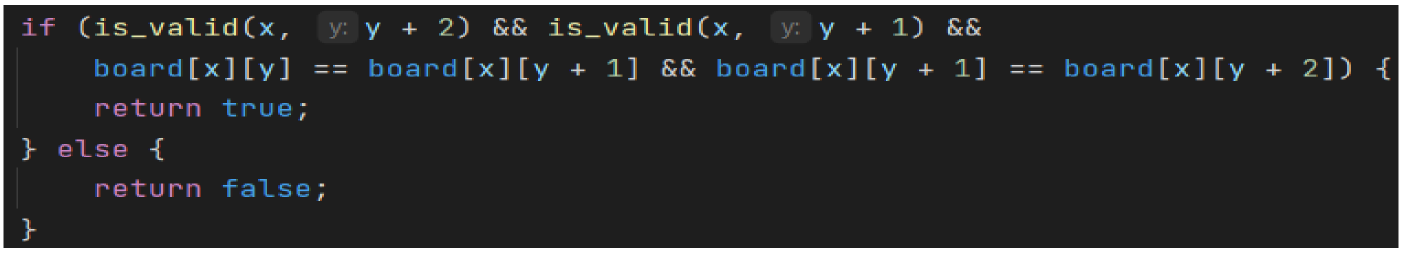
- For the problem **V501:**



```
if(is_valid(x, y: y+1) && is_valid(x, y: y+2)){
    if(board[x][y] == board[x][y] && board[x][y+1] == board[x][y] && board[x][y+2] == board[x][y]){
        return true;
    }
    return false;
}
```

It repeated many times and that’s because there was an unnecessary condition.

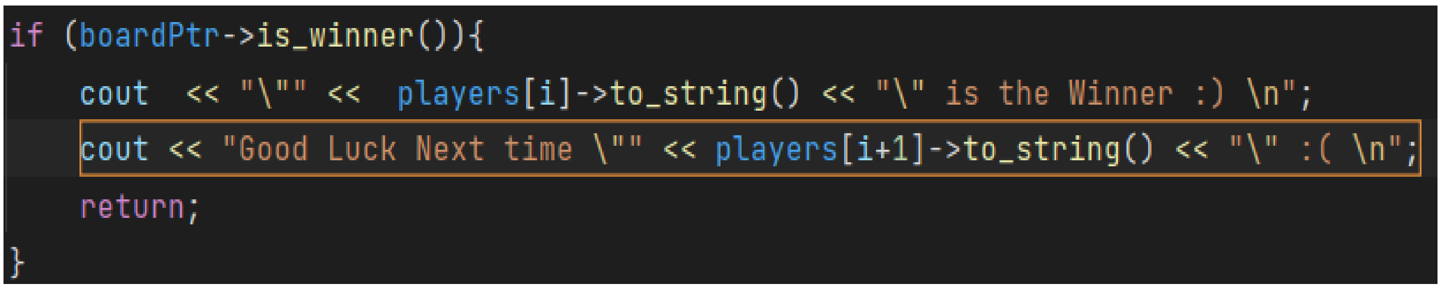
So, we solved the problem by replacing it with:



```
if (is_valid(x,  y: y + 2) && is_valid(x,  y: y + 1) &&
    board[x][y] == board[x][y + 1] && board[x][y + 1] == board[x][y + 2]) {
    return true;
} else {
    return false;
}
```

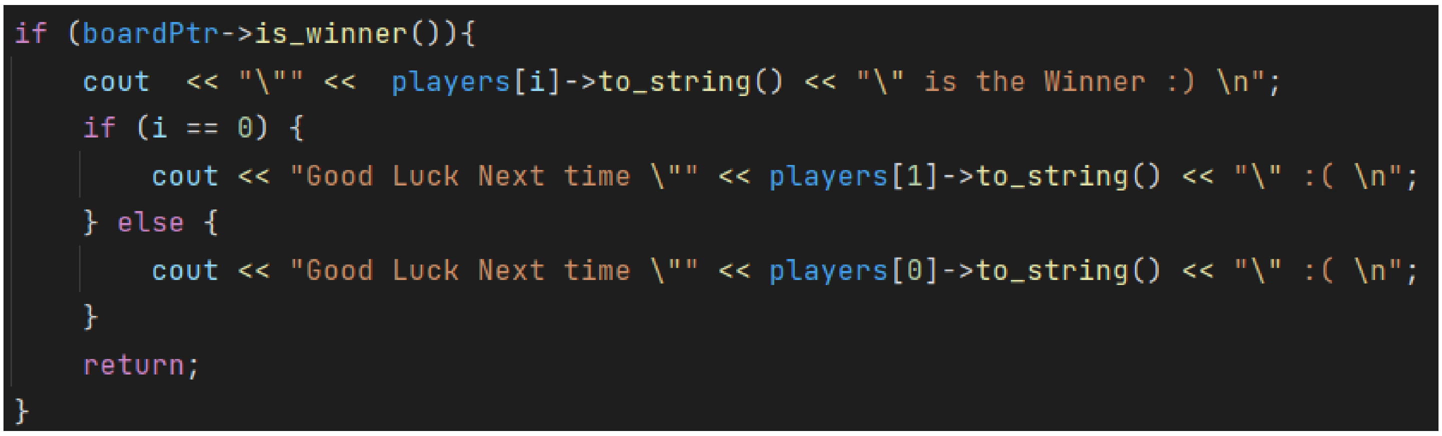
And we solved all of them with the same technique.

- For the Problem **V557:**



```
if (boardPtr->is_winner()){
    cout  << "\"" <<  players[i]->to_string() << "\" is the Winner :) \n";
    cout << "Good Luck Next time \"" << players[i+1]->to_string() << "\" :( \n";
    return;
}
```

In fact, it was repeated in the game manager of every game, so we replaced it with:



```
if (boardPtr->is_winner()){
    cout  << "\"" <<  players[i]->to_string() << "\" is the Winner :) \n";
    if (i == 0) {
        cout << "Good Luck Next time \"" << players[1]->to_string() << "\" :( \n";
    } else {
        cout << "Good Luck Next time \"" << players[0]->to_string() << "\" :( \n";
    }
    return;
}
```

And this solved the problem also.

- For the problems **V591** and **V751:**

They’re just false alarm and doesn’t affect on the code job.