**BÁO CÁO ĐỀ TÀI OOP**

**Nhóm 123**

**Chủ đề: THIẾT KẾ TIC-TẮC-TOE GAME CONSOLE THEO MÔ HÌNH ĐỐI TƯỢNG.**

Danh sách thành viên

1. Nguyễn Ngô Kim Loan - 44.01.104.120
2. Lê Đức Toàn - 46.01.104.184
3. Nguyễn Văn Sỹ - 46.01.103.063
4. Trương Vũ - 47.01.103.110

# MỤC LỤC

[MỤC LỤC 2](#_Toc174249808)

[Mô tả đề 4](#_Toc1905658945)

[Sơ đồ lớp minh họa 5](#_Toc840797197)

[Dữ liệu kiểm thử 5](#_Toc268450925)

[Link Github và Code: 8](#_Toc1036027163)

**GIỚI THIỆU ĐỀ TÀI**

Tic Tac toe là một trong những game đối kháng đơn giản nhất. Không rõ xuất xứ cũngnhư thời gian ra đời, tuy nhiên nó được coi là cổ xưa hơn cờ vua và cờ tướng. Bàn cờ gốc củaTic – tac – toe là bàn cờ 3x3, 2 bên X và O bên nào tạo được một hàng ngang, cột dọc hayđường chéo thì thắng cuộc.

Carô là một biến thể của trò chơi Tic – tac –toe, được phát triển dựa trên cùng ý tưởngtrò chơi, nhưng mở rộng bàn cờ ra khỏi mức 3x3. Cờ caro chính là môn cờ logic lâu đời và cổ xưa nhất trên Trái Đất. Cờ caro đã đượcsáng tạo từ nhiều nền văn minh khác nhau một cách độc lập. Nó bắt đầu xuất hiện từ năm 2000 trước CN ở sông Hoàng Hà, Trung Quốc. Một số nhà khoa học đã tìm thấy bằng chứng chứng minh Caro đã được phát minh ở Hy lạp cổ đại và ở Châu Mỹ trước thời Colombo.

## Mô tả đề

~ TicTacToe là một phiên bản game cờ ca-rô đơn giản trên giao diện dòng lệnh (CLI). Game cho phép hai người chơi (người và máy tính hoặc hai người hoặc máy tính với máy tính) thay phiên đánh cờ trên một bảng 3x3. Game sử dụng các khái niệm lập trình hướng đối tượng (OOP), bao gồm Object & Class, Inheritance, Abstract Class. Điều chỉnh màu sắc Console cho phù hợp.

- Các bước thực hiện:

Tạo các lớp cơ sở (base classes) cho game: Board, Player, và Game.

Sử dụng kế thừa (Inheritance) để tạo các lớp con cho Player, như HumanPlayer và

ComputerPlayer.

Điều chỉnh Abstract Class để định nghĩa các phương thức chung cho các lớp con của

Player.

Xử lý ngoại lệ (Exception Handling) khi nhập dữ liệu từ người dùng và trong các hàm

của game (nhập sai số phạm vi 1 -> 9, số đã được chọn rồi).

Cấu trúc mã nguồn:

\_Board: Lớp chứa các thuộc tính và phương thức liên quan đến bảng chơi (kiểm tra

thắng/thua, vẽ bảng chơi).

\_Player: Lớp cơ bản chứa các thuộc tính và phương thức chung của người chơi.

\_HumanPlayer: Lớp kế thừa từ Player, thêm các thuộc tính và phương thức riêng cho

người chơi thực.

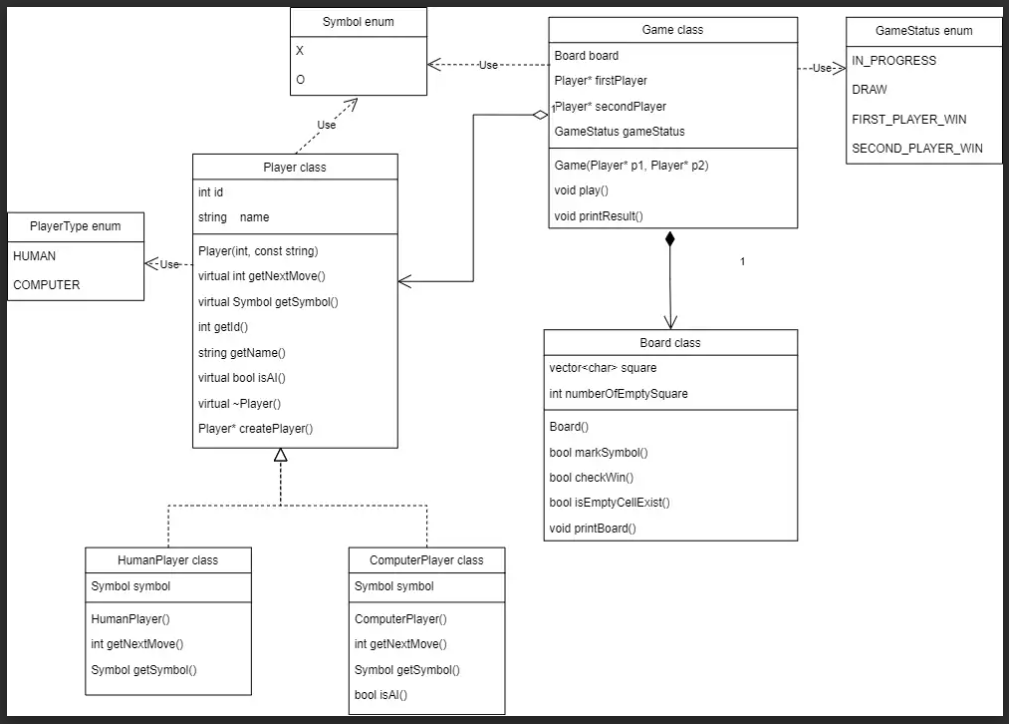
\_ComputerPlayer: Lớp kế thừa từ Player, thêm các thuộc tính và phương thức riêng

cho máy tính chơi.

\_Game: Lớp chứa các phương thức và thuộc tính để quản lý game, bao gồm cả hàm

Main().

## Sơ đồ lớp minh họa



## Dữ liệu kiểm thử

**INPUT:**

* Dòng 1: Nhập tên Player 1.
* Dòng 2: Chọn loại người chơi (human/computer), nếu là người chơi chọn y, ngược lại chọn n.
* Dòng 3: Nhập tên Player 2.
* Dòng 4: Chọn loại người chơi (human/computer), nếu là người chơi chọn y, ngược lại chọn n.

Sau đó bắt đầu ván cờ.

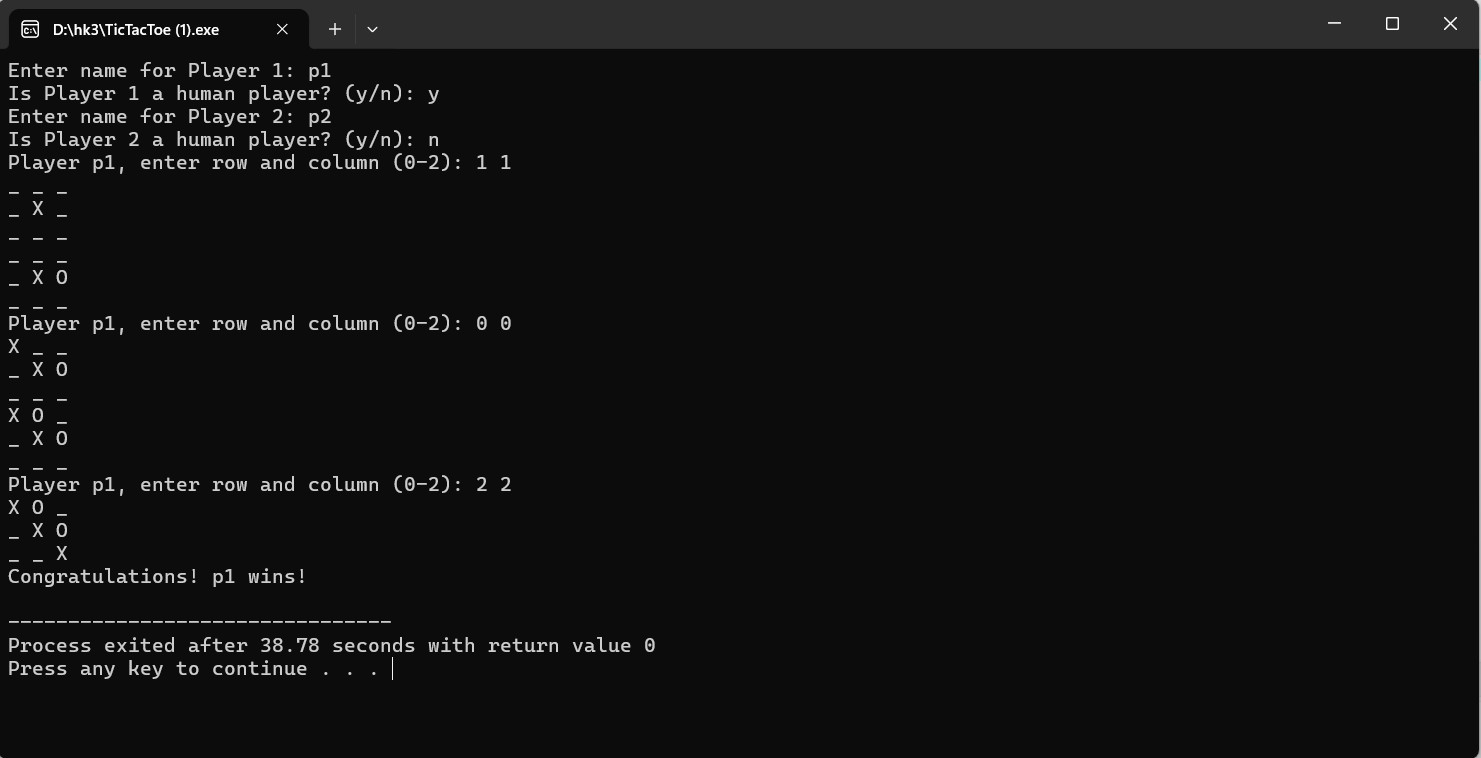
Máy tính sẽ yêu cầu nhập tọa độ của ô bạn sẽ đi (nếu là human).

**OUTPUT:**

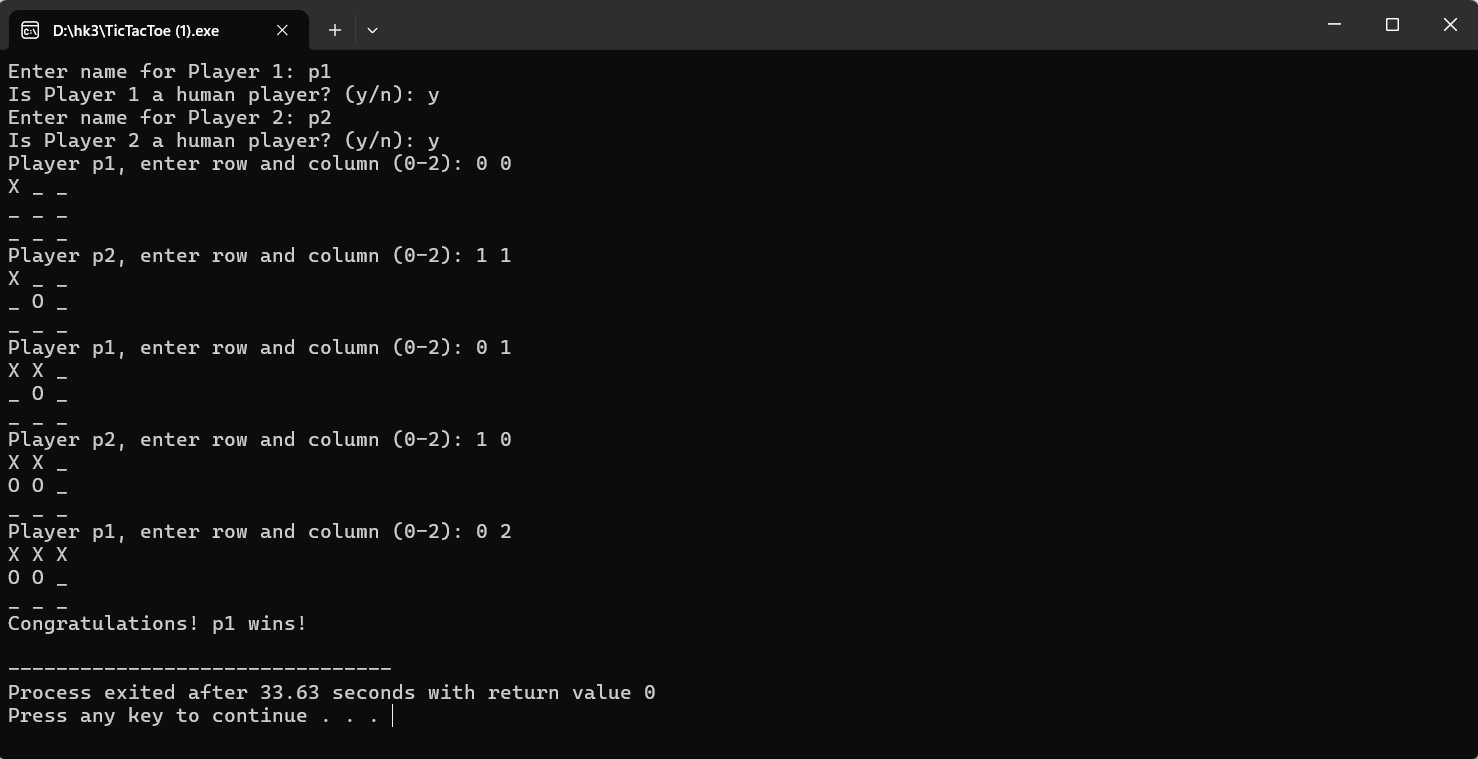
* Bàn cờ sau mỗi lượt đi.
* Thông báo kết quả người thắng/hòa sau khi bàn cờ đã đi hết các ô và không còn ô trống.

**Ví dụ:**

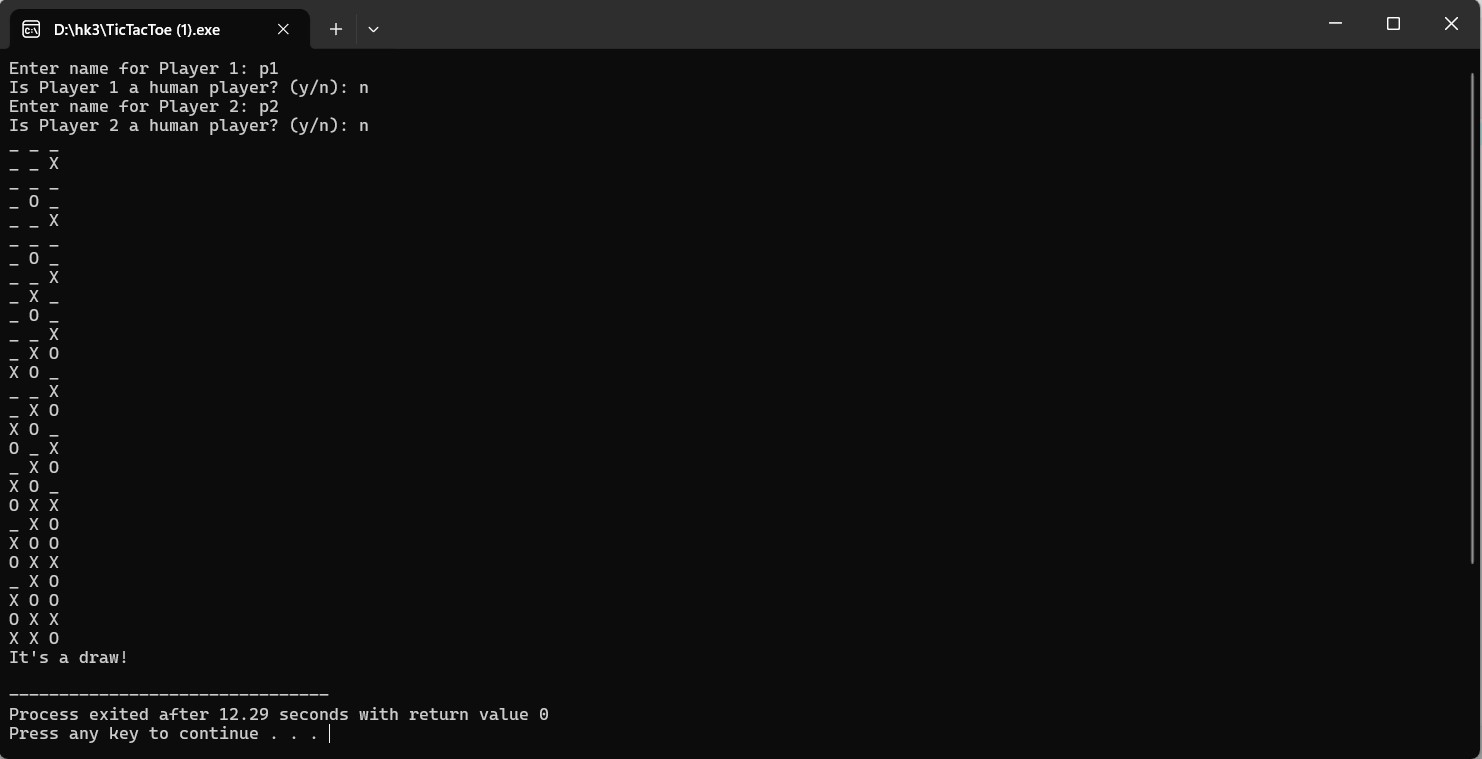
* Người chơi vs AI:



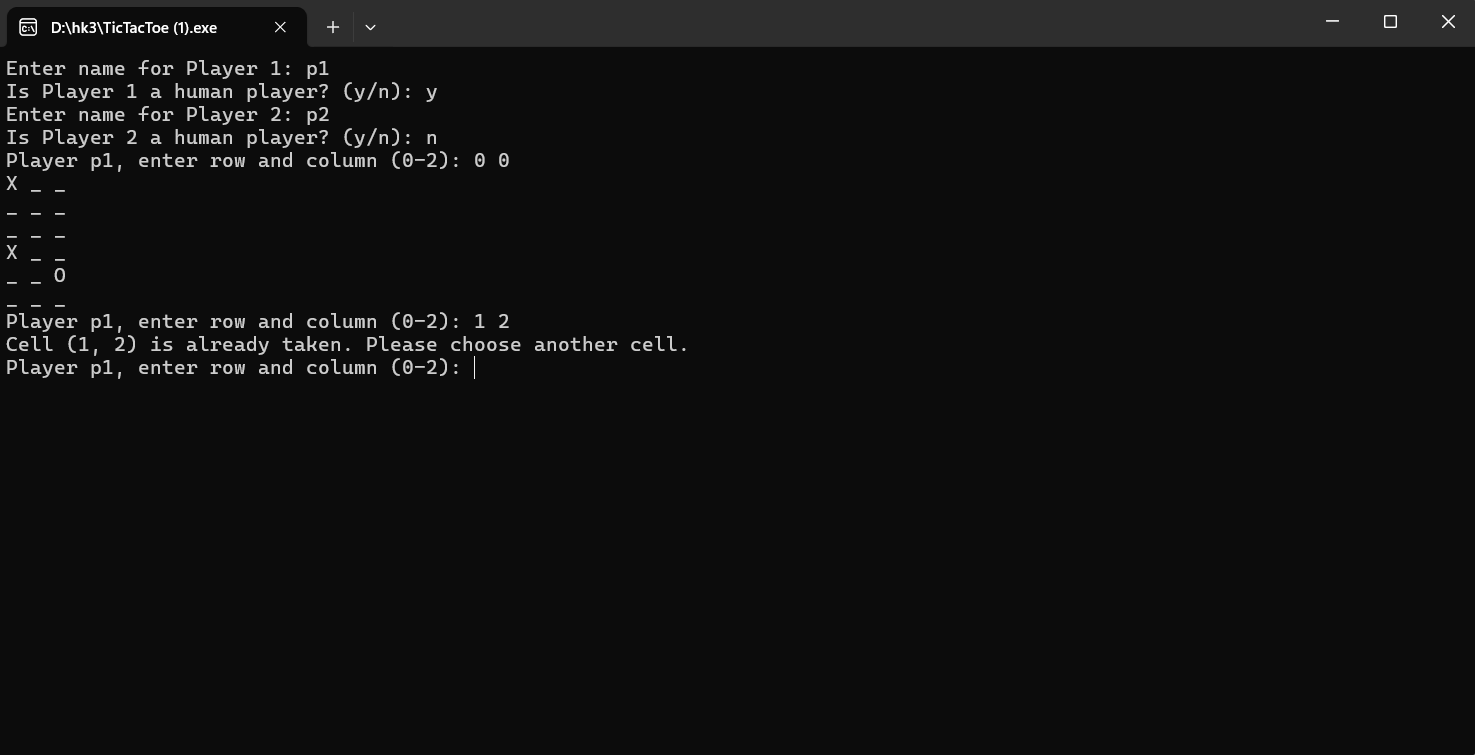
* Người chơi vs người chơi:



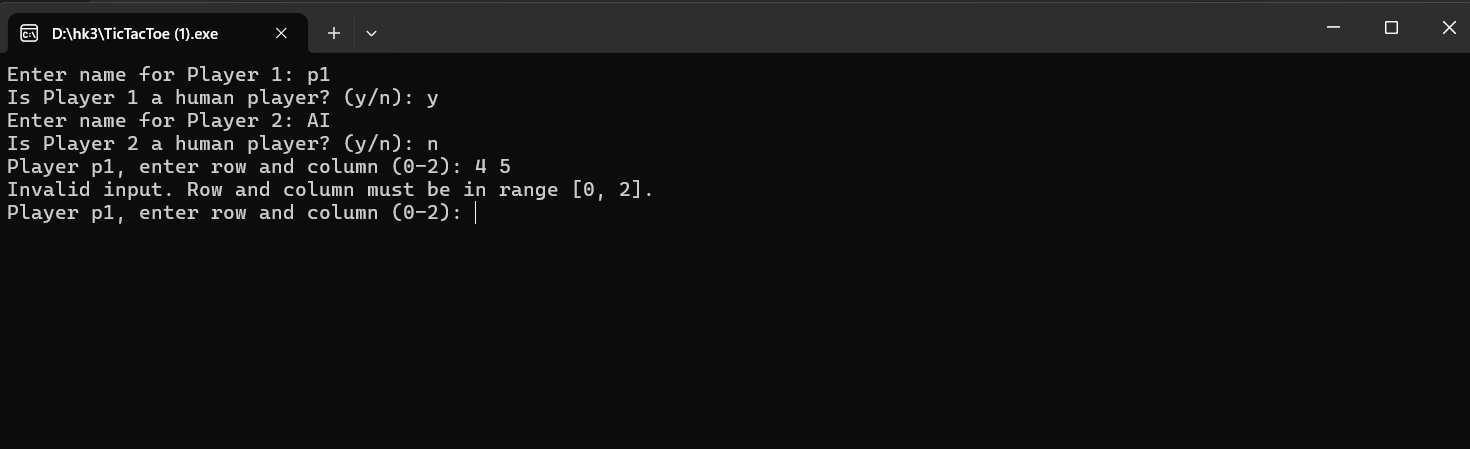
* AI vs AI



* Khi người chơi đánh vào ô đã được đánh trước đó



* Khi người chơi đánh vào ô nằm ngoài giới hạn



## Link Github và Code:

[**https://github.com/DUCTOAN1010/TIC-TAC-TOE.git**](https://github.com/DUCTOAN1010/TIC-TAC-TOE.git)

**https://drive.google.com/drive/folders/1ooxrElu78fcmrSJJIJPLz7ujhusgklsK?usp=drive\_link**

**Code**

#include <iostream>

#include <vector>

#include <cstdlib>

#include <ctime>

#include <random>

using namespace std;

// Li?t kê các bi?u tu?ng (Symbol) cho các ô trên b?ng

enum Symbol { X, O };

// Li?t kê tr?ng thái c?a trò choi

enum GameStatus { IN\_PROGRESS, DRAW, FIRST\_PLAYER\_WIN, SECOND\_PLAYER\_WIN };

// Li?t kê các lo?i ngu?i choi

enum PlayerType { HUMAN, COMPUTER };

// L?p Board d?i di?n cho b?ng trò choi

class Board {

private:

vector<char> square; // Vector ch?a các ô trên b?ng

int numberOfEmptySquare; // S? ô tr?ng trên b?ng

public:

// Constructor c?a Board

Board() : square(9, '\_'), numberOfEmptySquare(9) {}

// Phuong th?c d? dánh d?u ô trên b?ng v?i bi?u tu?ng (X ho?c O) tuong ?ng

bool markSymbol(int row, int col, Symbol symbol) {

int index = row \* 3 + col;

if (square[index] == '\_') {

square[index] = (symbol == X) ? 'X' : 'O';

numberOfEmptySquare--;

return true;

}

return false;

}

// Phuong th?c d? ki?m tra xem có ngu?i chi?n th?ng hay không

bool checkWin() {

for (int i = 0; i < 3; ++i) {

if (square[i] != '\_' && square[i] == square[i + 3] && square[i] == square[i + 6])

return true;

if (square[3 \* i] != '\_' && square[3 \* i] == square[3 \* i + 1] && square[3 \* i] == square[3 \* i + 2])

return true;

}

if (square[0] != '\_' && square[0] == square[4] && square[0] == square[8])

return true;

if (square[2] != '\_' && square[2] == square[4] && square[2] == square[6])

return true;

return false;

}

// Phuong th?c d? ki?m tra xem còn ô tr?ng nào trên b?ng hay không

bool isEmptyCellExist() {

return numberOfEmptySquare > 0;

}

// Phuong th?c d? in b?ng ra màn hình

void printBoard() {

for (int i = 0; i < 9; ++i) {

cout << square[i] << ' ';

if ((i + 1) % 3 == 0) cout << endl;

}

}

};

// L?p Player d?i di?n cho m?t ngu?i choi

class Player {

protected:

int id; // ID c?a ngu?i choi

string name; // Tên c?a ngu?i choi

public:

Player(int playerId, const string& playerName) : id(playerId), name(playerName) {}

// Phuong th?c ?o d? l?y bu?c di ti?p theo c?a ngu?i choi

virtual int getNextMove(Board& board, int& row, int& col) = 0;

// Phuong th?c ?o d? l?y bi?u tu?ng c?a ngu?i choi (X ho?c O)

virtual Symbol getSymbol() const = 0;

// Phuong th?c d? l?y ID c?a ngu?i choi

int getId() const { return id; }

// Phuong th?c d? l?y tên c?a ngu?i choi

string getName() const { return name; }

// Phuong th?c ?o d? ki?m tra xem ngu?i choi có ph?i là máy AI hay không

virtual bool isAI() const { return false; }

// H?y b? da hình

virtual ~Player() {}

};

// L?p HumanPlayer k? th?a t? l?p Player, d?i di?n cho ngu?i choi là con ngu?i

class HumanPlayer : public Player {

private:

Symbol symbol; // Bi?u tu?ng c?a ngu?i choi (X ho?c O)

public:

// Constructor c?a HumanPlayer

HumanPlayer(int playerId, const string& playerName, Symbol sym) : Player(playerId, playerName), symbol(sym) {}

// Phuong th?c d? ngu?i choi con ngu?i l?y bu?c di ti?p theo

int getNextMove(Board& board, int& row, int& col) override {

while (true) {

// Nh?p dòng và c?t t? ngu?i choi

cout << "Player " << getName() << ", enter row and column (0-2): ";

cin >> row >> col;

// Ki?m tra tính h?p l? c?a dòng và c?t nh?p vào

if (row < 0 || row > 2 || col < 0 || col > 2) {

cout << "Invalid input. Row and column must be in range [0, 2]." << endl;

continue;

}

// Ðánh d?u ô trên b?ng n?u ô dó chua du?c dánh d?u tru?c dó

if (board.markSymbol(row, col, symbol)) {

break;

} else {

cout << "Cell (" << row << ", " << col << ") is already taken. Please choose another cell." << endl;

}

}

return 0;

}

// Phuong th?c d? l?y bi?u tu?ng c?a ngu?i choi

Symbol getSymbol() const override {

return symbol;

}

};

// L?p ComputerPlayer k? th?a t? l?p Player, d?i di?n cho ngu?i choi là máy tính

class ComputerPlayer : public Player {

private:

Symbol symbol; // Bi?u tu?ng c?a ngu?i choi (X ho?c O)

public:

// Constructor c?a ComputerPlayer

ComputerPlayer(int playerId, const string& playerName, Symbol sym) : Player(playerId, playerName), symbol(sym) {}

// Phuong th?c d? ngu?i choi máy tính l?y bu?c di ti?p theo

int getNextMove(Board& board, int& row, int& col) override {

std::random\_device rd;

std::mt19937 gen(rd());

std::uniform\_int\_distribution<> distrib(0, 2);

do {

row = distrib(gen);

col = distrib(gen);

} while (!board.markSymbol(row, col, symbol));

return 0;

}

// Phuong th?c d? l?y bi?u tu?ng c?a ngu?i choi

Symbol getSymbol() const override {

return symbol;

}

// Phuong th?c d? ki?m tra xem ngu?i choi có ph?i là máy AI hay không

bool isAI() const override { return true; }

};

// L?p Game d?i di?n cho trò choi

class Game {

private:

Board board; // B?ng trò choi

Player\* firstPlayer; // Ngu?i choi th? nh?t

Player\* secondPlayer; // Ngu?i choi th? hai

GameStatus gameStatus; // Tr?ng thái c?a trò choi

public:

// Constructor c?a Game

Game(Player\* p1, Player\* p2) : firstPlayer(p1), secondPlayer(p2), gameStatus(IN\_PROGRESS) {}

// Phuong th?c d? th?c hi?n trò choi

void play() {

int row, col;

while (board.isEmptyCellExist() && gameStatus == IN\_PROGRESS) {

// Lu?t di c?a ngu?i choi th? nh?t

firstPlayer->getNextMove(board, row, col);

board.markSymbol(row, col, firstPlayer->getSymbol());

board.printBoard(); // In b?ng sau lu?t di c?a ngu?i choi 1

if (board.checkWin()) {

gameStatus = FIRST\_PLAYER\_WIN;

break;

}

if (!board.isEmptyCellExist()) {

gameStatus = DRAW;

break;

}

// Lu?t di c?a ngu?i choi th? hai

secondPlayer->getNextMove(board, row, col);

board.markSymbol(row, col, secondPlayer->getSymbol());

board.printBoard(); // In b?ng sau lu?t di c?a ngu?i choi 2

if (board.checkWin()) {

gameStatus = SECOND\_PLAYER\_WIN;

break;

}

}

}

// Phuong th?c d? in k?t qu? c?a trò choi ra màn hình

void printResult() const {

if (gameStatus == FIRST\_PLAYER\_WIN) {

cout << "Congratulations! " << firstPlayer->getName() << " wins!" << endl;

} else if (gameStatus == SECOND\_PLAYER\_WIN) {

cout << "Congratulations! " << secondPlayer->getName() << " wins!" << endl;

} else {

cout << "It's a draw!" << endl;

}

}

};

// Hàm t?o ngu?i choi d?a trên l?a ch?n c?a ngu?i dùng

Player\* createPlayer(int& playerId) {

string playerName;

cout << "Enter name for Player " << playerId << ": ";

cin >> playerName;

char playerTypeChoice;

cout << "Is Player " << playerId << " a human player? (y/n): ";

cin >> playerTypeChoice;

Symbol playerSymbol = (playerId == 1) ? X : O; // Ngu?i choi d?u tiên s? ch?n X, ngu?i choi th? hai s? ch?n O

if (tolower(playerTypeChoice) == 'y') {

return new HumanPlayer(playerId++, playerName, playerSymbol);

} else {

return new ComputerPlayer(playerId++, "Computer", playerSymbol);

}

}

// Hàm main

int main() {

int playerId = 1; // Kh?i t?o ID c?a ngu?i choi

Player\* p1 = createPlayer(playerId); // T?o ngu?i choi th? nh?t

Player\* p2 = createPlayer(playerId); // T?o ngu?i choi th? hai

// T?o trò choi v?i hai ngu?i choi dã du?c t?o

Game game(p1, p2);

// B?t d?u choi trò choi

game.play();

// In k?t qu? c?a trò choi ra màn hình

game.printResult();

// Gi?i phóng b? nh?

delete p1;

delete p2;

return 0;

}