TRƯỜNG ĐẠI HỌC KỸ THUẬT CÔNG NGHIỆP

KHOA ĐIỆN TỬ

Bộ môn: Công nghệ Thông tin.

BÀI TẬP KẾT THÚC MÔN HỌC

MÔN HỌC

LẬP TRÌNH PYTHON

Sinh viên: Lăng Nguyễn Minh Lượng

Lớp: K58KTP

Giáo viên GIẢNG DẠY: TS Nguyễn Văn Huy

Link GitHub: https://github.com/Minhluong999/BTL_PYTHON

Thái Nguyên – 2025

TRƯỜNG ĐHKTCN **CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM**

KHOA ĐIỆN TỬ ***Độc lập - Tự do - Hạnh phúc***

BÀI TẬP KẾT THÚC MÔN HỌC

MÔN HỌC: LẬP TRÌNH PYTHON

BỘ MÔN : CÔNG NGHỆ THÔNG TIN

Sinh viên: Lăng Nguyễn Minh Lượng

Lớp: ……. K58KTP…… Ngành : Kỹ thuật máy tính

Giáo viên hướng dẫn:TS Nguyễn Văn Huy

Ngày giao đề ..... Ngày hoàn thành: 10/06/2025 .....

Tên đề tài : **Tic-Tac-Toe GUI với OOP**

Yêu cầu :

- Đầu vào: Click vào ô vuông (Button).
- Đầu ra: X hoặc O hiện lên, hiển thị người thắng hoặc hoà.
- Theo dõi turn, legal_moves, winner.
- Reset game.
- Tắt nút sau khi click.

GIÁO VIÊN HƯỚNG DẪN

(Ký và ghi rõ họ tên)

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

Thái Nguyên, ngày tháng 06 năm 2025

GIÁO VIÊN HƯỚNG DẪN

(Ký ghi rõ họ tên)

Lời nói đầu

Trong thời đại công nghệ phát triển nhanh chóng, lập trình hướng đối tượng (Object-Oriented Programming – OOP) không chỉ là nền tảng trong việc xây dựng các ứng dụng phần mềm lớn mà còn là công cụ hữu hiệu để rèn luyện tư duy logic và khả năng tổ chức mã nguồn một cách khoa học. Để giúp sinh viên tiếp cận và ứng dụng hiệu quả các nguyên lý của OOP, các bài toán thực tế hoặc trò chơi cổ điển thường được lựa chọn làm ví dụ minh họa trực quan.

Trong đề tài này, em triển khai trò chơi **Tic-Tac-Toe (còn gọi là "Caro 3x3")** bằng ngôn ngữ Python, kết hợp giao diện đồ họa với thư viện Tkinter và áp dụng lập trình hướng đối tượng để tổ chức mã nguồn rõ ràng, dễ mở rộng. Trò chơi không chỉ đơn thuần là một sản phẩm giải trí, mà còn là một bài tập thực tiễn giúp củng cố các khái niệm OOP như class, đối tượng, kế thừa, và xử lý sự kiện trong GUI. Qua đó, đề tài góp phần nâng cao kỹ năng lập trình, khả năng phân tích và xây dựng phần mềm có tính tương tác.

QR github:

Mục Lục

Chương 1. Giới thiệu đề bài

1.1. Giới thiệu đề bài

Trong bài tập 5, sinh viên được yêu cầu xây dựng trò chơi Tic-Tac-Toe (Caro 3x3) sử dụng ngôn ngữ lập trình Python, kết hợp giữa lập trình hướng đối tượng (OOP) và giao diện đồ họa người dùng (GUI) với thư viện Tkinter. Đây là một bài toán đơn giản nhưng đầy tính thực tiễn, giúp người học luyện tập cách tổ chức logic chương trình thông qua class và đối tượng, đồng thời làm quen với lập trình tương tác qua giao diện đồ họa.

Trò chơi có dạng bảng 3×3, nơi hai người chơi lần lượt đánh dấu X hoặc O vào các ô vuông bằng cách click chuột. Sau mỗi lượt chơi, chương trình phải kiểm tra điều kiện thắng/thua hoặc hòa và hiển thị kết quả tương ứng. Ngoài ra, ứng dụng cần hỗ trợ tính năng reset để chơi lại và đảm bảo mỗi ô chỉ được click một lần duy nhất.

Bài tập yêu cầu sinh viên triển khai đầy đủ các tính năng cơ bản như theo dõi lượt chơi (turn), danh sách nước đi hợp lệ (legal_moves), người chiến thắng (winner), cùng với các chức năng giao diện như cập nhật nội dung nút sau mỗi lượt và xử lý kết thúc trận đấu. Việc thiết kế được thực hiện thông qua mô hình hướng đối tượng, trong đó class TTTBoard sẽ chịu trách nhiệm xử lý logic chính của trò chơi.

1.2. Các tính năng chính của chương trình

Chương trình Tic-Tac-Toe sử dụng giao diện Tkinter với bảng chơi 3x3 gồm 9 nút bấm, cho phép hai người chơi luân phiên đánh dấu X hoặc O. Mỗi ô chỉ được chọn một lần và sẽ bị vô hiệu hóa sau khi click. Sau mỗi lượt, chương trình tự động kiểm tra điều kiện thắng hoặc hòa và hiển thị kết quả. Trò chơi hỗ trợ nút “Reset” để bắt đầu lại ván mới. Toàn bộ logic được tổ chức theo mô hình lập trình hướng đối tượng thông qua lớp TTTBoard, giúp mã nguồn rõ ràng, dễ mở rộng và bảo trì.

1.3. Kiến thức được áp dụng

Trong quá trình xây dựng trò chơi Tic-Tac-Toe, sinh viên đã vận dụng nhiều kiến thức cơ bản và nâng cao đã học trong môn Lập trình Python. Cụ thể, chương trình sử dụng **lập trình hướng đối tượng (OOP)** để tổ chức mã nguồn thông qua các lớp, thuộc tính và phương thức. Các khái niệm như khởi tạo (__init__), xử lý sự kiện, và quản lý trạng thái được áp dụng hiệu quả. Bên cạnh đó, chương trình tận dụng **thư viện Tkinter** để tạo giao diện người dùng, bao gồm các nút bấm, xử lý click chuột, và cập nhật nội dung động. Ngoài ra, sinh viên còn áp dụng kiến thức về **cấu trúc điều kiện, vòng lặp, danh sách** và **xử lý logic trò chơi**, giúp kiểm tra kết quả sau mỗi lượt chơi. Qua bài tập này, sinh viên không chỉ củng cố kiến thức nền tảng mà còn rèn luyện kỹ năng thiết kế chương trình có tính tương tác và dễ mở rộng.

1.4. Thách thức khi làm bài

Trong quá trình thực hiện bài tập, em đã gặp một số thách thức đáng kể. Trước tiên là việc **thiết kế giao diện với Tkinter**, do đây là lần đầu tiếp xúc với lập trình GUI nên cần thời gian để hiểu cách tạo và sắp xếp các widget như Button, Label, và xử lý sự kiện click chuột. Thứ hai, việc **xây dựng logic kiểm tra điều kiện thắng** cho ba hàng, ba cột và hai đường chéo đòi hỏi sự chính xác trong lập trình và tư duy logic tốt. Ngoài ra, nhóm cũng gặp khó khăn trong **việc quản lý trạng thái trò chơi** như lượt chơi hiện tại, danh sách nước đi hợp lệ, và xử lý reset game sao cho không ảnh hưởng đến logic tổng thể. Cuối cùng, việc **tổ chức chương trình theo hướng đối tượng** là một thử thách, đòi hỏi sự hiểu rõ về cách sử dụng lớp, phương thức, và cách phân chia trách nhiệm giữa giao diện và xử lý logic. Tuy nhiên, chính những thách thức này đã giúp nhóm nâng cao kỹ năng và hiểu sâu hơn về cách xây dựng một ứng dụng Python hoàn chỉnh.

Chương 2 : Cơ sở lý thuyết

2.1. Lập trình hướng đối tượng (OOP)

- Là phương pháp tổ chức chương trình dựa trên các đối tượng (object) và lớp (class).
- Giúp phân chia rõ ràng giữa phần xử lý logic và giao diện, dễ bảo trì và mở rộng.
- Các khái niệm chính được áp dụng: class, phương thức (def), thuộc tính (self), và khởi tạo (__init__).

2.2. Thư viện Tkinter – Giao diện đồ họa (GUI)

- Tkinter là thư viện chuẩn của Python dùng để xây dựng giao diện người dùng.
- Trong đề tài, Tkinter được dùng để tạo các nút Button, khung Frame, xử lý sự kiện khi người chơi click chuột, và hiển thị kết quả trò chơi.

2.3. Cấu trúc điều kiện và vòng lặp

- Dùng để kiểm tra điều kiện thắng/thua, luân phiên lượt chơi giữa X và O, và xử lý khi người dùng chọn nút reset.

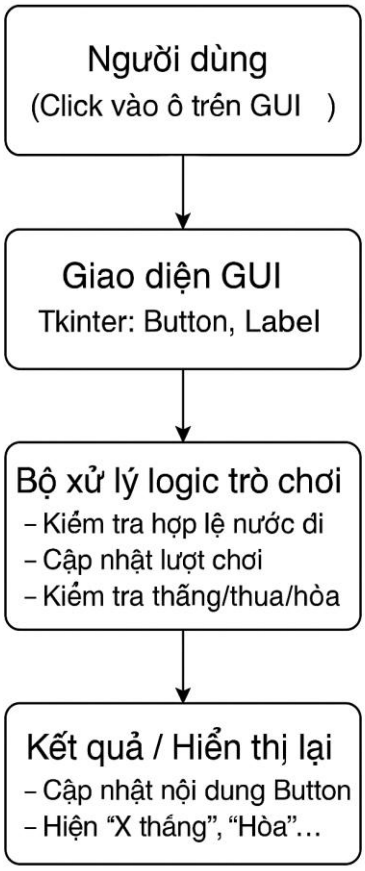
2.4. Tư duy giải thuật và xử lý logic trò chơi

- Tính toán điều kiện chiến thắng theo hàng, cột, đường chéo.
- Quản lý trạng thái lượt chơi (turn), các nước đi hợp lệ (legal_moves), và xác định người thắng (winner).

Chương 3 : Thuyết kế và xây dựng chương trình

3.1. Sơ đồ khối hệ thống.

Sơ đồ chia làm 3 khối chính



a) Các modul chính trong chương trình

Module TTTBoard (Xử lý logic trò chơi)

Đây là lớp trung tâm xử lý toàn bộ logic của trò chơi. Nó bao gồm:

- Khởi tạo bảng chơi dưới dạng danh sách 2 chiều.
- Theo dõi lượt chơi hiện tại (X hoặc O).
- Kiểm tra hợp lệ của nước đi.

- Cập nhật trạng thái bảng sau mỗi lượt.
- Kiểm tra điều kiện thắng cuộc hoặc hòa.
- Cung cấp phương thức reset để bắt đầu lại trò chơi.

Module Giao diện GUI (Tkinter)

Sử dụng thư viện Tkinter để tạo và quản lý giao diện người dùng:

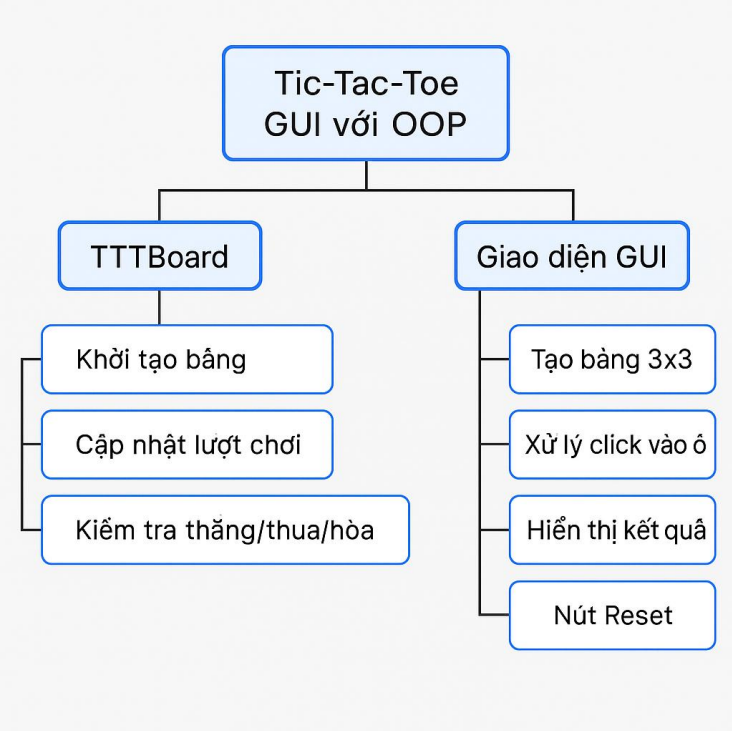
- Tạo bảng 3x3 bằng các nút Button.
- Bắt sự kiện click chuột vào từng ô.
- Cập nhật nội dung nút (hiển thị X hoặc O) theo lượt chơi.
- Hiển thị kết quả: “X thắng”, “O thắng” hoặc “Hòa”.
- Cung cấp nút Reset để người dùng chơi lại ván mới.

Module điều phối (Main loop)

Là phần khởi chạy chương trình:

- Tạo cửa sổ chính của Tkinter.
- Khởi tạo đối tượng TTTBoard.
- Liên kết giao diện với logic game.
- Duy trì vòng lặp chính (mainloop()) để cập nhật giao diện liên tục.

b) Biểu đồ phân cấp chức năng



3.2. Sơ đồ khối các thuật toán chính

Khởi tạo chương trình

- Tạo cửa sổ GUI bằng Tkinter
- Tạo đối tượng board (class TTTBoard)
- Gán các biến ban đầu:

```
turn = "X"
board = [ ["", "", ""],
          ["", "", ""],
          ["", "", ""] ]
winner = None
```

Xử lý sự kiện click vào ô

Khi người dùng click vào một ô:

Nếu ô đó còn trống và chưa có người thắng:

- Ghi giá trị turn ("X" hoặc "O") vào ô
- Cập nhật giao diện (hiển thị X hoặc O)
- Gọi hàm kiểm tra người thắng
- Nếu có người thắng:
  - Lưu giá trị winner
  - Hiển thị thông báo thắng
  - Vô hiệu hóa tất cả các nút
- Nếu không có người thắng:
  - Kiểm tra hòa (nếu không còn ô trống)
  - Nếu hòa → hiển thị thông báo "Hòa!"
- Đổi lượt chơi (X ↔ O)

Thuật toán kiểm tra thắng cuộc

- Tạo danh sách các dòng cần kiểm tra:
  - 3 hàng ngang
  - 3 cột dọc
  - 2 đường chéo
- Với mỗi dòng:

  Nếu cả 3 ô giống nhau và khác rỗng:
  - Trả về giá trị người thắng (X hoặc O)
  - Kết thúc kiểm tra
- Nếu không có ai thắng → tiếp tục chơi

Hàm kiểm tra hòa

- Duyệt toàn bộ bảng:

  Nếu còn ít nhất 1 ô rỗng:

  - Không hòa → tiếp tục chơi

  Nếu không còn ô nào rỗng và chưa có người thắng:

  - Trò chơi hòa → hiển thị thông báo

Reset trò chơi

Khi người dùng bấm nút “Reset”:

- Đặt lại giá trị turn = "X"
- Làm rỗng toàn bộ bảng
- Xóa thông báo kết quả (nếu có)
- Kích hoạt lại tất cả các nút

3.3. Cấu trúc dữ liệu.

- **Danh sách 1 chiều board**

Khai báo: self.board = [" "] * 9

Vai trò: Lưu trạng thái của 9 ô trên bàn cờ (3x3), với chỉ số từ 0 đến 8.

Ý nghĩa phần tử:

- " " : ô trống
- "X" : người chơi
- "O" : máy

- **Biến chuỗi turn**

Khai báo: self.turn = self.human

Vai trò: Xác định lượt chơi hiện tại là của "X" hay "O".

- **Danh sách buttons**

Khai báo: self.buttons = []

Vai trò: Lưu trữ 9 nút Button tương ứng với các ô trên giao diện Tkinter.

Sử dụng: Cho phép cập nhật nội dung nút (text), trạng thái (state), và reset khi cần.

- **Tuple WAYS_TO_WIN**

Khai báo:

```
WAYS_TO_WIN = [
    (0, 1, 2), (3, 4, 5), (6, 7, 8),
    (0, 3, 6), (1, 4, 7), (2, 5, 8),
    (0, 4, 8), (2, 4, 6)
]
```

Vai trò: Xác định các tổ hợp chiến thắng hợp lệ (theo hàng, cột, chéo).

- **Danh sách BEST trong AI**

Khai báo: BEST = [4, 0, 2, 6, 8, 1, 3, 5, 7]

Vai trò: Định hướng chiến lược chơi của AI (ưu tiên giữa → góc → cạnh).

3.4. Các hàm trong chương trình.

Lớp TTTBoard – Xử lý logic trò chơi

Tên hàm	Chức năng
__init__()	Khởi tạo bảng, lượt chơi, người và máy.
make_move(index, piece)	Đánh dấu vào ô chỉ định nếu còn trống.
legal_moves()	Trả về danh sách các ô còn trống (hợp lệ).
check_winner()	Kiểm tra và trả về người thắng, hòa hoặc None.
computer_move()	Trả về nước đi tối ưu của AI (ưu tiên thắng, chặn, chọn tốt nhất).
_check_winner (temp_board)	Hàm phụ hỗ trợ kiểm tra thắng trên bảng tạm.
reset()	Đặt lại bảng chơi và lượt chơi ban đầu.

Lớp TicTacToeGUI – Giao diện người dùng

Tên hàm	Chức năng
__init__(root)	Khởi tạo giao diện, nút bấm, trạng thái, gắn sự kiện.
build_grid()	Tạo bảng 3x3 gồm 9 nút Button.
human_click(index)	Xử lý sự kiện người chơi click vào ô.
computer_play()	Gọi AI đánh nước tiếp theo, cập nhật giao diện.
update_game_state()	Cập nhật trạng thái sau mỗi lượt, kiểm tra kết thúc.

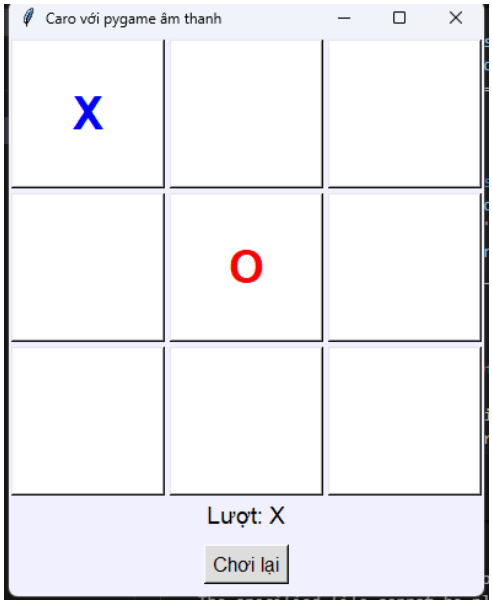
Tên hàm	Chức năng
disable_all_buttons()	Vô hiệu hóa toàn bộ nút khi game kết thúc.
reset_game()	Đặt lại bảng và trạng thái game khi người chơi bấm Reset.

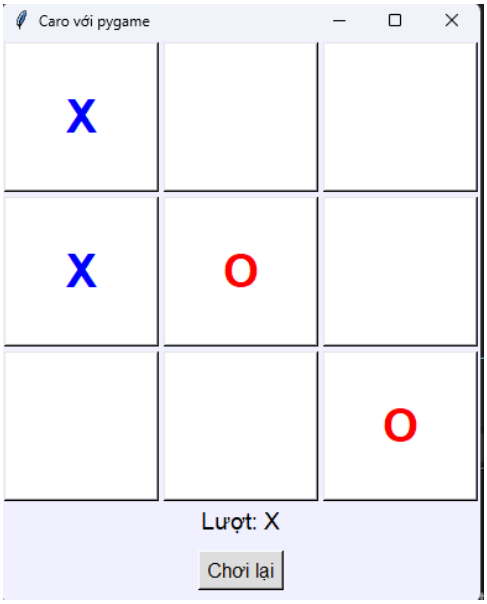
Chương 4 : Thực nghiệm và kết luận

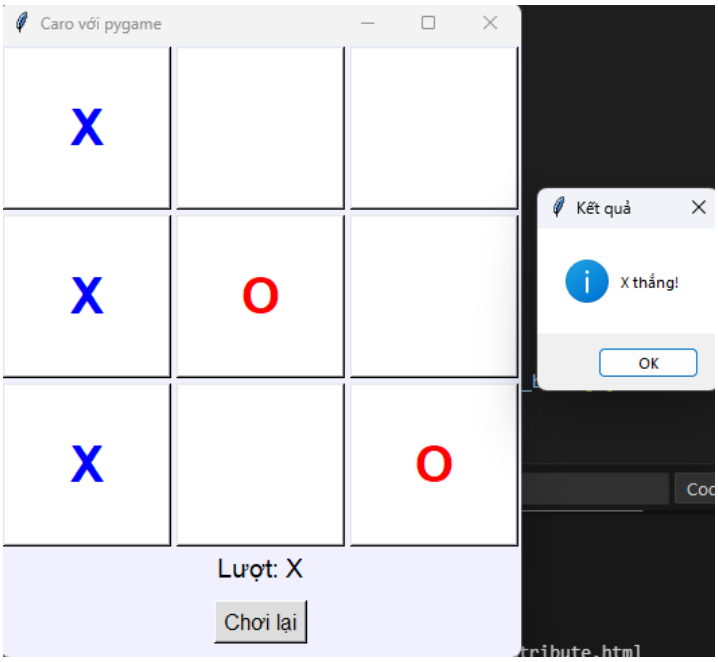
4.1. Thực nghiệm.

Sau khi hoàn thiện chương trình, tôi đã tiến hành chạy thử và kiểm tra lần lượt các chức năng chính của ứng dụng ***Tic-Tac-Toe GUI với OOP*** . Các kết quả thực nghiệm được ghi lại như sau:

Chạy thử lần 1 (X đi trước) :



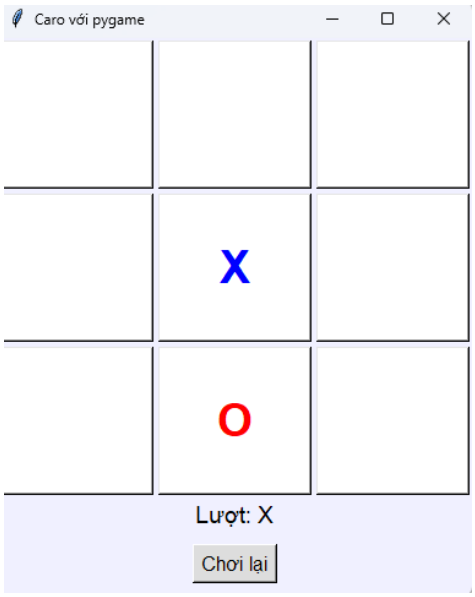


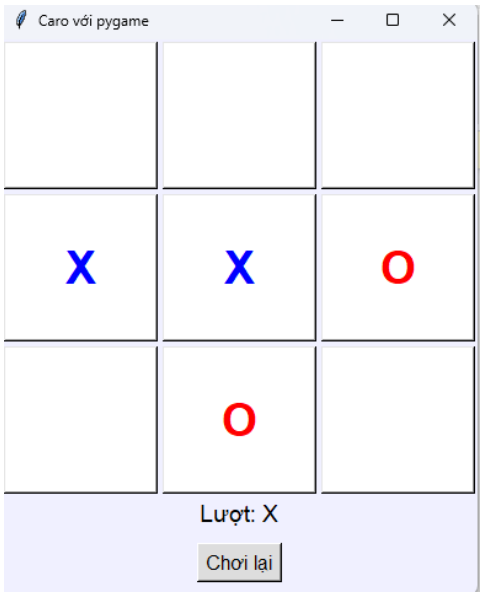


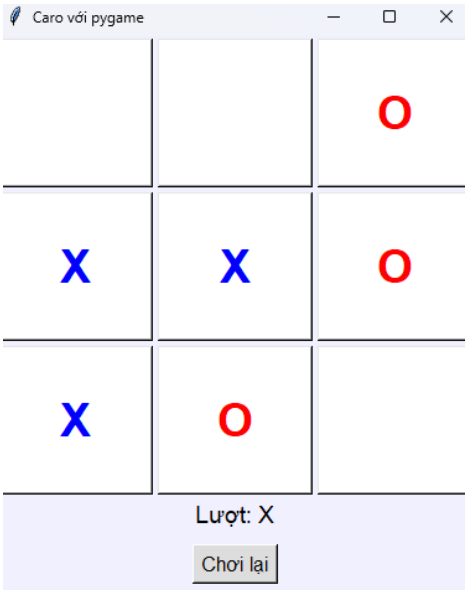
X đi (1 , 4 , 7) hoàn thành 1 hàng dọc đạt điều kiện thắng

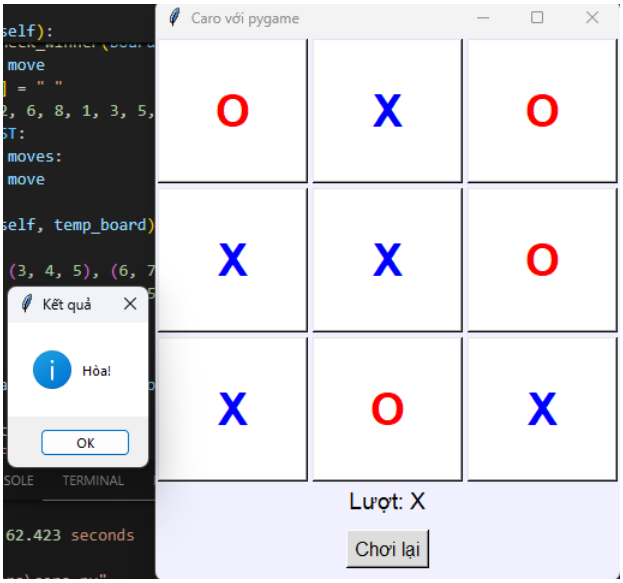
⇨ Kết quả X thắng

Chạy thử lần 2 (X đi trước) :









Cả X và O không đạt điều kiện thắng

⇨ Kết quả Hòa

Từ 2 lần chạy thử chương trình tôi thấy các tính năng đã tương đối đáp ứng được yêu cầu của đề bài

- Tính năng theo dõi turn đã hoàn thành với đối tượng X luôn đi trước rồi đến đối tượng O . Luân phiên đánh dấu đến khi không còn ô nào trống trong bảng trò chơi 3x3
- Label luôn hiển thị rõ ràng hiện đang là lượt đi của X hoặc O
- Hàm legal_moves() trong class TTTBoard:

```
def legal_moves(self):
    return [i for i, val in enumerate(self.board) if val == " "]
```

Dùng để kiểm tra ô còn trống và xác định nước đi hợp lệ .

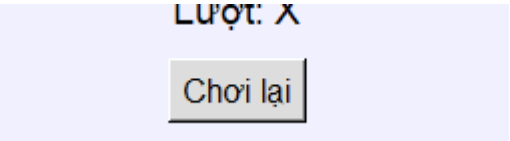
- Hàm check_winner() kiểm tra các tổ hợp thắng hoặc hòa:

```
def check_winner(self):
```

```
            ...
            if self.board[a] == self.board[b] == self.board[c] != " ":
                return self.board[a]
    ...
```

→ Kết quả thắng "X", "O" hoặc "Hòa" sẽ được thông báo bằng messagebox.showinfo(...).

- Nút "Chơi lại" gọi hàm reset_game():
    - Gọi self.board.reset()
    - Làm mới trạng thái nút, trạng thái game



- Tắt nút sau khi click

Ngay sau mỗi nước đi, nút được tắt bằng:

```
self.buttons[index].config(text=..., state="disabled")
```

→ Không thể bấm lại ô đã chơi.

4.2. Code

```
import tkinter as tk
from tkinter import messagebox
import pygame


class TTTBoard:
    def __init__(self):
        self.board = [" "] * 9
```

```
    self.turn = "X"


def make_move(self, index):
    if self.board[index] == " ":
        self.board[index] = self.turn
        return True
    return False


def legal_moves(self):
    return [i for i, val in enumerate(self.board) if val == " "]


def check_winner(self):
    WAYS = [
        (0, 1, 2), (3, 4, 5), (6, 7, 8),
        (0, 3, 6), (1, 4, 7), (2, 5, 8),
        (0, 4, 8), (2, 4, 6)
    ]
    for a, b, c in WAYS:
        if self.board[a] == self.board[b] == self.board[c] != " ":
            return self.board[a]
    if " " not in self.board:
        return "TIE"
    return None


def switch_turn(self):
```

```
        self.turn = "O" if self.turn == "X" else "X"


    def reset(self):
        self.board = [" "] * 9
        self.turn = "X"


    def computer_move(self):
        board = self.board[:]
        moves = self.legal_moves()


        for move in moves:
            board[move] = "O"
            if self._check_winner(board) == "O":
                return move
            board[move] = " "
        for move in moves:
            board[move] = "X"
            if self._check_winner(board) == "X":
                return move
            board[move] = " "
        BEST = [4, 0, 2, 6, 8, 1, 3, 5, 7]
        for move in BEST:
            if move in moves:
                return move
```

```
    def _check_winner(self, temp_board):
        WAYS = [
            (0, 1, 2), (3, 4, 5), (6, 7, 8),
            (0, 3, 6), (1, 4, 7), (2, 5, 8),
            (0, 4, 8), (2, 4, 6)
        ]
        for a, b, c in WAYS:
            if temp_board[a] == temp_board[b] == temp_board[c] != " ":
                return temp_board[a]
        if " " not in temp_board:
            return "TIE"
        return None


class TicTacToeGUI:
    def __init__(self, root):
        self.root = root
        self.root.title("Caro với pygame ")
        self.board = TTTBoard()
        self.buttons = []
        self.game_over = False
        self.mode = self.ask_mode()


        pygame.init()
        pygame.mixer.init()
```

```
        self.build_grid()


        self.status_label = tk.Label(root, text="Lượt: X", font=("Arial", 14),
bg="#f0f0ff")
        self.status_label.grid(row=3, column=0, columnspan=3, sticky="nsew")


        self.reset_button = tk.Button(root, text="Chơi lại", command=self.reset_game,
bg="#ddd", font=("Arial", 12))
        self.reset_button.grid(row=4, column=0, columnspan=3, pady=10)


    def ask_mode(self):
        answer = messagebox.askquestion("Chế độ chơi", "Bạn muốn chơi với máy?")
        return "AI" if answer == "yes" else "PVP"


    def build_grid(self):
        for i in range(9):
            btn = tk.Button(
                self.root, text=" ", font=("Helvetica", 28, "bold"), width=5, height=2,
                bg="#ffffff", activebackground="#dff6f0",
                command=lambda i=i: self.handle_click(i)
            )
            btn.grid(row=i//3, column=i%3, padx=2, pady=2)
            self.buttons.append(btn)


    def play_sound(self, filename):
        try:
```

```
            pygame.mixer.music.load(filename)
            pygame.mixer.music.play()
        except Exception as e:
            print("Lỗi âm thanh:", e)


    def handle_click(self, index):
        if self.game_over:
            return
        if self.board.make_move(index):
            self.buttons[index].config(
                text=self.board.turn,
                state="disabled",
                disabledforeground="blue" if self.board.turn == "X" else "red"
            )
            self.play_sound("click.mp3")
            winner = self.board.check_winner()
            if winner:
                self.end_game(winner)
            else:
                if self.mode == "PVP":
                    self.board.switch_turn()
                    self.status_label.config(text=f"Lượt: {self.board.turn}")
                else:
                    self.board.switch_turn()
                    self.status_label.config(text="Máy đang nghĩ...")
```

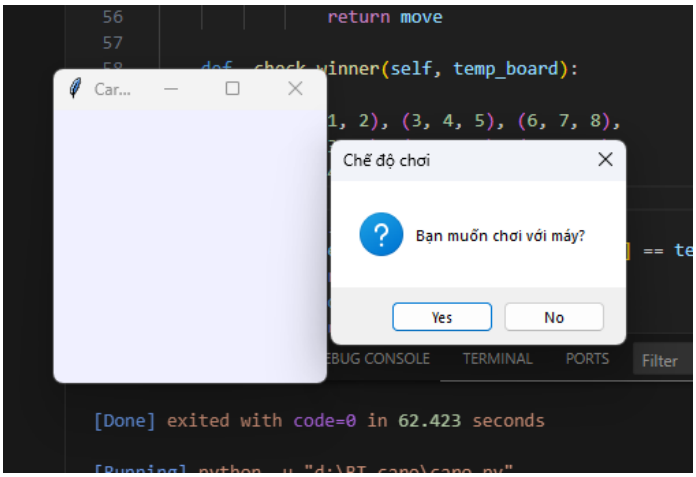
```
            self.root.after(500, self.computer_play)


def computer_play(self):
    if self.game_over:
        return
    move = self.board.computer_move()
    if move is not None:
        self.board.make_move(move)
        self.buttons[move].config(
            text="O", state="disabled", disabledforeground="red"
        )
        self.play_sound("click.mp3")
        winner = self.board.check_winner()
        if winner:
            self.end_game(winner)
        else:
            self.board.switch_turn()
            self.status_label.config(text=f"Lượt: {self.board.turn}")


def end_game(self, winner):
    self.game_over = True
    if winner == "TIE":
        self.play_sound("tie.mp3")
        messagebox.showinfo("Kết quả", "Hòa!")
        self.status_label.config(text="Hòa!")
```

```
        else:
            self.play_sound("win.mp3")
            messagebox.showinfo("Kết quả", f"{winner} thắng!")
            self.status_label.config(text=f"{winner} thắng!")
        self.disable_all_buttons()

    def disable_all_buttons(self):
        for btn in self.buttons:
            btn.config(state="disabled")
    def reset_game(self):
        self.board.reset()
        self.game_over = False
        for btn in self.buttons:
            btn.config(text=" ", state="normal", bg="#ffffff")
        self.status_label.config(text="Lượt: X")
        self.mode = self.ask_mode()

if __name__ == "__main__":
    root = tk.Tk()
    root.configure(bg="#f0f0ff")
    for i in range(3):
        root.grid_columnconfigure(i, weight=1)
        root.grid_rowconfigure(i, weight=1)
    game = TicTacToeGUI(root)
    root.mainloop()
```

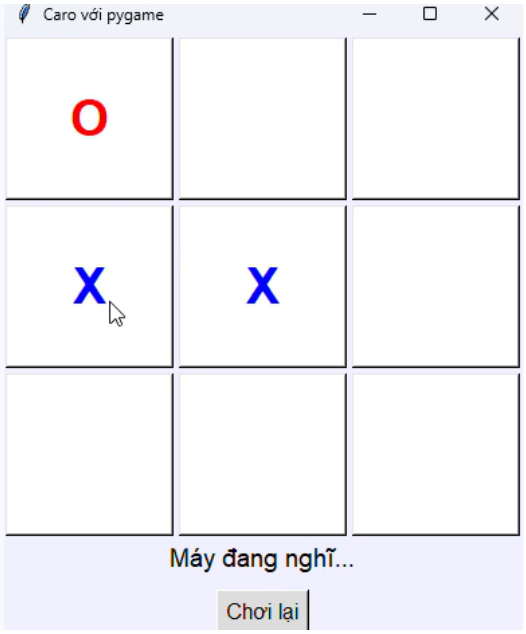
Trong phần code này chương trình không chỉ còn 2 người chơi với nhau mà tôi đã nâng cấp phần PvE (người chơi với AI)

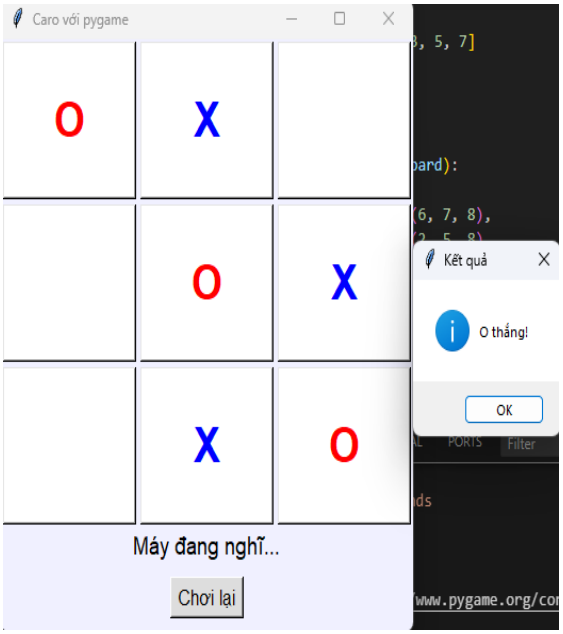
Khi chạy chương trình sẽ có thông báo chọn chế đồ chơi

- Yes để chơi với AI
- No để 2 người chơi với nhau

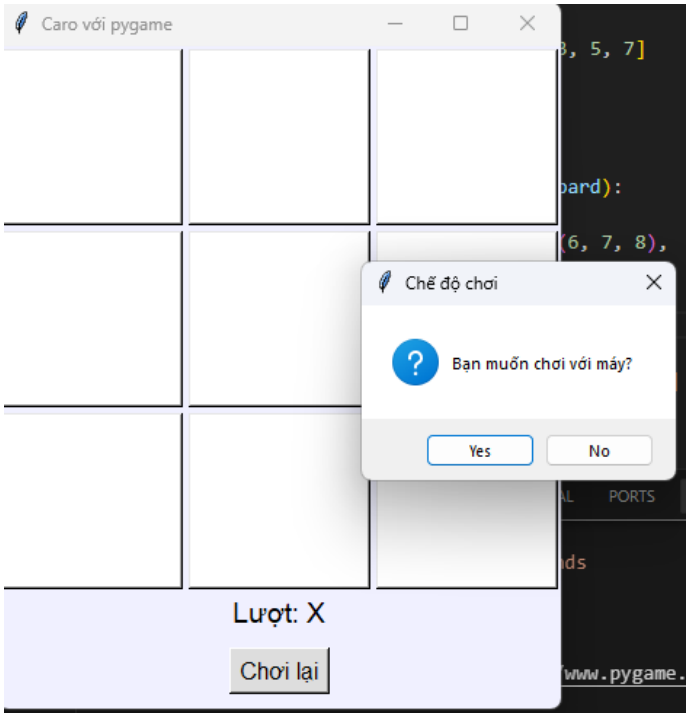


Khi vào chế độ PvE bạn sẽ đi trước với X





Khi ấn chơi lại cửa sổ chọn chế độ chơi lại hiện lên



4.3. Kết luận

4.3.1. Những gì sản phẩm làm được

Sản phẩm trò chơi **Tic-Tac-Toe (Caro 3x3)** được xây dựng bằng ngôn ngữ Python với thư viện tkinter và pygame, cho phép người chơi lựa chọn chế độ chơi với máy (AI) hoặc chơi đối kháng 2 người. Trò chơi đáp ứng đầy đủ các yêu cầu chức năng như theo dõi lượt chơi, kiểm tra nước đi hợp lệ, xác định người thắng hoặc hòa, và vô hiệu hóa nút đã chọn. Giao diện trực quan, dễ sử dụng với thiết kế nút đẹp, phân biệt màu sắc giữa người chơi X và O. Ngoài ra, sản phẩm còn tích hợp hiệu ứng âm thanh sống động, phát tiếng khi người chơi thực hiện nước đi, khi trò chơi kết thúc với kết quả thắng hoặc hòa, mang lại trải nghiệm sinh động và

hấp dẫn. Người chơi có thể bấm nút "Chơi lại" để bắt đầu ván mới và chọn lại chế độ chơi. Sản phẩm hoạt động ổn định, thân thiện với người dùng và có thể dễ dàng mở rộng trong tương lai.

4.3.2. Những gì đã học được.

Sau khi hoàn thành sản phẩm trò chơi Tic-Tac-Toe, em đã học được cách xây dựng giao diện người dùng bằng thư viện Tkinter, xử lý logic game theo hướng đối tượng, và quản lý luồng chơi một cách chính xác. Em hiểu rõ hơn về cách kiểm tra lượt chơi, xác định nước đi hợp lệ, kiểm tra điều kiện thắng hoặc hòa, cũng như cách thiết kế giao diện trực quan, dễ sử dụng. Bên cạnh đó, em còn biết cách tích hợp âm thanh hiệu ứng bằng thư viện Pygame để tăng trải nghiệm người dùng. Trong quá trình làm, em rèn luyện được kỹ năng phân tích lỗi, tự cài đặt và sử dụng thư viện ngoài, đồng thời nâng cao khả năng tư duy hệ thống và lập trình hướng đối tượng. Dự án giúp em tự tin hơn trong việc triển khai các ứng dụng có giao diện đồ họa và xử lý tương tác thực tế.

4.3.3. Cải tiến trong tương lai.

Trong tương lai, em có thể cải tiến sản phẩm theo nhiều hướng để nâng cao trải nghiệm người chơi và mở rộng tính năng. Trước hết, có thể **nâng cấp AI bằng thuật toán Minimax** để máy chơi thông minh hơn và không thể bị đánh bại. Bên cạnh đó, việc **ghi điểm và hiển thị bảng xếp hạng** sẽ tạo tính cạnh tranh, hấp dẫn hơn cho người chơi. Em cũng có thể **thiết kế giao diện tùy biến** như cho phép người chơi chọn biểu tượng, màu sắc hoặc kích thước bảng cờ. Ngoài ra, có thể **xuất bản trò chơi thành ứng dụng .exe hoặc bản web** để dễ dàng chia sẻ và chơi trên nhiều nền tảng. Cuối cùng, việc thêm **âm thanh nền, hiệu ứng khi thắng**, hoặc các chế độ chơi khác như 5x5 cũng là những cải tiến thú vị để trò chơi trở nên chuyên nghiệp và hấp dẫn hơn.