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

HỌ TÊN	**:**	**NGUYỄN HOÀNG VIỆT**
MSV	**:**	**K225480106074**
LỚP	**:**	**K58KTP**
GIÁO VIÊN GIẢNG DẠY	**:**	**TS. NGUYỄN VĂN HUY**

Thái Nguyên – 2025

TRƯỜNG ĐHKTCN
KHOA ĐIỆN TỬ

CỘNG HOÀ XÃ HỘI CHỦ NGHĨA VIỆT NAM
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: Nguyễn Hoàng Việt Ngành: Kỹ thuật máy tính

MSV: K225480106074

Lớp: K58KTP

Giáo viên hướng dẫn: TS. Nguyễn Văn Huy

Ngày giao đề: 20/05/2025

Ngày hoàn thành: 09/06/2025

Đầu bài:

Triển khai game Tic-Tac-Toe (Chapter 6) với giao diện tkinter: bảng 3×3 và thông báo kết quả.

Đầu vào – đầu ra:

- Đầu vào: Click vào ô vuông (Button).
- Đầu ra: X hoặc O hiện lên, hiển thị người thắng hoặc hoà.

Tính năng yêu cầu:

- Theo dõi turn, legal_moves, winner.
- Reset game.
- Tắt nút sau khi click.

Kiểm tra & kết quả mẫu:

- Dàn xếp thắng hàng ngang → Hiển thị “X thắng!”

GIÁO VIÊN HƯỚNG DẪN

(Ký và ghi rõ họ tên)

NHẬN XÉT CỦA GIÁO VIÊN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Thái Nguyên, ngày….tháng…..năm 2025

GIÁO VIÊN HƯỚNG DẪN

(Ký ghi rõ họ tên

MỤC LỤC

DANH MỤC HÌNH ẢNH

LỜI NÓI ĐẦU

Trong thời đại công nghệ phát triển mạnh mẽ như hiện nay, việc kết hợp giữa lập trình logic và giao diện người dùng (GUI) đóng vai trò quan trọng trong việc tạo nên những phần mềm thân thiện và trực quan. Đề tài "Triển khai game Tic-Tac-Toe với giao diện Tkinter" nhằm mục tiêu giúp sinh viên vận dụng kiến thức lập trình Python, đặc biệt là thư viện Tkinter, để xây dựng một trò chơi đơn giản nhưng đầy tính tương tác.

Tic-Tac-Toe (hay còn gọi là trò chơi caro 3x3) là một trò chơi quen thuộc, đòi hỏi tư duy chiến thuật và khả năng phản xạ nhanh. Trong đề tài này, trò chơi sẽ được xây dựng trên bảng 3x3 bằng các nút bấm (Button), với chức năng hiển thị lần lượt ký hiệu "X" hoặc "O" tương ứng với lượt chơi của từng người. Ứng dụng sẽ theo dõi lượt đi (turn), danh sách các ô hợp lệ (legal_moves), xác định người chiến thắng (winner), đồng thời cung cấp chức năng đặt lại trò chơi (reset) và vô hiệu hóa nút sau khi đã được chọn.

Qua việc hoàn thiện đề tài, người học không chỉ củng cố kiến thức lập trình hướng đối tượng và xử lý sự kiện trong Python, mà còn phát triển tư duy tổ chức giao diện và quản lý trạng thái trong ứng dụng. Đây là bước đệm quan trọng trong việc xây dựng các chương trình GUI phức tạp hơn trong tương lai.

CHƯƠNG I: GIỚI THIỆU

1. Giới thiệu về Python

Python là ngôn ngữ lập trình bậc cao đa năng. Triết lý thiết kế của nó nhấn mạnh khả năng đọc mã bằng cách sử dụng thụt lề đáng kể.[32]

Python có kiểu động và thu gom rác. Ngôn ngữ này hỗ trợ nhiều mô hình lập trình, bao gồm lập trình cấu trúc (đặc biệt là lập trình thủ tục), lập trình hướng đối tượng và lập trình chức năng. Nó thường được mô tả là ngôn ngữ "bao gồm pin" do có thư viện tiêu chuẩn toàn diện.

Hình 1. Ngôn ngữ Python

1.2. Ứng dụng của Python

1. Phát triển Web (Web Development)

Python được sử dụng rộng rãi để xây dựng các trang web và ứng dụng web thông qua các framework nổi tiếng như:

- Django: Framework mạnh mẽ, bảo mật cao, giúp xây dựng web nhanh chóng với cấu trúc MVC rõ ràng.

- Flask: Framework nhẹ, linh hoạt, phù hợp với các dự án nhỏ hoặc các API đơn giản.
- FastAPI: Framework hiện đại dùng để xây dựng API hiệu năng cao.

2. Khoa học dữ liệu & Phân tích dữ liệu (Data Science & Data Analysis)

Python là ngôn ngữ hàng đầu trong lĩnh vực dữ liệu nhờ vào các thư viện mạnh mẽ:

- NumPy: Xử lý mảng số học đa chiều.
- Pandas: Phân tích và thao tác dữ liệu dạng bảng (DataFrame).
- Matplotlib, Seaborn: Vẽ biểu đồ và trực quan hóa dữ liệu.
- Jupyter Notebook: Môi trường tương tác để viết và hiển thị kết quả phân tích.

3. Trí tuệ nhân tạo và Máy học (AI & Machine Learning)

Python là nền tảng chính trong lĩnh vực AI và học máy (ML) nhờ các thư viện:

- Scikit-learn: Xây dựng các mô hình học máy như hồi quy, phân loại, clustering.
- TensorFlow, PyTorch, Keras: Dùng trong Deep Learning và xử lý mạng nơ-ron.
- OpenCV: Xử lý ảnh và thị giác máy tính.

4. Tự động hóa và DevOps (Automation & Scripting)

Python rất mạnh trong việc tự động hóa các tác vụ lặp đi lặp lại:

- Viết script để xử lý file, gửi email, đọc file Excel.
- Tự động hoá kiểm thử phần mềm (test automation).
- Triển khai hệ thống với Ansible, Fabric, hoặc các công cụ DevOps khác.

5. Phát triển ứng dụng máy tính để bàn (Desktop Applications)

Python hỗ trợ xây dựng phần mềm với giao diện đồ họa (GUI):

- Tkinter: Thư viện GUI có sẵn, đơn giản, dễ sử dụng.
- PyQt / PySide: Xây dựng GUI chuyên nghiệp, phức tạp.

- Kivy: Hỗ trợ xây dựng ứng dụng chạy đa nền tảng (Windows, Android, iOS).

6. Lập trình nhúng & IoT (Embedded Programming & Internet of Things)

Python có thể chạy trên các thiết bị nhỏ như Raspberry Pi để điều khiển phần cứng:

- Đọc cảm biến, điều khiển LED, động cơ.
- Tạo hệ thống giám sát, báo động, nhà thông minh.
- Tạo máy chủ web mini chạy trực tiếp trên Raspberry Pi.

7. Lập trình Game (Game Development)

Python hỗ trợ phát triển game thông qua:

- Pygame: Thư viện đơn giản dùng để xây dựng trò chơi 2D.
- Godot (với GDScript gần giống Python): Dùng cho game chuyên nghiệp hơn.

8. Lĩnh vực giáo dục (Education)

Với cú pháp đơn giản, Python là ngôn ngữ lý tưởng cho người mới học lập trình:

- Được sử dụng rộng rãi trong các trường đại học và chương trình STEM.
- Nhiều công cụ học tập như Thonny, IDLE, hoặc nền tảng học online hỗ trợ Python.

1.3. Ưu điểm của Python

Python nổi bật nhờ sự đơn giản, linh hoạt và mạnh mẽ. Dưới đây là những ưu điểm chính:

❖ Cú pháp đơn giản, dễ học

- Python có cú pháp gần giống ngôn ngữ tự nhiên, dễ hiểu và dễ viết.
- Rất phù hợp cho người mới bắt đầu học lập trình.

❖ Mã nguồn mở, miễn phí

- Python là phần mềm mã nguồn mở (open source), có thể sử dụng miễn phí cho mọi mục đích, kể cả thương mại.

❖ Đa nền tảng, dễ triển khai

- Python chạy được trên nhiều hệ điều hành như Windows, Linux, macOS mà không cần thay đổi mã nguồn.

❖ Thư viện phong phú và cộng đồng mạnh

- Có hàng ngàn thư viện hỗ trợ cho các lĩnh vực như: web, dữ liệu, AI, game, mạng...
- Cộng đồng Python toàn cầu rất lớn, luôn sẵn sàng chia sẻ và hỗ trợ.

❖ Đa mục đích (general-purpose)

- Dùng được trong nhiều lĩnh vực: Web, Game, Phân tích dữ liệu, Trí tuệ nhân tạo, Tự động hóa, IoT, v.

❖ Lập trình nhanh và phát triển nhanh

- Vì viết ít dòng code hơn so với các ngôn ngữ khác, Python giúp tăng tốc độ phát triển ứng dụng.

1.4. Nhược điểm của Python

Dù có nhiều ưu điểm, Python vẫn có những hạn chế đáng lưu ý:

❖ Tốc độ thực thi chậm

- Là ngôn ngữ thông dịch, Python chậm hơn so với các ngôn ngữ biên dịch như C/C++ hoặc Java, đặc biệt trong các tác vụ tính toán nặng.

❖ Không phù hợp với ứng dụng thời gian thực hoặc hiệu suất cao

- Những ứng dụng yêu cầu tốc độ xử lý cực nhanh như game 3D, trình điều khiển hệ thống, hoặc các phần mềm nhúng công nghiệp ít khi dùng Python.

❖ Quản lý bộ nhớ chưa tối ưu

- Python tiêu tốn nhiều tài nguyên bộ nhớ, do đó không phù hợp với các thiết bị có cấu hình thấp như vi điều khiển hoặc hệ thống nhúng hạn chế RAM.

❖ Giao diện đồ họa (GUI) chưa mạnh mẽ

- Các thư viện GUI như Tkinter hay PyQt có thể tạo giao diện, nhưng không đủ chuyên sâu hoặc hiện đại như các công cụ như Electron, .NET, hoặc JavaFX.

❖ Khó khăn khi chuyển sang thiết bị di động

- Việc viết ứng dụng Android/iOS bằng Python gặp giới hạn về hiệu năng và tài nguyên hỗ trợ, không mạnh mẽ bằng Java (Android) hoặc Swift (iOS).

2. Mô tả đề tài

2.1. Giới thiệu đề tài

Game Tic-Tac-Toe, hay còn gọi là cờ caro 3x3, là một trò chơi hai người với luật chơi đơn giản: hai người chơi lần lượt đặt ký hiệu "X" hoặc "O" trên một bảng 3x3, người chơi nào xếp được ba ký hiệu giống nhau theo hàng ngang, hàng dọc hoặc đường chéo trước sẽ giành chiến thắng. Nếu tất cả các ô đều được điền mà không có người thắng, trò chơi kết thúc với kết quả hòa. Với tính chất đơn giản nhưng giàu tính chiến thuật, Tic-Tac-Toe là một bài toán phổ biến trong lập trình, đặc biệt trong việc học cách kết hợp logic trò chơi và giao diện người dùng.

Đề tài này tập trung vào việc thiết kế và phát triển game Tic-Tac-Toe sử dụng thư viện tkinter của Python, một công cụ mạnh mẽ để xây dựng giao diện đồ họa. Ứng dụng bao gồm một bảng chơi 3x3 với các nút tương tác, cho phép người chơi thực hiện các nước đi, hiển thị kết quả và hỗ trợ reset game.

- **Đầu vào** của chương trình là các thao tác nhấp chuột vào các ô trên bảng để đặt ký hiệu "X" hoặc "O", cùng với nút reset để bắt đầu lại ván chơi.
- **Đầu ra** bao gồm việc hiển thị ký hiệu "X" hoặc "O" trên các ô được chọn, cập nhật trạng thái lượt chơi trên nhãn giao diện, và thông báo kết quả (thắng hoặc hòa) thông qua hộp thoại và nhãn trạng thái.

2.2. Tính năng của chương trình

1. Giao diện đồ họa (GUI) trực quan

- Sử dụng Tkinter để thiết kế bảng chơi gồm 9 ô (3×3) được hiển thị bằng các nút (Button).

- Người chơi tương tác trực tiếp bằng cách nhấn chuột vào các ô để thực hiện nước đi.

2. Theo dõi lượt chơi (turn tracking)
   - Trò chơi hỗ trợ 2 người chơi, lần lượt đánh dấu X và O.
   - Chương trình tự động chuyển lượt sau mỗi lần click và hiển thị đúng ký hiệu của người chơi tương ứng.
3. Tắt nút sau khi chọn (disable button after click)
   - Sau khi một ô đã được chọn, nút đó sẽ bị vô hiệu hóa (disabled) để tránh việc người chơi click lại cùng một ô.
4. Kiểm tra và phát hiện người thắng cuộc (winner checking)
   - Chương trình tự động kiểm tra sau mỗi lượt đi xem có ai chiến thắng hay chưa.
   - Các điều kiện thắng được kiểm tra:
     - 3 ký hiệu giống nhau theo hàng ngang.
     - 3 ký hiệu giống nhau theo cột dọc.
     - 3 ký hiệu giống nhau theo đường chéo.
   - Nếu có người thắng, chương trình sẽ hiển thị thông báo: "X thắng!" hoặc "O thắng!".
5. Kiểm tra hòa trận (draw checking)
   - Nếu sau 9 lượt chơi không có người thắng, chương trình tự động xác định kết quả hòa và hiển thị thông báo "Hòa!".
6. Chức năng chơi lại (Reset Game)
   - Sau khi ván chơi kết thúc (thắng hoặc hòa), người dùng có thể nhấn nút "Chơi lại" để bắt đầu ván mới.
   - Tất cả các nút sẽ được bật lại và làm mới nội dung.
7. Xử lý hợp lệ nước đi (legal moves only)
   - Chương trình chỉ cho phép người chơi đi vào những ô còn trống.
   - Tránh được tình trạng đè chồng ký hiệu hoặc chơi sai luật.

2.3. Thách thức

1. Thiết kế giao diện đồ họa (Tkinter GUI)
    - Tkinter là thư viện GUI đơn giản nhưng không hỗ trợ kéo thả giao diện như một số công cụ hiện đại.
    - Việc bố trí lưới nút 3x3 và xử lý sự kiện khi nhấn vào từng ô yêu cầu phải nắm vững lập trình hướng sự kiện.
    - Cần quản lý đúng vị trí và trạng thái của từng nút (Button) để đảm bảo tương tác chính xác.
2. Quản lý lượt chơi (Turn tracking)
    - Việc luân phiên lượt chơi giữa hai người (X và O) tưởng đơn giản nhưng phải xử lý logic chặt chẽ, tránh việc người chơi đánh liên tiếp 2 lượt hoặc thay đổi sai biểu tượng.
3. Kiểm tra điều kiện thắng – thua
    - Một trong những phần khó nhất là xây dựng thuật toán kiểm tra thắng cuộc, cần kiểm tra tất cả các hàng, cột và đường chéo sau mỗi lượt đi.
    - Phải đảm bảo thuật toán chính xác, không bỏ sót trường hợp hoặc kiểm tra sai.
4. Xử lý trường hợp hòa (draw game)
    - Không chỉ kiểm tra thắng, chương trình cần phát hiện khi toàn bộ ô đã được đi nhưng không có ai thắng.
    - Việc xác định chính xác thời điểm "hòa" và không nhầm với trạng thái đang chơi là một điểm cần xử lý cẩn thận.
5. Chức năng reset (chơi lại)
    - Reset game không đơn giản là xóa các ô, mà còn phải đặt lại toàn bộ trạng thái chương trình: lượt chơi, trạng thái các nút, biến theo dõi kết quả...
    - Việc không reset đúng cách có thể gây lỗi logic khi chơi lại.
6. Quản lý trạng thái nút
    - Sau khi người chơi click vào một ô, phải vô hiệu hóa (disable) nút đó để ngăn người chơi chọn lại.

- Cần đảm bảo các nút được kích hoạt hoặc khóa đúng thời điểm trong chu trình trò chơi.

7. Thiếu kinh nghiệm với Tkinter

- Với những người mới bắt đầu lập trình Python, việc tiếp cận Tkinter có thể gây khó khăn ban đầu:
    - Khó hiểu khi tổ chức mã theo cấu trúc hướng đối tượng.
    - Khó khăn trong việc xử lý sự kiện (command, lambda).

2.4. Các kiến thức sử dụng

Để xây dựng thành công trò chơi Tic-Tac-Toe (Caro 3x3) sử dụng ngôn ngữ lập trình Python kết hợp với thư viện Tkinter, người thực hiện cần nắm vững một số kiến thức cơ bản và chuyên sâu trong lập trình, xử lý logic trò chơi và thiết kế giao diện người dùng. Cụ thể như sau:

1. Kiến thức về ngôn ngữ lập trình Python

- Biến, kiểu dữ liệu và phép toán: Hiểu và sử dụng các kiểu dữ liệu cơ bản như int, str, bool, list, tuple, None, cùng các phép toán toán học và logic.
- Câu lệnh điều kiện: if, elif, else để xử lý điều kiện thắng – thua, hòa.
- Vòng lặp: for, while dùng để xử lý các ô trong bảng hoặc reset trò chơi.
- Hàm (function): Viết các hàm để xử lý logic riêng biệt như kiểm tra thắng, xử lý click, reset game...
- Biến toàn cục và cục bộ: Hiểu cách sử dụng và truyền dữ liệu giữa các hàm hoặc trong phạm vi chương trình.

2. Kiến thức về xử lý logic trò chơi (Game Logic)

- Luân phiên lượt chơi (Turn-based logic): Quản lý người chơi hiện tại là X hay O, đổi lượt sau mỗi lần đánh.
- Kiểm tra điều kiện thắng cuộc: Biết cách xác định người chơi thắng thông qua các tổ hợp hàng ngang, dọc, chéo.
- Xử lý trạng thái hòa: Xác định khi nào bảng đầy mà không có ai thắng.

- Quản lý trạng thái bảng: Lưu trữ trạng thái của từng ô trong bảng (đã đánh hay chưa, là X hay O...).

3. Kiến thức về lập trình GUI với Tkinter
   - Khởi tạo cửa sổ chính (Tk): Tạo giao diện cửa sổ chính của trò chơi.
   - Tạo và bố trí các widget: Sử dụng Button, Label, Frame, và sắp xếp bằng grid() để tạo bảng 3x3.
   - Xử lý sự kiện click chuột: Gắn sự kiện cho mỗi nút bằng tham số command, sử dụng lambda nếu cần truyền tham số.
   - Cập nhật nội dung nút và giao diện: Sử dụng .config() để thay đổi nội dung, màu sắc, trạng thái nút sau khi click.
   - Tắt/bật nút: Dùng .config(state="disabled") hoặc "normal" để điều khiển khả năng tương tác.
   - Hiển thị kết quả: Sử dụng Label hoặc messagebox.showinfo() để thông báo người thắng hoặc hòa.
4. Tư duy lập trình hướng mô-đun (Modular Programming)
   - Tách chương trình thành các phần hợp lý: Viết từng chức năng như kiểm tra thắng, reset, xử lý click thành hàm riêng để dễ quản lý, sửa lỗi và nâng cấp.
   - Tái sử dụng mã: Giảm lặp lại bằng cách viết các hàm dùng chung, như kiểm tra điều kiện thắng hoặc cập nhật giao diện.
5. Kỹ năng kiểm thử và gỡ lỗi
   - Test từng chức năng nhỏ: Kiểm tra logic thắng, reset, xử lý sự kiện từng bước.
   - Xử lý lỗi phát sinh khi chạy chương trình: Sửa lỗi logic, lỗi liên quan đến giao diện, hoặc thứ tự thực thi mã.

CHƯƠNG II. CƠ SỞ LÝ THUYẾT

2.1. Ngôn ngữ lập trình sử dụng

- Chương trình được xây dựng bằng Python, một ngôn ngữ bậc cao, dễ học, cú pháp gọn gàng và hỗ trợ tốt cho lập trình hướng đối tượng cũng như lập trình giao diện người dùng (GUI). Python rất phù hợp với mô hình mô phỏng và tương tác

2.2. Các kiến thức sử dụng trong chương trình

2.2.1. Lập trình hướng đối tượng (OOP):

- **Cấu trúc lớp**: Chương trình sử dụng hai lớp chính là TTTBoard và TicTacToeGUI. Lớp TTTBoard chịu trách nhiệm quản lý logic trò chơi (bảng chơi, lượt chơi, kiểm tra thắng/hòa), trong khi lớp TicTacToeGUI xử lý giao diện đồ họa và tương tác người dùng. Sự tách biệt này tuân thủ nguyên tắc trách nhiệm đơn nhất (Single Responsibility Principle).
- Thuộc tính và phương thức: Lớp TTTBoard sử dụng các thuộc tính như self.board (danh sách 1 chiều lưu trạng thái bảng), self.current_turn (lượt chơi hiện tại: "X" hoặc "O"), và self.winner (kết quả trò chơi). Các phương thức như make_move(), check_winner(), và reset() được định nghĩa để xử lý các hành vi cụ thể, đảm bảo tính đóng gói và tái sử dụng.
- Tính mô-đun: Việc tách logic và giao diện cho phép dễ dàng bảo trì, mở rộng (ví dụ, thêm chế độ chơi với máy) hoặc tái sử dụng mã nguồn trong các dự án khác. Các phương thức được thiết kế độc lập, giảm sự phụ thuộc lẫn nhau.

2.2.2. Thư viện Tkinter – Tạo giao diện người dùng (GUI)

a) Giới thiệu về Tkinter

Tkinter là thư viện tiêu chuẩn (standard GUI library) của Python, được tích hợp sẵn mà không cần cài đặt thêm. Nó là giao diện Python cho bộ công cụ

GUI Tk (Tcl/Tk) — một bộ công cụ phát triển giao diện người dùng rất phổ biến và nhẹ.

- Được tích hợp mặc định trong mọi phiên bản Python.
- Là công cụ đơn giản và hiệu quả để xây dựng ứng dụng giao diện đồ họa (GUI).
- Hoạt động trên đa nền tảng: Windows, macOS và Linux.

b) Vai trò chủa Tkinter trong đề tài

Trong đề tài triển khai trò chơi Tic-Tac-Toe, thư viện Tkinter được sử dụng để:

- Xây dựng giao diện 3x3 của bàn cờ bằng các nút (Button).
- Hiển thị lượt chơi, thông báo kết quả bằng nhãn (Label) và hộp thoại (messagebox).
- Gắn các sự kiện người dùng như nhấn nút, reset game.
- Tương tác thời gian thực: Khi người chơi click vào ô, nội dung sẽ được cập nhật ngay lập tức trên màn hình.

c) Các Widget của Tkinter sử dụng trong chương trình

Thành phần	**Mục đích sử dụng**
Tk()	Tạo cửa sổ chính cho ứng dụng
Button()	Tạo các ô vuông trên bàn cờ (9 nút tương ứng 9 ô)
Label()	Hiển thị thông tin lượt chơi hoặc kết quả thắng – hòa
messagebox.showinfo()	Hiển thị hộp thoại thông báo người thắng hoặc hòa
.grid(row, column)	Bố trí các widget (nút, nhãn) theo dạng bảng

Bảng 2. Widget sử dụng trong chương trình

2.2.3. Kiểu dữ liệu List

- Kiểu dữ liệu List trong Python là một collection lưu trữ các phần tử theo tứ tự đã cho, có thể thay đổi. Cho phép chứa dữ liệu trùng lặp. List có cấu trúc dữ liệu mà có khả năng lưu giữ các kiểu dữ liệu khác nhau.
- List trong Python là cấu trúc dữ liệu mà có khả năng lưu giữ các kiểu dữ liệu khác nhau.
- List trong Python là thay đổi (mutable), nghĩa là Python sẽ không tạo một List mới nếu bạn sửa đổi một phần tử trong List.
- List là một container mà giữ các đối tượng khác nhau trong một thứ tự đã cho. Các hoạt động khác nhau như chèn hoặc xóa có thể được thực hiện trên List.
- List trong Python được viết với dấu ngoặc vuông [ ].

CHƯƠNG III. THIẾT KẾ VÀ XÂY DỰNG CHƯƠNG TRÌNH

3.1. Sơ đồ khối hệ thống

3.1.1. Mô tả các module chính trong chương trình

Chương trình được chia thành hai module chính: TTTBoard (logic trò chơi) và TicTacToeGUI (giao diện người dùng), cùng với một khối chính để khởi động. Thiết kế mô-đun này đảm bảo mã nguồn rõ ràng, dễ hiểu, và dễ mở rộng, thể hiện sự áp dụng hiệu quả của lập trình hướng đối tượng và thư viện tkinter.

a) Module Logic Trò Chơi (Lớp TTTBoard):

- ❖ Chức năng: Quản lý toàn bộ logic của trò chơi Tic-Tac-Toe, bao gồm trạng thái bảng chơi, lượt chơi, kiểm tra nước đi hợp lệ, xác định người thắng hoặc hòa, và reset trò chơi.
- ❖ Các thành phần chính:
  - Thuộc tính:
    - self.board: Danh sách 1 chiều ([''] * 9) lưu trạng thái 9 ô trên bảng 3x3, với giá trị là '' (trống), "X", hoặc "O".
    - self.current_turn: Chuỗi ký tự ("X" hoặc "O") theo dõi lượt chơi hiện tại.
    - self.winner: Biến lưu kết quả trò chơi (None, "X", "O", hoặc "Hòa").
  - Phương thức:
    - __init__(): Khởi tạo bảng chơi rỗng, đặt lượt chơi ban đầu là "X" và winner là None.
    - make_move(index): Xử lý nước đi tại ô index, kiểm tra tính hợp lệ (ô trống và chưa có người thắng), cập nhật bảng chơi, kiểm tra thắng/hòa, và chuyển lượt.
    - check_winner(): Kiểm tra điều kiện thắng bằng cách xét các hàng, cột, và đường chéo. Trả về True nếu người chơi hiện tại thắng.

- reset(): Đặt lại bảng chơi, lượt chơi, và kết quả về trạng thái ban đầu.

❖ Vai trò: Module này là lõi logic của trò chơi, đảm bảo các quy tắc được thực thi chính xác, độc lập với giao diện. Nó cung cấp các phương thức để lớp giao diện gọi và lấy thông tin trạng thái.

❖ Tương tác: Cung cấp dữ liệu và trạng thái cho lớp TicTacToeGUI thông qua các phương thức như make_move() và thuộc tính như board, current_turn, winner.

b) Module Giao Diện Người Dùng (Lớp TicTacToeGUI):

❖ Chức năng: Xây dựng và quản lý giao diện đồ họa, xử lý tương tác người dùng (nhấp chuột, hiển thị thông báo), và đồng bộ trạng thái với module logic.

❖ Các thành phần chính:

- Thuộc tính:
  - self.root: Đối tượng tk.Tk, đại diện cho cửa sổ chính của ứng dụng.
  - self.board: Tham chiếu đến đối tượng của lớp TTTBoard, cho phép truy cập logic trò chơi.
  - self.buttons: Danh sách chứa 9 đối tượng tk.Button, đại diện cho các ô trên bảng chơi.
  - self.label: Đối tượng tk.Label, hiển thị trạng thái lượt chơi hoặc kết quả.
  - self.reset_button: Đối tượng tk.Button, dùng để reset trò chơi.
- Phương thức:
  - __init__(root): Khởi tạo giao diện với cửa sổ chính, tạo 9 nút ô játszói, nhãn trạng thái, và nút reset. Gắn sự kiện nhấp chuột cho các nút.

- on_click(index): Xử lý sự kiện nhấp chuột, gọi make_move(index) từ TTTBoard, cập nhật giao diện (ký hiệu trên nút, trạng thái nhãn), và hiển thị thông báo kết quả.
- disable_all_buttons(): Vô hiệu hóa tất cả các nút khi trò chơi kết thúc.
- reset_game(): Gọi reset() từ TTTBoard và cập nhật giao diện (xóa ký hiệu, kích hoạt nút, đặt lại nhãn).

❖ Vai trò: Module này chịu trách nhiệm tạo giao diện trực quan, xử lý tương tác người dùng, và hiển thị trạng thái trò chơi, đồng thời đồng bộ với module logic để phản ánh các thay đổi.

❖ Tương tác: Gọi các phương thức của TTTBoard (như make_move(), reset()) để cập nhật trạng thái trò chơi và sử dụng kết quả để điều chỉnh giao diện (như cập nhật self.buttons, self.label).

c) Module Tương Tác Chính (Main Block):

❖ Chức năng: Kết nối các module và khởi động ứng dụng.

❖ Code minh họa:

```
if __name__ == '__main__':
    root = tk.Tk()
    game = TicTacToeGUI(root)
    root.mainloop()
```

Hình 3 : Code kết nối module

- ❖ Vai trò: Khởi tạo cửa sổ chính (tk.Tk), tạo đối tượng TicTacToeGUI, và chạy vòng lặp chính của tkinter (mainloop) để xử lý các sự kiện giao diện.
- ❖ Tương tác: Là điểm nhập chính của chương trình, kết nối module giao diện (TicTacToeGUI) với hệ thống tkinter và đảm bảo ứng dụng chạy liên tục.

d) Tích hợp các module:

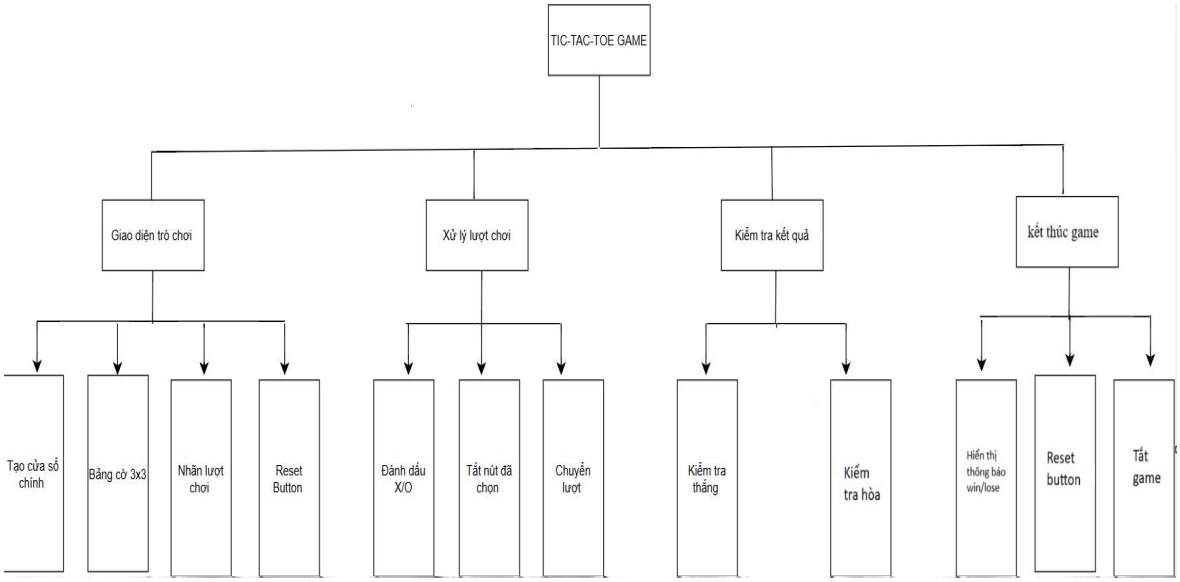
- ❖ Luồng hoạt động:
  - Khi người dùng nhấp vào một ô, TicTacToeGUI.on_click(index) được gọi, gửi yêu cầu nước đi đến TTTBoard.make_move(index).
  - TTTBoard kiểm tra nước đi, cập nhật self.board, kiểm tra thắng/hòa, và chuyển lượt. Kết quả được trả về để TicTacToeGUI cập nhật giao diện (ký hiệu trên nút, nhãn trạng thái, hoặc thông báo).
  - Khi nhấp nút reset, TicTacToeGUI.reset_game() gọi TTTBoard.reset() để đặt lại logic, sau đó cập nhật giao diện.
- ❖ Đồng bộ hóa: Module TicTacToeGUI sử dụng các thuộc tính (self.board.board, self.board.current_turn, self.board.winner) từ TTTBoard để đồng bộ trạng thái giao diện với logic trò chơi.

e) Lợi ích của thiết kế mô-đun:

- Tách biệt trách nhiệm: Logic và giao diện được tách biệt, giúp dễ dàng bảo trì và mở rộng (ví dụ, thêm tính năng chơi với máy hoặc lưu điểm số).
- Tái sử dụng: Module TTTBoard có thể được tái sử dụng trong các ứng dụng khác (như chơi trên giao diện dòng lệnh) mà không cần sửa đổi.
- Dễ kiểm thử: Các phương thức trong TTTBoard có thể được kiểm thử độc lập với giao diện.

3.1.2. Biểu đồ phân cấp chức năng

Dưới đây là biểu đồ phân cấp chức năng (Functional Hierarchy Diagram) cho đề tài "Triển khai game Tic-Tac-Toe với giao diện tkinter". Biểu đồ này được chia thành các chức năng chính và các chức năng con theo dạng phân cấp:



Hình 4. Biểu đồ phân cấp chức năng

Giải thích:

1. Giao diện trò chơi

Đây là phần xây dựng giao diện người dùng bằng thư viện tkinter.

- Tạo cửa sổ chính: Tạo cửa sổ ứng dụng (tk.Tk()) để chứa tất cả thành phần.
- Bảng cờ 3x3: Tạo 9 nút (Button), sắp xếp thành 3 hàng 3 cột. Mỗi ô tương ứng với một vị trí trên bảng.
- Nhãn lượt chơi: Hiển thị người chơi hiện tại (X hoặc O), cập nhật sau mỗi lượt.
- Nút Reset: Cho phép người dùng chơi lại từ đầu bằng cách đặt lại bảng cờ.

2. Xử lý lượt chơi

Phần xử lý logic khi người dùng nhấn vào ô cờ.

- Đánh dấu X/O: Ghi dấu người chơi hiện tại vào ô đã chọn.
- Tắt nút đã chọn: Tránh người chơi chọn lại ô đã dùng bằng cách state="disabled".
- Chuyển lượt: Sau khi người chơi đi, chuyển lượt sang người còn lại.

3. Kiểm tra kết quả

Sau mỗi nước đi, cần kiểm tra xem trò chơi kết thúc chưa.

- Kiểm tra thắng: Kiểm tra hàng, cột và đường chéo để xem có 3 dấu giống nhau không.
- Kiểm tra hòa: Nếu không còn ô trống mà không có người thắng → ván hòa.

4. Kết thúc trò chơi

- Hiển thị kết quả: Dùng messagebox.showinfo() để thông báo thắng hoặc hòa.
- Bật lại nút: Cho phép tương tác lại với các ô (nút được bật lại sau khi đã bị disable).

3.2. Sơ đồ khối các thuật toán chính

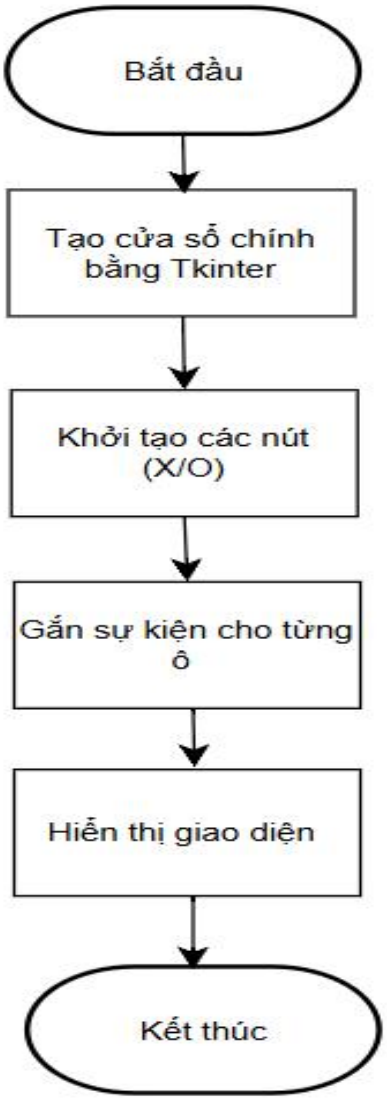
3.2.1. Khởi tạo giao diện

Mục đích: Thiết lập giao diện người dùng với các nút tương tác.

Giải thích các bước:

- Bắt đầu: Khởi động chương trình.
- Tạo cửa sổ chính bằng Tkinter: Sử dụng Tk() để khởi tạo cửa sổ chính của trò chơi.
- Khởi tạo các nút X/O: Tạo các nút tương ứng với 9 ô trên bảng chơi (3x3).
- Gán sự kiện cho từng ô: Mỗi nút (ô) được gắn một sự kiện khi người dùng nhấn (onclick).

- Hiển thị giao diện: Dùng mainloop() để chạy chương trình và hiển thị giao diện.
- Kết thúc: Kết thúc khởi tạo và chương trình sẵn sàng đợi tương tác.



Hình 5. Sơ đồ khối tạo giao diện

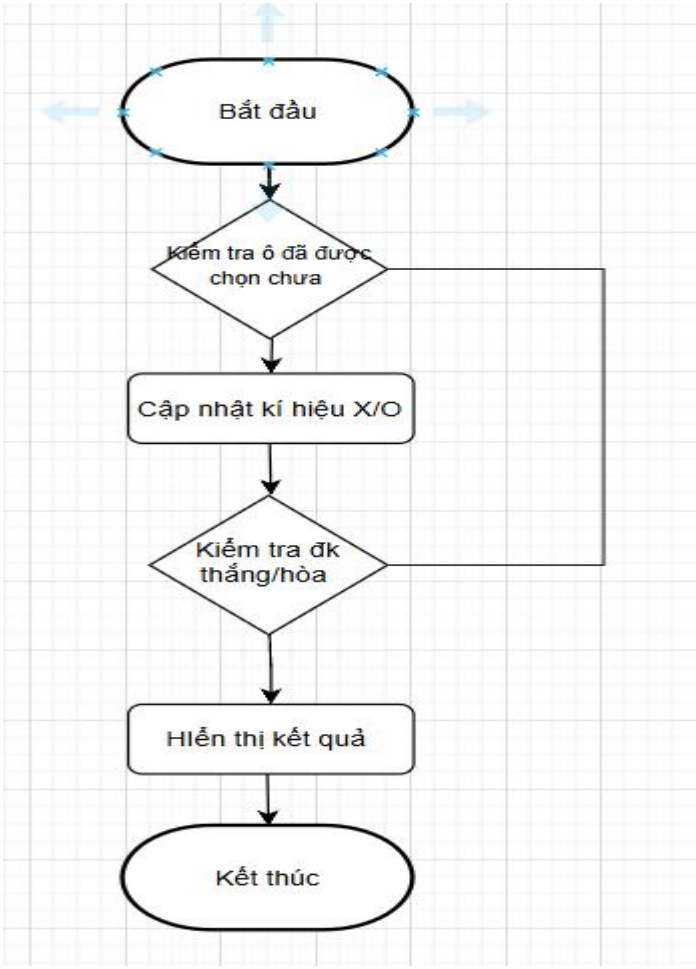
3.2.2. Xử lý lượt chơi

Mục đích: Xử lý mỗi lượt người chơi nhấn vào ô.

Giải thích các bước:

- Bắt đầu: Khi người chơi nhấn vào ô bất kỳ.

- Kiểm tra ô đã được chọn chưa: Nếu ô đã có ký hiệu (X hoặc O), thì không cho đi tiếp.
- Cập nhật ký hiệu X/O: Nếu ô trống, chèn ký hiệu của người chơi hiện tại vào.
- Kiểm tra điều kiện thắng/hòa: Sau khi đánh, kiểm tra xem có ai thắng hoặc trò chơi hòa chưa.
- Hiển thị kết quả: Nếu có người thắng hoặc hòa, hiển thị thông báo.
- Kết thúc: Dừng xử lý lượt hiện tại.



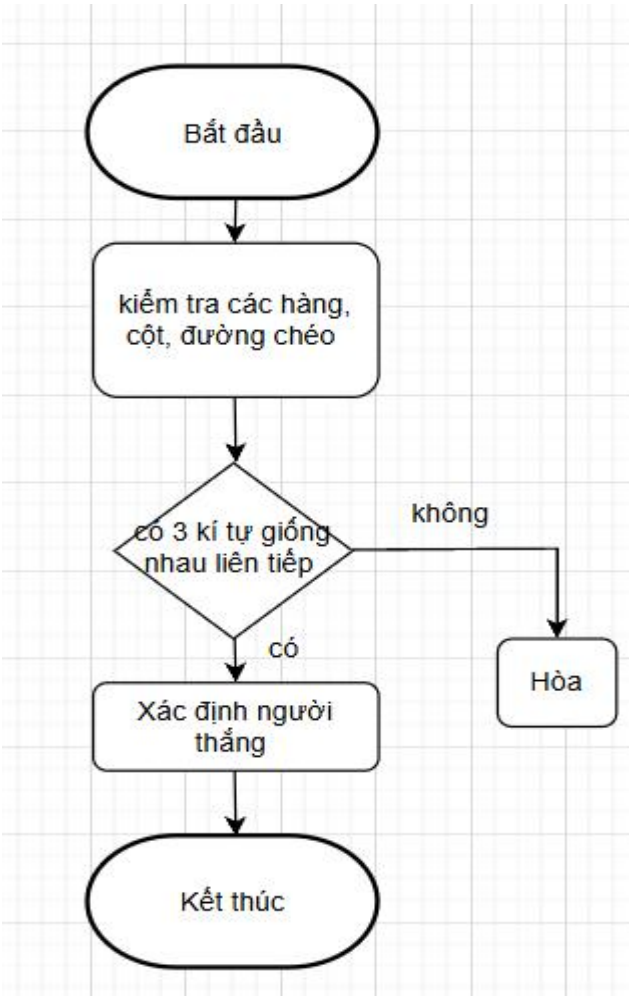
Hình 6. Sơ đồ khối xử lý lượt chơi

3.2.3 Kiểm tra điều kiện thắng

Mục đích: Xác định người thắng hoặc trạng thái hòa.

Giải thích các bước:

- Bắt đầu: Sau mỗi lượt đánh, hệ thống kiểm tra trạng thái bảng.
- Kiểm tra hàng, cột, đường chéo: Lấy dữ liệu từ bảng và kiểm tra theo 3 hàng, 3 cột và 2 đường chéo.
- Có 3 ký tự giống nhau liên tiếp?:
    - Có: Người chơi tương ứng thắng, chuyển sang bước xác định người thắng.
    - Không: Nếu không còn ô trống thì xác định là hòa.
- Xác định người thắng: Cập nhật thông báo và trạng thái trò chơi.
- Kết thúc: Dừng kiểm tra.



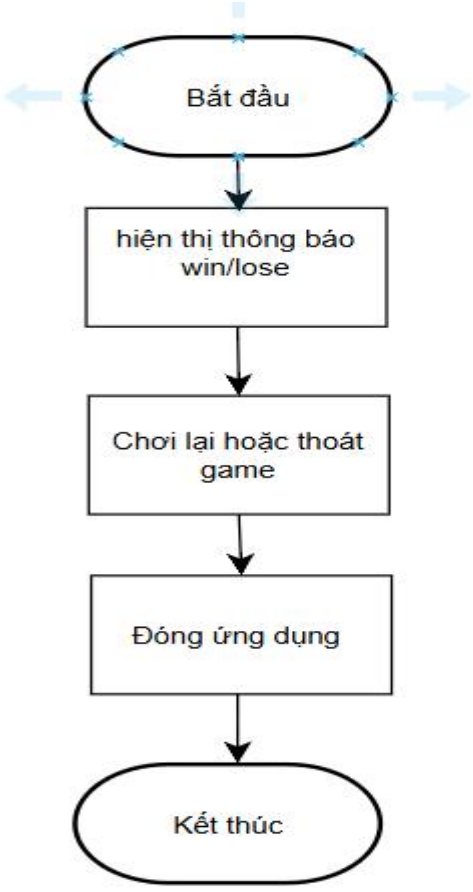
Hình 7. Sơ đồ khối kiểm tra điều kiện thắng

3.2.4. Sau khi trò chơi kết thúc

Mục đích: Xử lý sau khi biết kết quả thắng/thua hoặc hòa.

Giải thích các bước:

- Bắt đầu: Khi có kết quả trò chơi.
- Hiển thị thông báo win/lose: Sử dụng messagebox hoặc label để thông báo kết quả.
- Chơi lại hoặc thoát game: Người chơi chọn "Chơi lại" thì reset trò chơi; chọn "Thoát" thì đóng ứng dụng.
- Đóng ứng dụng: Kết thúc chương trình sau khi hiển thị thông báo thắng/thua hoặc hòa
- Kết thúc: Dừng chương trình.



Hình 8. Sơ đồ khối sau khi kết thúc trò chơi

3.3. Cấu trúc dữ liệu

3.3.1. Kiểu dữ liệu chính được sử dụng

Tên biến / cấu trúc	Kiểu dữ liệu	Mục đích sử dụng
self.board	list[str]	Bàn cờ 3x3
self.current_turn	str	'X' hoặc 'O'
self.winner	str hoặc None	Người thắng: 'X', 'O', 'Hòa' hoặc None
self.buttons	list[tk.Button]	Nút giao diện tương ứng với ô bàn cờ
self.label	tk.Label	Hiển thị thông tin lượt chơi hoặc kết quả
self.reset_button	tk.Button	Nút để đặt lại game

Hình 9. Cấu trúc dữ liệu thường được sử dụng của game

3.4. Chương trình

3.4.1. Các hàm trong chương trình chính

a) Hàm khởi tạo lớp TTTBoard

```
class TTTBoard:
    def __init__(self):
        self.board = [''] * 9
        self.current_turn = 'X'
        self.winner = None
```

Hình 10. Cấu trúc hàm tạo lớp TTTBoard

- Hàm này khởi tạo bảng gồm 9 ô trống, người chơi bắt đầu là 'X', và chưa có người thắng.

b) Hàm xử lý lượt đi

```
def make_move(self, index):
    if self.board[index] == '' and self.winner is None:
        self.board[index] = self.current_turn
        if self.check_winner(index):
            self.winner = self.current_turn
        elif '' not in self.board:
            self.winner = 'Hòa'
        else:
            self.current_turn = 'O' if self.current_turn == 'X' else 'X'
        return True
    return False
```

Hình 11. Cấu trúc hàm xử lý lượt đi

- **Mục đích**: Xử lý nước đi khi người chơi nhấp vào ô index trên bảng.
- **Luồng xử lý**:
  1. Kiểm tra hợp lệ: Nếu ô tại index trống (self.board[index] == '') và chưa có người thắng (self.winner is None), tiếp tục; nếu không, trả về False.
  2. Cập nhật ô: Đặt ký hiệu hiện tại (self.current_turn) vào ô index.
  3. Kiểm tra thắng: Nếu check_winner(index) trả về True, gán self.winner = self.current_turn.
  4. Kiểm tra hòa: Nếu không thắng và bảng đầy ('' not in self.board), gán self.winner = 'Hòa'.
  5. Chuyển lượt: Nếu không thắng/hòa, đổi lượt ('O' nếu là 'X', ngược lại).
  6. Trả về: Trả về True nếu nước đi hợp lệ, False nếu không.
- **Vai trò**: Quản lý nước đi, xác định kết quả (thắng/hòa), và duy trì lượt chơi.

c) Hàm kiểm tra thắng

```
def check_winner(self, index):
    row = index // 3
    col = index % 3

    # Kiểm tra hàng
    if self.board[row * 3] == self.board[row * 3 + 1] == self.board[row * 3 + 2] == self.current_turn:
        return True

    # Kiểm tra cột
    if self.board[col] == self.board[col + 3] == self.board[col + 6] == self.current_turn:
        return True

    # Kiểm tra đường chéo
    if index % 2 == 0:
        if self.board[0] == self.board[4] == self.board[8] == self.current_turn:
            return True
        if self.board[2] == self.board[4] == self.board[6] == self.current_turn:
            return True

    return False
```

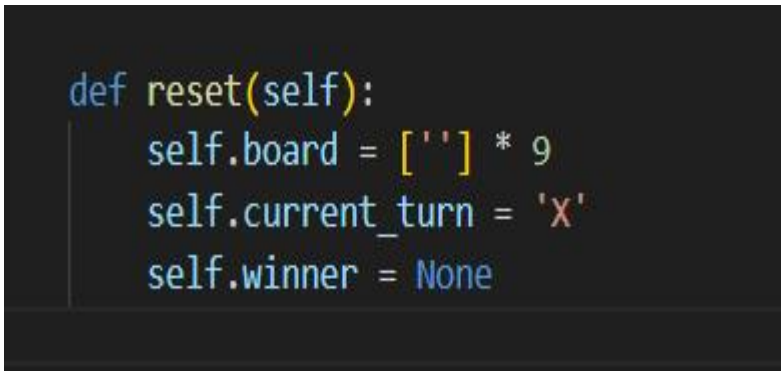
Hình 12. Cấu trúc hàm kiểm tra thắng

- **Mục đích**: Kiểm tra xem người chơi hiện tại có thắng sau nước đi tại ô index không, bằng cách xét hàng, cột, và đường chéo.
- **Luồng xử lý**:
  1. Tính toán vị trí:
     - row = index // 3: Xác định hàng (0, 1, hoặc 2).
     - col = index % 3: Xác định cột (0, 1, hoặc 2).
  2. Kiểm tra hàng: Nếu 3 ô trong hàng row * 3, row * 3 + 1, row * 3 + 2 đều bằng self.current_turn, trả về True.
  3. Kiểm tra cột: Nếu 3 ô trong cột col, col + 3, col + 6 đều bằng self.current_turn, trả về True.
  4. Kiểm tra đường chéo: Nếu index % 2 == 0 (ô ở vị trí chẵn), thì:
     - Kiểm tra đường chéo chính (ô 0, 4, 8) đều bằng self.current_turn, trả về True.
     - Kiểm tra đường chéo phụ (ô 2, 4, 6) đều bằng self.current_turn, trả về True.

5. Kết thúc: Nếu không thỏa mãn điều kiện nào, trả về False.

❖ **Vai trò**: Xác định người thắng dựa trên trạng thái bảng sau mỗi nước đi, với điều kiện chỉ kiểm tra đường chéo khi index ở vị trí chẵn (lỗi tiềm ẩn, nên sửa để kiểm tra cả hai đường chéo mọi lúc).

d) Hàm reset trò chơi



Hình 13. Cấu trúc hàm reset trò chơi

Giải thích hàm reset(self)

- **Mục đích**: Đặt lại trạng thái trò chơi về ban đầu để bắt đầu ván mới.
- **Luồng xử lý**:
  1. Khởi tạo lại bảng: Gán self.board = [''] * 9, tạo danh sách 9 ô trống.
  2. Đặt lại lượt chơi: Gán self.current_turn = 'X', bắt đầu với người chơi "X".
  3. Xóa người thắng: Gán self.winner = None, xóa kết quả thắng/hòa trước đó.
- **Vai trò**: Chuẩn bị trò chơi cho ván mới bằng cách khôi phục trạng thái ban đầu của logic trò chơi.

e) Hàm khởi tạo giao diện

Hình 14. Cấu trúc hàm khởi tạo trò chơi

- **Mục đích**: Khởi tạo giao diện và trạng thái ban đầu của trò chơi trong lớp TicTacToeGUI.
- **Luồng xử lý**:
  1. Gán cửa sổ chính: Lưu đối tượng root (cửa sổ tk.Tk) vào self.root.
  2. Đặt tiêu đề: Gán tiêu đề "Tic Tac Toe OOP" cho cửa sổ.
  3. Khởi tạo logic: Tạo một đối tượng TTTBoard để quản lý logic trò chơi.
  4. Tạo nút chơi: Tạo danh sách self.buttons với 9 nút (tk.Button) cho bảng 3x3, sắp xếp bằng grid, và gắn sự kiện nhấp chuột.
  5. Tạo nhãn trạng thái: Tạo self.label để hiển thị "Lượt: X" ban đầu.
  6. Tạo nút reset: Tạo self.reset_button với nhãn "Reset Game" và gắn phương thức reset_game.
- **Vai trò**: Thiết lập giao diện đồ họa và kết nối với logic trò chơi, chuẩn bị sẵn sàng cho người dùng tương tác.

f) Hàm khi người chơi click

Hình 15. Cấu trúc hàm khi người chơi click vào ô trống

- **Mục đích**: Xử lý sự kiện nhấp chuột vào ô index trên giao diện.
- **Luồng xử lý**:
  1. Thực hiện nước đi: Gọi self.board.make_move(index) để kiểm tra và cập nhật nước đi.
  2. Cập nhật giao diện: Nếu nước đi hợp lệ, đặt ký hiệu (self.board.board[index]) lên nút tại index và vô hiệu hóa nút (state="disabled").
  3. Kiểm tra kết quả: Nếu có self.board.winner:
     - Nếu là "Hòa", cập nhật nhãn thành "Kết quả: Hòa!" và hiển thị thông báo hòa.
     - Nếu là "X" hoặc "O", cập nhật nhãn thành "X thắng!" hoặc "O thắng!" và hiển thị thông báo tương ứng.
     - Vô hiệu hóa tất cả nút.
  4. Cập nhật lượt: Nếu chưa kết thúc, cập nhật nhãn thành "Lượt: [current_turn]".
- **Vai trò**: Kết nối sự kiện người dùng với logic trò chơi, cập nhật giao diện và hiển thị kết quả.

g) Hàm khóa toàn bộ nút

```
def disable_all_buttons(self):
```

Hình 16. Cấu trúc hàm khóa toàn bộ nút

- **Mục đích**: Vô hiệu hóa tất cả các nút trên giao diện sau khi trò chơi kết thúc.
- **Luồng xử lý**:
  1. **Duyệt danh sách nút**: Sử dụng vòng lặp for button in self.buttons để duyệt qua tất cả các nút trong self.buttons.
  2. **Vô hiệu hóa**: Gọi button.config(state="disabled") để khóa từng nút.
- **Vai trò**: Đảm bảo người chơi không thể tiếp tục tương tác khi trò chơi đã kết thúc (thắng hoặc hòa)

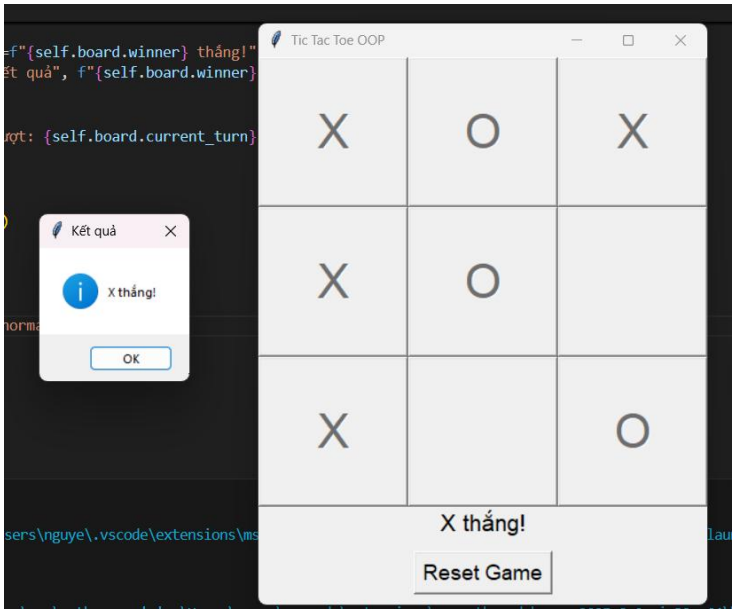
CHƯƠNG IV: THỰC NGHIỆM VÀ KẾT LUẬN

4.1. Thực nghiệm

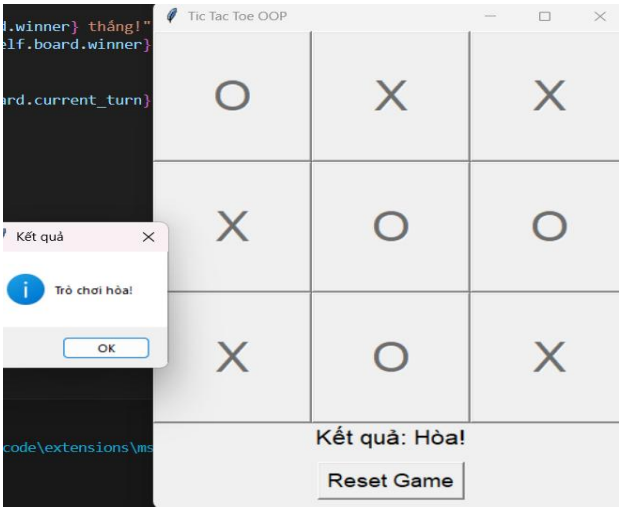
4.1.1. Kiểm thử game

Hình 17. Giao diện game tic-tac-toe

4.1.2. Chức năng của game



Hình 18. Giao diện khi thắng



Hình 19. Giao diện game khi hòa

4.2. Kết luận

4.2.1. Sản phẩm đã làm đã làm được những gì?

❖ **Sản phẩm đã làm được những gì**:

- Hoàn thiện giao diện đồ họa 3x3 trực quan cho phép nhấp chuột đặt "X" hoặc "O".
- Hiển thị kết quả thắng/hòa qua nhãn và hộp thoại ("X thắng!", "O thắng!", "Trò chơi hòa!").
- Hỗ trợ theo dõi lượt chơi giữa "X" và "O".
- Kiểm tra nước đi hợp lệ và xác định người thắng qua hàng, cột, hoặc đường chéo.
- Vô hiệu hóa ô sau khi chọn (state="disabled").
- Cung cấp nút "Reset Game" để bắt đầu lại ván mới.
- Đáp ứng đầy đủ các yêu cầu đề bài.

❖ **Những điều học được**:

- Nắm vững lập trình hướng đối tượng (OOP) qua việc tổ chức mã thành lớp TTTBoard và TicTacToeGUI.

- Hiểu cách sử dụng tkinter để xây dựng giao diện, quản lý bố cục (grid), và xử lý sự kiện.
- Củng cố kỹ năng lập trình logic, quản lý trạng thái (danh sách self.board, biến self.current_turn, self.winner).
- Tối ưu hóa mã nguồn để kiểm tra thắng/hòa hiệu quả.

❖ **Hướng cải tiến trong tương lai**:

- Tích hợp chế độ chơi với máy (AI) bằng thuật toán như Minimax.
- Thêm tính năng đếm số ván thắng để tăng tính cạnh tranh.
- Cải thiện giao diện với hiệu ứng animation và hỗ trợ đa ngôn ngữ.
- Tạo nền tảng cho phát triển các ứng dụng game phức tạp hơn.

TÀI LIỆU THAM KHẢO

1. Cuốn sách ”Python Programming for the Absolute Beginner- 3rd Edition”
2. https://vi.wikipedia.org/wiki/Python_(ng%C3%B4n_ng%E1%BB%AF_l%E1%BA%ADp_tr%C3%ACnh)
3. https://codelearn.io
4. https://www.w3schools.com/

MÃ QR GITHUB: