TRƯỜNG ĐẠI HỌC KỸ THUẬT CÔNG NGHIỆP

KHOA ĐIỆN TỬ - BỘ MÔN CÔNG NGHỆ THÔNG TIN


# BÀI TẬP LỚN

# LẬP TRÌNH PYTHON

**NGÀNH : KỸ THUẬT MÁY TÍNH**

**HỆ : ĐẠI HỌC CHÍNH QUY**

**THÁI NGUYÊN – 2025**

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

**Giảng viên giảng dậy: T.S Nguyễn Văn Huy**

**Sinh viên: Vũ Việt Anh**

**Lớp: K58KTP.K01**

**Link github: https://github.com/vuvietanh01102004/BaiTapPython**

**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

**Sinh viên:** Vũ Việt Anh **MSSV**: K225480106082

**Lớp:** K58KTP.K01

**Ngành:** Kĩ thuật máy tính

**Giảng viên giảng dậy:** T.S Nguyễn Văn Huy

1. Tên đề tài

*Bài 3. Trivia Challenge có GUI & file I/O*

2. Yêu cầu

Đọc file, bắt lỗi file không tồn tại hoặc format sai.

GUI: Label câu hỏi, Entry đáp án, nút "Nộp", Label điểm.

Tính điểm, chuyển câu hỏi kế tiếp.

Nút "Kết thúc" hiển thị tổng điểm.

**GIÁO VIÊN HƯỚNG DẪN**

*(Ký và ghi rõ họ tên)*

## NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

……

……..……

……..……

……..

……..……

Thái Nguyên, ngày…tháng…năm 2025.

**GIÁO VIÊN HƯỚNG DẪN**

*(Ký và ghi rõ họ tên)*

# LỜI CAM ĐOAN

Em xin cam kết bài tập lớn môn Python với đề tài "Bài 3: Trivia Challenge có GUI và thao tác với file I/O" là sản phẩm do chính em thực hiện, dựa trên kiến thức đã học trong môn học và quá trình tự nghiên cứu, tìm hiểu thêm.

Trong suốt quá trình làm bài, em hoàn toàn không sao chép, sử dụng trái phép sản phẩm của người khác hoặc sử dụng công cụ hỗ trợ một cách gian lận. Em đã tuân thủ đầy đủ các quy tắc về đạo đức học tập và tinh thần tự giác trong học tập.

Các tài liệu tham khảo (nếu có) đều được trích dẫn rõ ràng, minh bạch trong phần tài liệu tham khảo. Em hiểu rằng mọi hành vi gian lận đều sẽ bị xử lý nghiêm theo quy định của nhà trường.

Em xin chịu hoàn toàn trách nhiệm về tính trung thực và nội dung của bài làm trước giảng viên phụ trách môn học và nhà trường. Em rất mong nhận được những góp ý từ thầy để em có thể cải thiện và hoàn thiện hơn trong các bài nghiên cứu sau này.

Em xin chân thành cảm ơn!

**Sinh viên thực hiện**

Anh

Vũ Việt Anh

# MỤC LỤC

# LỜI NÓI ĐẦU

Trong quá trình học tập và rèn luyện kỹ năng lập trình với ngôn ngữ Python, việc kết hợp giữa xử lý tệp dữ liệu (file I/O) và xây dựng giao diện đồ họa người dùng (GUI) đóng vai trò quan trọng trong việc phát triển các ứng dụng có tính tương tác cao. Đề tài "Trivia Challenge có GUI & file I/O" là một ví dụ điển hình, giúp người học ứng dụng đồng thời các kiến thức về xử lý dữ liệu, tổ chức chương trình, và thiết kế giao diện người dùng.

Cụ thể, đề tài yêu cầu chuyển đổi chương trình "Trivia Challenge" từ phiên bản chạy dòng lệnh (sử dụng kiến thức từ chương 7) sang một ứng dụng giao diện đồ họa sử dụng thư viện tkinter. Chương trình không chỉ thực hiện việc đọc và xử lý nội dung câu hỏi từ một tệp văn bản có định dạng sẵn, mà còn cung cấp giao diện cho người dùng nhập đáp án, tính điểm dựa trên kết quả trả lời, và cập nhật nội dung câu hỏi sau mỗi lượt tương tác.

Thông qua việc thực hiện đề tài này, người học sẽ có cơ hội củng cố và nâng cao các kỹ năng lập trình cơ bản như thao tác với file văn bản, quản lý lỗi khi xử lý tệp (ví dụ: tệp không tồn tại hoặc định dạng sai), và làm việc với cấu trúc dữ liệu. Đồng thời, đề tài cũng giúp phát triển tư duy tổ chức chương trình theo hướng module hóa, tách biệt phần xử lý dữ liệu và phần giao diện, nhằm tăng tính rõ ràng, dễ mở rộng và dễ bảo trì cho ứng dụng.

Bên cạnh đó, việc thiết kế giao diện bằng thư viện tkinter còn giúp sinh viên làm quen với cách xây dựng ứng dụng tương tác thực tế – một kỹ năng thiết yếu trong lĩnh vực phát triển phần mềm. Đây là bước chuẩn bị cần thiết cho các dự án lập trình phức tạp hơn trong tương lai, đồng thời góp phần trang bị cho sinh viên nền tảng vững chắc để tiếp cận các công nghệ lập trình hiện đại.

# CHƯƠNG 1. GIỚI THIỆU ĐẦU BÀI

## 1.1. Đặt vấn đề

Trong các chương trình Python cơ bản, nhiều ứng dụng chỉ hoạt động trên giao diện dòng lệnh, gây hạn chế trong trải nghiệm người dùng. Đề tài "Trivia Challenge có GUI & file I/O" được thực hiện nhằm nâng cấp chương trình hỏi đáp đơn giản thành một ứng dụng có giao diện đồ họa trực quan, kết hợp giữa xử lý file dữ liệu và tương tác người dùng qua thư viện tkinter. Qua đó, sinh viên sẽ rèn luyện kỹ năng tổ chức chương trình, xử lý lỗi và xây dựng giao diện thân thiện.

## 1.2. Mô tả đề bài

Đề bài yêu cầu xây dựng một ứng dụng Trivia Challenge sử dụng ngôn ngữ Python với giao diện đồ họa được tạo bởi thư viện tkinter. Ứng dụng phải đọc dữ liệu câu hỏi và đáp án từ một file văn bản theo định dạng quy chuẩn giống như trong sách giáo trình.

Chương trình cần hiển thị lần lượt từng câu hỏi lên giao diện, cung cấp ô nhập liệu để người dùng nhập đáp án, và có nút “Nộp” để kiểm tra kết quả. Sau mỗi câu trả lời, ứng dụng sẽ cập nhật điểm số tương ứng và chuyển sang câu hỏi tiếp theo. Ngoài ra, ứng dụng còn có nút “Kết thúc” để người dùng có thể xem tổng điểm khi muốn dừng trò chơi.

Bài toán còn yêu cầu xử lý các trường hợp lỗi khi đọc file, ví dụ file không tồn tại hoặc có định dạng sai, nhằm đảm bảo tính ổn định của chương trình.

## 1.3. Các tính năng chính

- **Đọc câu hỏi từ file văn bản:**
- Ứng dụng đọc dữ liệu câu hỏi, các lựa chọn và đáp án từ file định dạng chuẩn.

- Có xử lý lỗi khi file không tồn tại hoặc định dạng không đúng.

- **Hiển thị câu hỏi trên giao diện GUI:**

  Câu hỏi được hiện thị rõ ràng trên cửa sổ ứng dụng với nhãn (Label) phù hợp.
- **Nhập và kiểm tra đáp án:**

  Người dùng nhập câu trả lời vào ô Entry, sau đó nhấn nút "Nộp" để chương trình kiểm tra và tính điểm.
- **Tính điểm và cập nhật liên tục:**

  Điểm số được tính tự động sau mỗi câu trả lời đúng và hiển thị trực tiếp trên giao diện.
- **Chuyển câu hỏi kế tiếp:**

  Sau khi nộp đáp án, chương trình tự động chuyển sang câu hỏi tiếp theo để người dùng trả lời.
- **Nút "Kết thúc":**

  Cho phép người dùng kết thúc trò chơi bất kỳ lúc nào và hiển thị tổng điểm hiện có.
- **Xử lý lỗi:**

  Bao gồm xử lý lỗi file không tìm thấy, lỗi định dạng file, và các trường hợp nhập dữ liệu không hợp lệ.

### 1.4. Những thách thức đặt ra

- Đọc và xử lý file dữ liệu đúng định dạng
- Thiết kế giao diện GUI thân thiện, dễ sử dụng
- Kết nối logic xử lý và giao diện
- Xử lý sự kiện và cập nhật trạng thái chương trình
- Đảm bảo tính mở rộng và dễ bảo trì

## 1.5. Kiến thức vận dụng

- **Xử lý file trong Python:**
- Biết cách mở, đọc, và xử lý dữ liệu từ file văn bản
- Sử dụng các thao tác đọc dòng, xử lý chuỗi, và kiểm tra lỗi file như file không tồn tại hoặc sai định dạng.
- **Lập trình hướng sự kiện với Tkinter:**
- Sử dụng thư viện tkinter để xây dựng giao diện đồ họa, tạo và quản lý các widget như Label, Entry, Button
- Hiểu cách gán và xử lý sự kiện (event handling) như nhấn nút, nhập dữ liệu.
- **Thiết kế và tổ chức chương trình theo module:** Phân chia chương trình thành các module riêng biệt như module xử lý file (đọc câu hỏi), module giao diện GUI và logic tính điểm, giúp mã nguồn rõ ràng, dễ bảo trì.
- **Xử lý logic và tính điểm:** Viết thuật toán kiểm tra câu trả lời người dùng, tính điểm dựa trên kết quả và cập nhật trạng thái câu hỏi tiếp theo.
- **Quản lý trạng thái ứng dụng:** Theo dõi tiến trình chơi, số câu hỏi đã trả lời, điểm số hiện tại và xử lý sự kiện kết thúc trò chơi.
- **Xử lý ngoại lệ và đảm bảo an toàn chương trình:** Sử dụng try-except để bắt lỗi khi đọc file hoặc nhập dữ liệu không hợp lệ, đảm bảo chương trình không bị dừng đột ngột.

# CHƯƠNG 2. CƠ SỞ LÝ THUYẾT

## 2.1. Làm việc với tệp văn bản trong Python (File I/O)

- Xử lý tệp là một kỹ năng cơ bản và quan trọng trong lập trình Python. Trong chương trình Trivia Challenge, các câu hỏi và đáp án được lưu trữ trong một tệp văn bản định dạng .txt, sau đó được đọc và xử lý tuần tự để hiển thị lên giao diện người dùng. Việc làm việc với tệp bao gồm các thao tác như mở tệp, đọc nội dung từng dòng, xử lý chuỗi và xử lý ngoại lệ khi tệp bị lỗi hoặc không tồn tại.
- Một số hàm và cú pháp quan trọng được sử dụng:

*open(filename, mode, encoding)*: Dùng để mở tệp.

Ví dụ: open("trivia.txt", "r", encoding="utf-8") để mở tệp đọc theo định dạng UTF-8.

- *readline()*: Đọc một dòng văn bản từ tệp. Mỗi lần gọi sẽ trả về dòng tiếp theo trong tệp.
- *strip()*: Loại bỏ ký tự xuống dòng \n và khoảng trắng ở đầu/cuối dòng.
- Điều này rất quan trọng khi so sánh đáp án người dùng với đáp án đúng.
- *try – except*: Dùng để xử lý lỗi khi làm việc với tệp, ví dụ khi tệp không tồn tại, chương trình sẽ không bị dừng đột ngột mà hiện thông báo lỗi thân thiện.
- Việc sử dụng đúng các thao tác với tệp giúp chương trình hoạt động ổn định, xử lý dữ liệu đầu vào hiệu quả và tránh được các lỗi phổ biến liên quan đến nhập xuất (I/O).

## 2.2. Xây dựng giao diện người dùng bằng Tkinter

- Trong Python, Tkinter là thư viện tiêu chuẩn dùng để xây dựng giao diện đồ họa (GUI – Graphical User Interface). Đây là công cụ phổ biến và dễ tiếp cận, đặc biệt phù hợp cho các ứng dụng nhỏ như trò chơi, công cụ tiện ích hay

phần mềm hỗ trợ học tập. Trong đề tài Trivia Challenge, Tkinter được sử dụng để tạo giao diện trực quan, thân thiện cho người dùng khi tương tác với câu hỏi và nhập đáp án.

- Một số thành phần cơ bản của Tkinter được sử dụng trong chương trình:

- *Tk()*: Tạo cửa sổ chính của ứng dụng.
- *Label*: Dùng để hiển thị văn bản như tiêu đề, câu hỏi, hoặc điểm số.
- *Entry*: Ô nhập liệu, cho phép người dùng gõ đáp án.
- *Button*: Nút nhấn, ví dụ như "Nộp" để xác nhận đáp án hoặc "Kết thúc" để hiện điểm.
- *Frame*: Khung chứa các thành phần khác, giúp bố cục giao diện gọn gàng, dễ tổ chức.

- Việc kết hợp hiệu quả các thành phần giao diện và sự kiện trong Tkinter giúp chương trình vận hành mượt mà, thân thiện với người dùng và dễ mở rộng.

## 2.3. Lập trình hướng đối tượng trong thiết kế chương trình (OOP)

Lập trình hướng đối tượng (Object-Oriented Programming – OOP) là một phương pháp tổ chức chương trình dựa trên việc mô hình hóa các đối tượng trong thực tế thành các class và object trong mã nguồn. OOP giúp mã dễ bảo trì, dễ mở rộng và tăng tính tái sử dụng.

Trong đề tài Trivia Challenge, OOP được vận dụng để tổ chức chương trình thành các thành phần có trách nhiệm rõ ràng. Ví dụ, có thể xây dựng một lớp để quản lý giao diện và xử lý logic trò chơi.

- Một số khái niệm OOP cơ bản được sử dụng:

- **Class (lớp):** Khuôn mẫu định nghĩa thuộc tính (dữ liệu) và phương thức (hành vi).
- **Object (đối tượng):** Thực thể được tạo ra từ lớp.
- Ví dụ: game = TriviaGame()

- **Phương thức (method):** Hàm định nghĩa bên trong lớp, thao tác trên dữ liệu của lớp.
- **Tính đóng gói (Encapsulation):** Ẩn giấu dữ liệu và chỉ cho phép truy cập thông qua các phương thức nhất định.

➢ Trong chương trình, class có thể dùng để:

- Quản lý câu hỏi, điểm số.
- Cập nhật giao diện mỗi khi sang câu hỏi mới.
- Kiểm tra đáp án và tính điểm.

➢ Việc áp dụng OOP giúp tách biệt rõ ràng giữa giao diện (GUI) và xử lý logic, giúp chương trình có cấu trúc rõ ràng, dễ kiểm soát và nâng cấp trong tương lai.

## 2.4. Kiểu dữ liệu chuỗi và xử lý nhập liệu

Chuỗi (string) là một kiểu dữ liệu cơ bản và thường xuyên được sử dụng, đặc biệt trong các chương trình có tương tác với người dùng. Trong đề tài Trivia Challenge, đáp án từ người dùng luôn được nhập dưới dạng chuỗi, do đó việc xử lý chuỗi đóng vai trò quan trọng để đảm bảo tính đúng đắn của kết quả.

❖ **Xử lý chuỗi**

➢ Gồm một số phương thức chuỗi cơ bản được sử dụng:

- *strip()*: Loại bỏ các ký tự khoảng trắng ở đầu và cuối chuỗi.
  Ví dụ: " python ".strip() → "python"
- *lower() / upper()*: Chuyển đổi chuỗi sang chữ thường hoặc chữ hoa để dễ dàng so sánh.
  Ví dụ: "Yes".lower() → "yes"

❖ **So sánh chuỗi**: sau khi xử lý chuẩn hóa, chuỗi đáp án của người dùng được so sánh với đáp án đúng từ tệp dữ liệu.

❖ **Xử lý nhập liệu**

- Vì người dùng có thể nhập thiếu, sai chính tả hoặc bỏ trống đáp án, chương trình cần kiểm tra tính hợp lệ trước khi chấm điểm. Cụ thể:
- Kiểm tra ô nhập có rỗng hay không.
- Xử lý trường hợp nhập dữ liệu không mong muốn (nếu có phân loại kiểu đáp án, như True/False hoặc số).
- Việc xử lý chuỗi và nhập liệu cẩn thận giúp chương trình vận hành mượt mà, chính xác và thân thiện hơn với người dùng cuối.

### 2.5. Xử lý điều kiện và vòng lặp

Trong lập trình nói chung và Python nói riêng, câu lệnh điều kiện và vòng lặp là hai cấu trúc điều khiển quan trọng giúp chương trình có thể ra quyết định và thực hiện thao tác lặp lại một cách linh hoạt. Trong chương trình Trivia Challenge, chúng được sử dụng để:

- Kiểm tra đáp án đúng hay sai.
- Chuyển sang câu hỏi tiếp theo.
- Lặp qua từng khối câu hỏi trong tệp dữ liệu.

❖ **Câu lệnh điều kiện (if, elif, else)**

- Được sử dụng để xử lý các tình huống có thể xảy ra khi người dùng tương tác:
- Kiểm tra ô nhập có bị bỏ trống không.
- So sánh đáp án nhập vào với đáp án đúng.
- Hiển thị điểm hoặc thông báo sau mỗi lượt trả lời.

❖ **Vòng lặp (while, for)**

- Trong chương trình, vòng lặp thường dùng để đọc tuần tự các câu hỏi từ file hoặc xử lý từng khối câu hỏi
- Cũng có thể dùng vòng lặp “for” nếu đọc dữ liệu từ danh sách có sẵn.
- Việc sử dụng hợp lý các cấu trúc điều kiện và vòng lặp giúp chương trình:
- Hoạt động theo luồng logic rõ ràng.

- Xử lý tuần tự các câu hỏi một cách tự động.
- Giảm lặp mã và tránh lỗi thủ công khi chuyển câu hỏi.

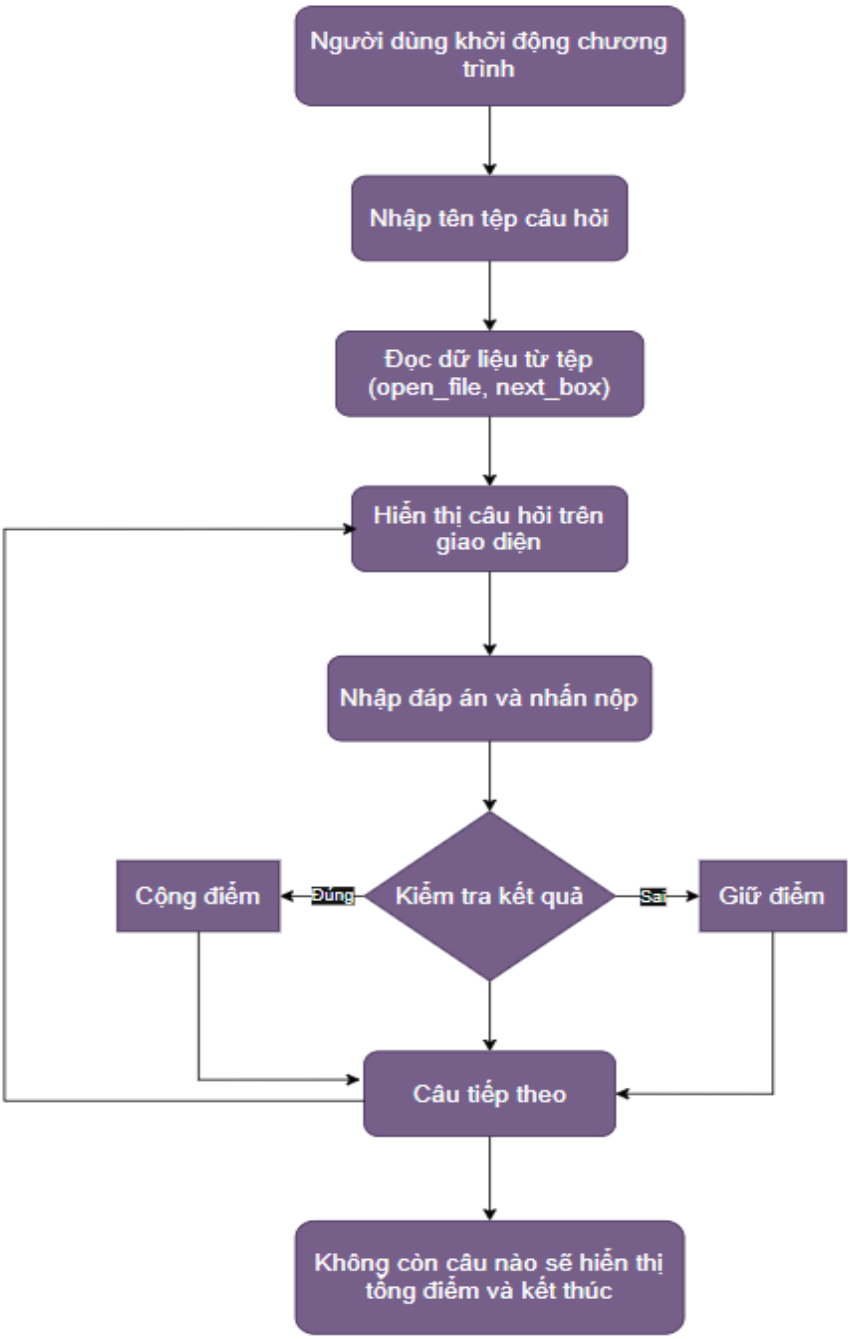
# CHƯƠNG 3. THIẾT KẾ VÀ XÂY DỰNG CHƯƠNG TRÌNH

## 3.1. Sơ đồ khối hệ thống

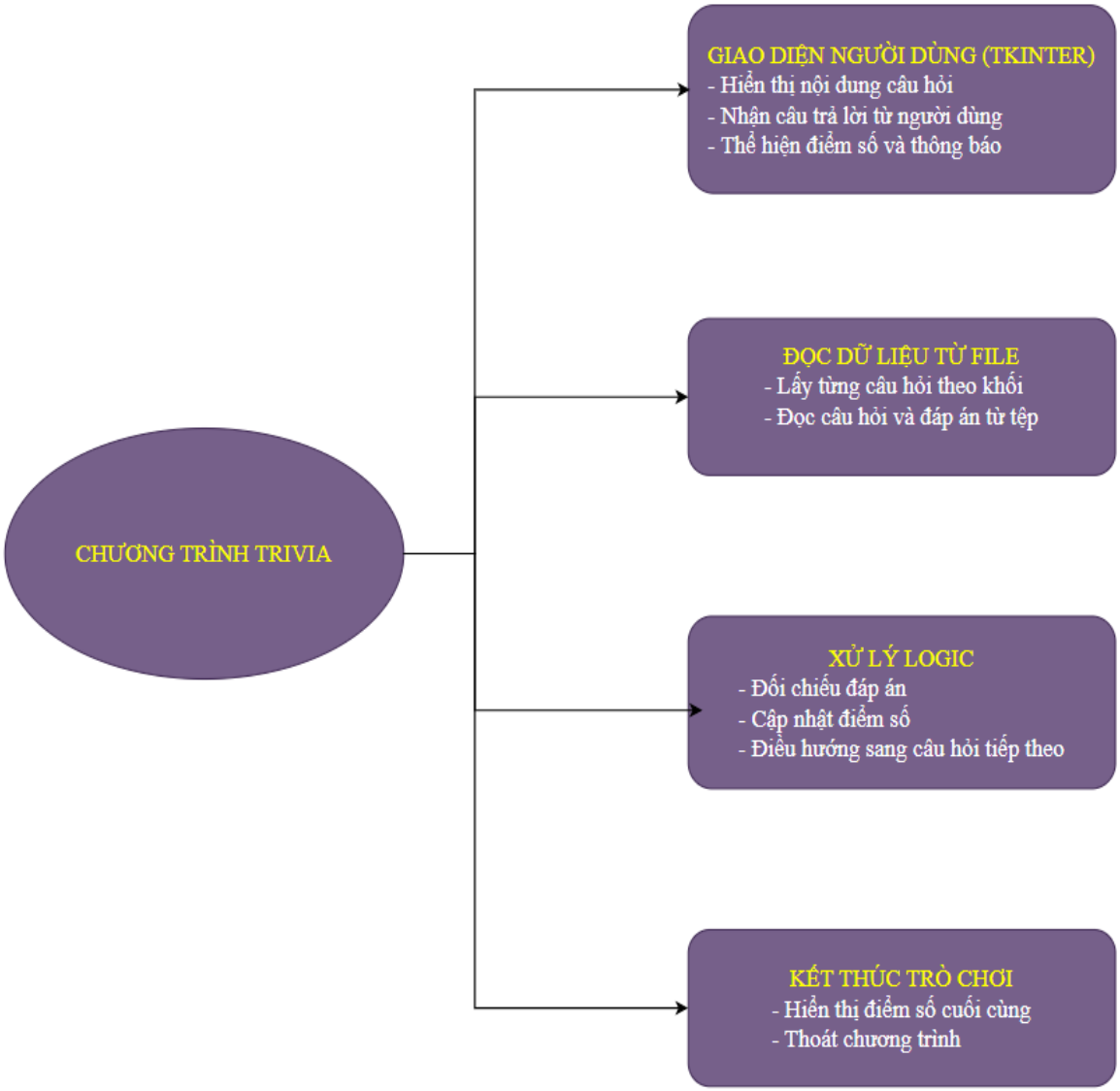
Chương trình Trivia Challenge được thiết kế với tư duy hướng đối tượng, chia thành các module chính, mỗi module đảm nhiệm một vai trò cụ thể trong toàn bộ quá trình hoạt động. Việc phân chia module rõ ràng giúp chương trình dễ phát triển, dễ bảo trì và thuận tiện khi mở rộng về sau.

Các module chính bao gồm:

- **Module đọc dữ liệu từ file**
  - Chức năng: Đọc câu hỏi, các lựa chọn và đáp án từ tệp văn bản định dạng sẵn.
  - Xử lý lỗi: Kiểm tra định dạng file, đảm bảo mỗi khối có đủ thông tin cần thiết.
- **Module xử lý giao diện người dùng (GUI)**
  - Chức năng: Tạo và điều khiển các thành phần giao diện bằng thư viện Tkinter.
  - Tương tác người dùng: Giao diện trực quan giúp người dùng dễ sử dụng, phản hồi ngay lập tức sau khi nộp câu trả lời.
- **Module kiểm tra đáp án và tính điểm**
  - Chức năng: So sánh đáp án người dùng với đáp án đúng, cập nhật điểm.
  - Xử lý chuỗi: Sử dụng **strip()**, **lower()** để đảm bảo so sánh chính xác bất kể cách nhập.
  - Cập nhật giao diện: Sau mỗi câu trả lời, chương trình cập nhật điểm và hiển thị câu tiếp theo.
- **Module điều khiển luồng trò chơi**
  - Kiểm soát toàn bộ quá trình trò chơi: bắt đầu, chuyển câu hỏi, kết thúc.

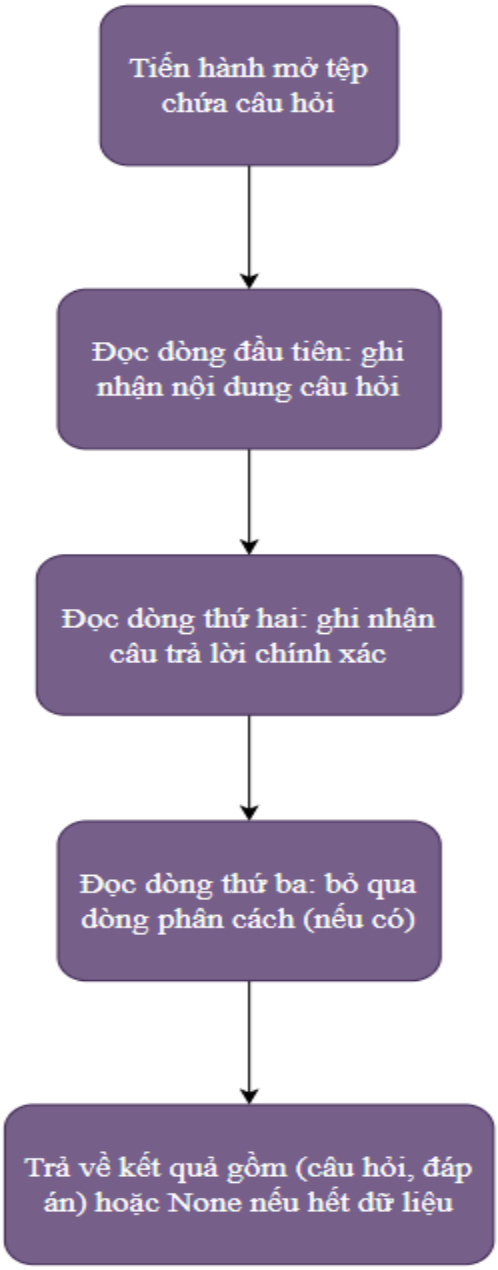



*Hình 1. Sơ đồ khối hệ thống*

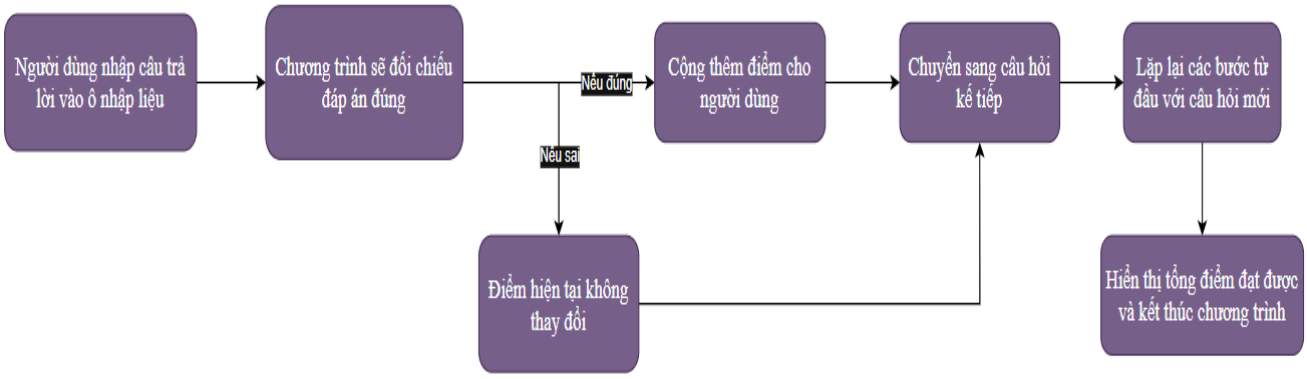



*Hình 2. Sơ đồ phân cấp chức năng*

### 3.2. Sơ đồ khối các thuật toán chính




*Hình 3. Sơ đồ khối xử lý*




*Hình 4. Sơ đồ thuật toán xử lý câu hỏi*

## 3.3. Cấu trúc dữ liệu

❖ **Dữ liệu câu hỏi lưu trữ trong tệp văn bản**

➢ Trong chương trình, toàn bộ nội dung câu hỏi và đáp án được lưu trữ trong một tệp văn bản có tên **trivia.txt**, sử dụng định dạng cố định để thuận tiện cho việc đọc và xử lý dữ liệu. Cụ thể, mỗi câu hỏi được trình bày dưới dạng 3 dòng liên tiếp, theo cấu trúc sau:

- **Dòng thứ nhất**: chứa nội dung của câu hỏi.
- **Dòng thứ hai**: là đáp án đúng tương ứng với câu hỏi.
- **Dòng thứ ba**: là một dòng trống, đóng vai trò ngăn cách giữa các câu hỏi.

➢ Với cấu trúc đơn giản và nhất quán này, chương trình có thể lần lượt đọc từng nhóm dòng để xử lý câu hỏi một cách hiệu quả và tránh nhầm lẫn.

❖ **Cấu trúc dữ liệu trong chương trình**

➢ Trong chương trình, cấu trúc dữ liệu được tổ chức đơn giản để thuận tiện cho việc đọc, xử lý và hiển thị thông tin từ file.

➢ Câu hỏi và đáp án được lưu dưới dạng **tuple** gồm hai phần: nội dung câu hỏi và đáp án đúng. Danh sách các câu hỏi sẽ được lưu trong **một list**, mỗi phần tử là một tuple chứa dữ liệu của một câu hỏi.

- Ngoài ra, chương trình sử dụng một số biến để lưu **trạng thái tạm thời**, bao gồm:
- Biến *score*: lưu tổng điểm người chơi đạt được.
- Biến *current_question*: chỉ mục của câu hỏi đang hiển thị.
- Biến *question_list*: danh sách toàn bộ các câu hỏi đọc từ file.
- Cấu trúc này giúp chương trình dễ dàng truy xuất từng câu hỏi theo chỉ mục, so sánh đáp án người dùng nhập và cập nhật điểm trong giao diện GUI.

❖ Nội dung trong tệp *trivia.txt*:

*What is 2 + 2 = ?*

*4*

*What is your name?*

*Việt Anh*

*Where are you from?*

*Việt Nam*

*How old are you?*

*21*

Danh sách kết quả sau khi xử lý:

[(“What is 2 + 2 = ?”, “4”, “1”),

(“What is your name?”, “Việt Anh”, “1”),

(“Where are you from?”, “Việt Nam”, “1”)

(“How old are you?”, “21”, “1”)]

### 3.4. Chương trình

```
import tkinter as tk
from tkinter import messagebox, simpledialog
from PIL import Image, ImageTk  # Import Pillow để xử lý ảnh

# --- File I/O functions ---
def open_file(filename):
    try:
        return open(filename, "r", encoding="utf-8")
    except FileNotFoundError:
        messagebox.showerror("Lỗi", f"Không tìm thấy file: {filename}")
        return None

def next_block(file):
    while True:
        question = file.readline()
        if question == "":
            return None, None
        question = question.strip()
        if question != "":
            break
    while True:
        answer = file.readline()
        if answer == "":
            return None, None
        answer = answer.strip()
        if answer != "":
```

```
            break
    file.readline()  # Dòng trống để ngăn cách câu hỏi
    return question, answer

# --- GUI Class ---
class TriviaGUI:
    def __init__(self, master, filename):
        self.master = master
        self.master.title("Trivia Challenge")

        # --- Kích hoạt toàn màn hình ---
        self.master.attributes('-fullscreen', True)
        self.master.bind("<Escape>", lambda e: self.master.attributes('-fullscreen',
False))  # Nhấn Esc để thoát toàn màn hình

        # --- Thêm hình nền ---
        image = Image.open("background.jpg")  # Đổi tên file ảnh nếu cần
        self.bg_image = ImageTk.PhotoImage(image)
        self.bg_label = tk.Label(self.master, image=self.bg_image)
        self.bg_label.place(x=0, y=0, relwidth=1, relheight=1)

        self.score = 0
        self.filename = filename
        self.file = open_file(self.filename)
        if not self.file:
            self.master.destroy()
            return
```

```
        self.question, self.answer = next_block(self.file)
        if self.question is None:
            messagebox.showerror("Lỗi", "File không đúng định dạng hoặc rỗng.")
            self.master.destroy()
            return

        self.create_widgets()

    def create_widgets(self):
        # --- Khung chứa nội dung ---
        self.frame = tk.Frame(self.master, bg="#fffaf0", width=600, height=300)
        self.frame.place(relx=0.5, rely=0.5, anchor="center")  # Đặt chính giữa màn
hình

        self.lbl_score = tk.Label(self.frame, text=f"Điểm: {self.score}",
font=("Verdana", 16, "bold"),
                                  bg="#fffaf0", fg="#228b22")
        self.lbl_score.pack(pady=5)

        self.lbl_question = tk.Label(self.frame, text=self.question, wraplength=500,
font=("Helvetica", 18),
                                     bg="#fffaf0", fg="#000080", justify="center")
        self.lbl_question.pack(pady=15)

        self.entry_answer = tk.Entry(self.frame, font=("Helvetica", 16), width=40,
justify="center", bd=3, relief="sunken")
```

```
        self.entry_answer.pack(pady=5)
        self.entry_answer.bind("<Return>", lambda e: self.submit_answer())
        self.entry_answer.focus()

        # --- Nút hành động ---
        self.button_frame = tk.Frame(self.master, bg="#fffaf0")
        self.button_frame.place(relx=0.5, rely=0.7, anchor="center")  # Đặt chính
giữa dưới phần câu hỏi

        self.btn_submit = tk.Button(self.button_frame, text="Nộp", font=("Helvetica",
14, "bold"),
                                    command=self.submit_answer, bg="#4CAF50", fg="white",
width=15, height=2, bd=0, activebackground="#45a049")
        self.btn_submit.grid(row=0, column=0, padx=10)

        self.btn_end = tk.Button(self.button_frame, text="Kết thúc",
font=("Helvetica", 14, "bold"),
                                 command=self.end_game, bg="#f44336", fg="white",
width=15, height=2, bd=0, activebackground="#e53935")
        self.btn_end.grid(row=0, column=1, padx=10)

    def submit_answer(self):
        user_answer = self.entry_answer.get().strip()

        # --- Kiểm tra nếu người dùng không nhập gì ---
        if not user_answer:
            messagebox.showwarning("Cảnh báo", "Vui lòng nhập câu trả lời trước
```

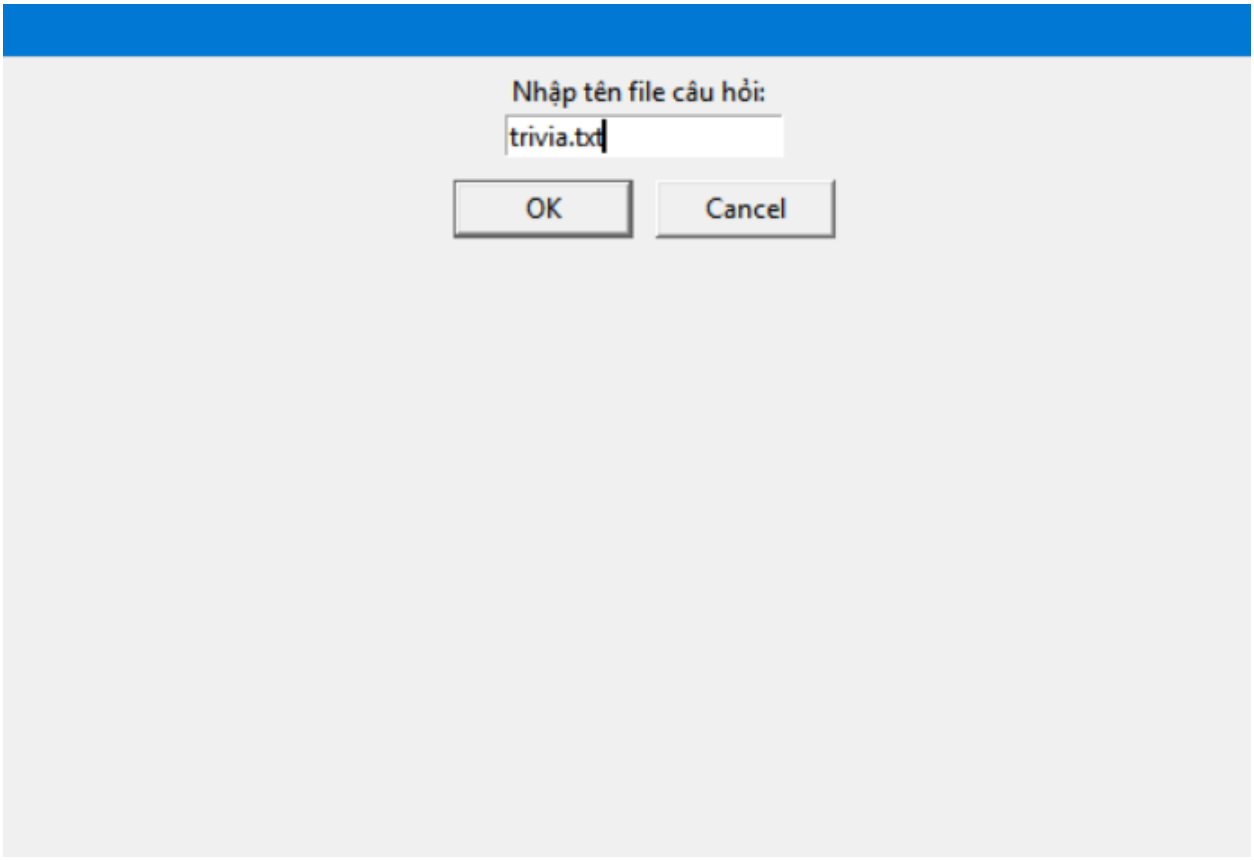
```
khi nộp!")
            return

        if user_answer.lower() == self.answer.lower():
            self.score += 1
            self.lbl_score.config(text=f"Điểm: {self.score}")
            messagebox.showinfo("Đúng rồi!", "Chính xác! Bạn thật tuyệt!")
        else:
            messagebox.showinfo("Sai rồi!", f"Đáp án đúng là: {self.answer}")

        self.next_question()

    def next_question(self):
        self.question, self.answer = next_block(self.file)
        if self.question is None:
            self.end_game()
        else:
            self.lbl_question.config(text=self.question)
            self.entry_answer.delete(0, tk.END)
            self.entry_answer.focus()

    def end_game(self):
        messagebox.showinfo("Kết thúc trò chơi", f"Tổng điểm của bạn:
{self.score}")
        self.master.destroy()
        if self.file:
            self.file.close()
```

```
# --- Main ---
if __name__ == "__main__":
    root = tk.Tk()
    filename = simpledialog.askstring("File câu hỏi", "Nhập tên file câu hỏi:",
initialvalue="trivia.txt")
    if filename:
        app = TriviaGUI(root, filename)
        root.mainloop()
```

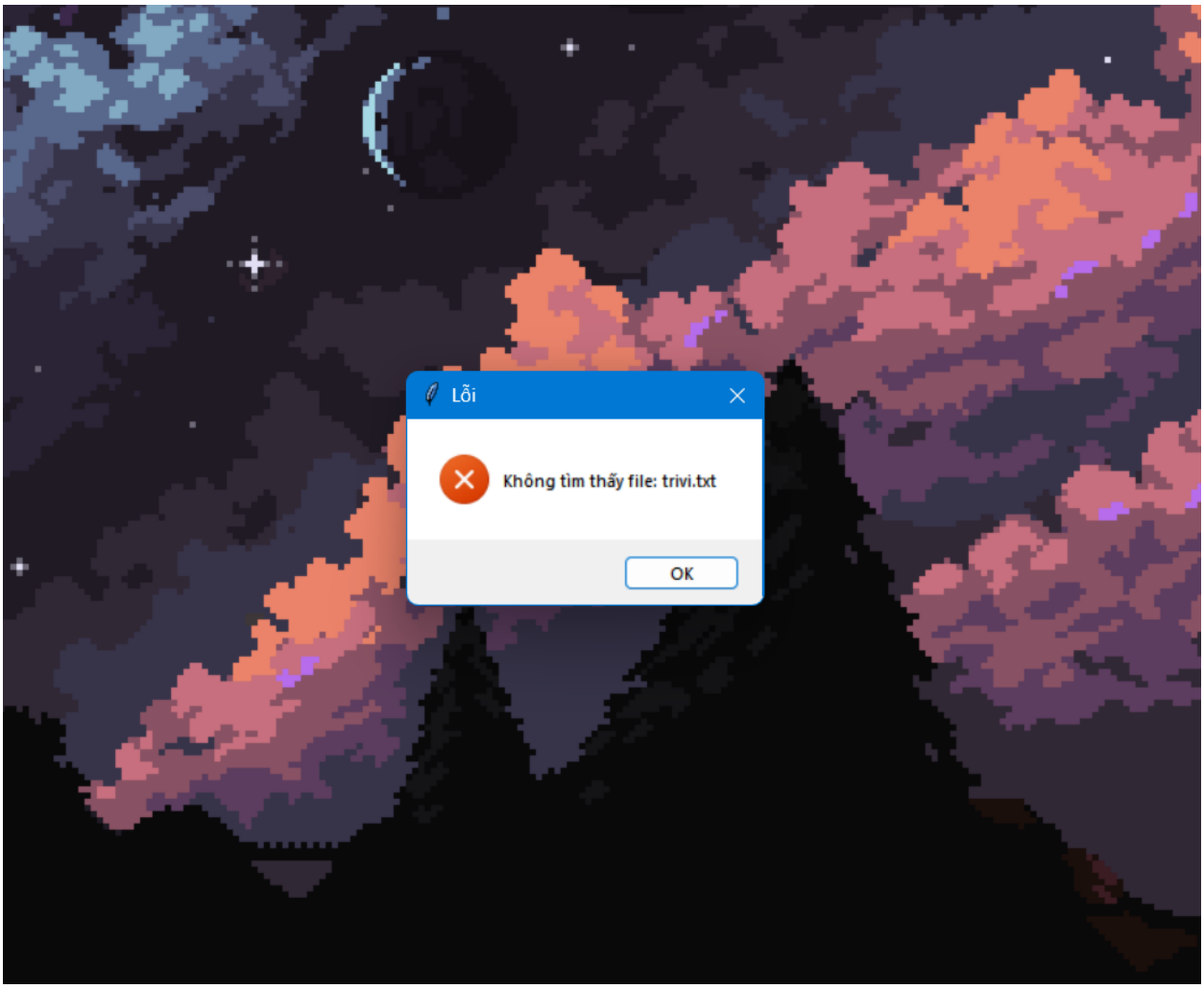
# CHƯƠNG 4. THỰC NGHIỆM VÀ KẾT LUẬN

## 4.1. Thực nghiệm

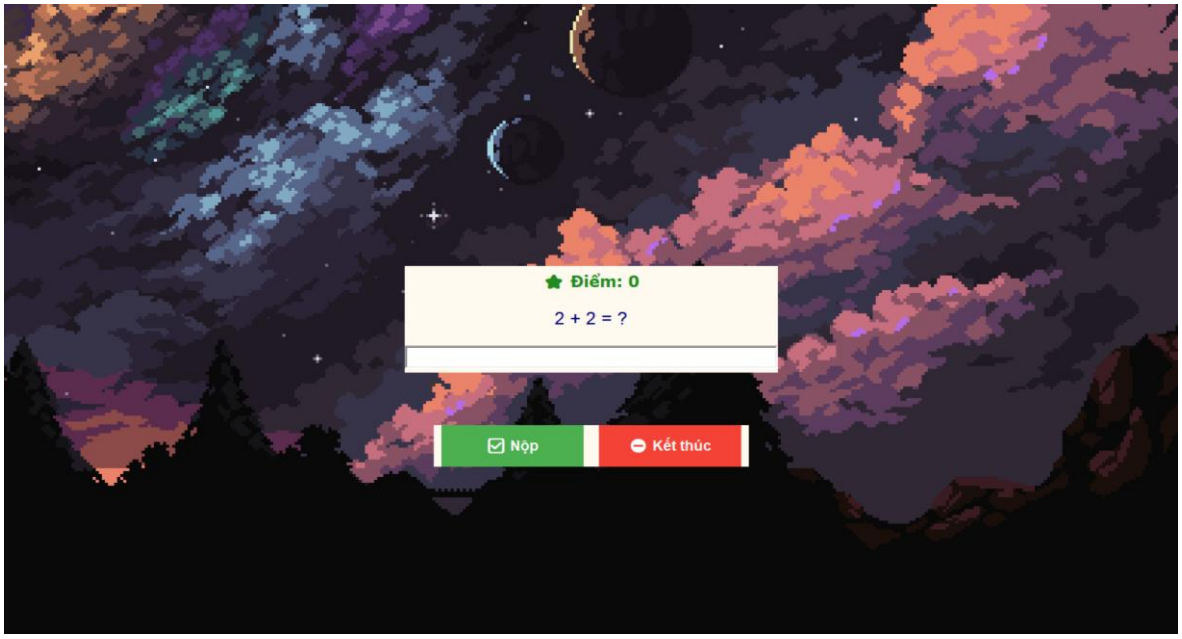
- Bắt đầu chạy chương trình, kiểm tra các tính năng và luôn đảm bảo chương trình chạy đúng theo yêu cầu của đề tài
- Dưới đây là các ảnh minh hoạ chạy thực thi chương trình



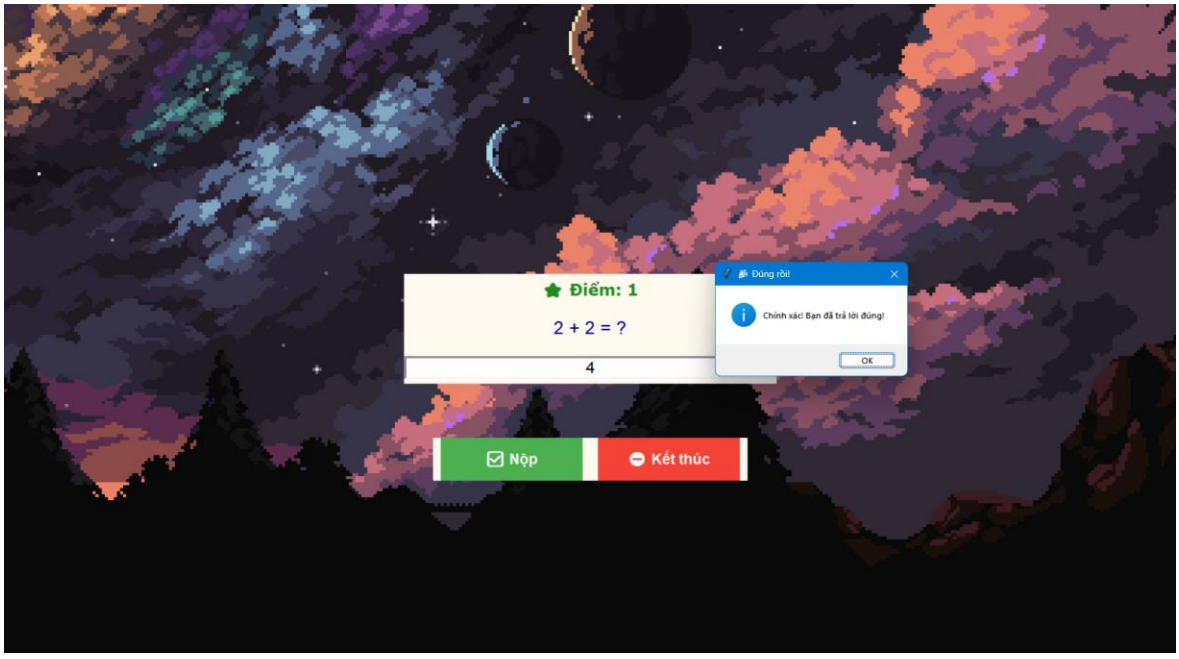

*Nhập đúng tên tệp là "trivia.txt" theo đề tài sẽ mở chương trình và bắt đầu đọc nội dung*



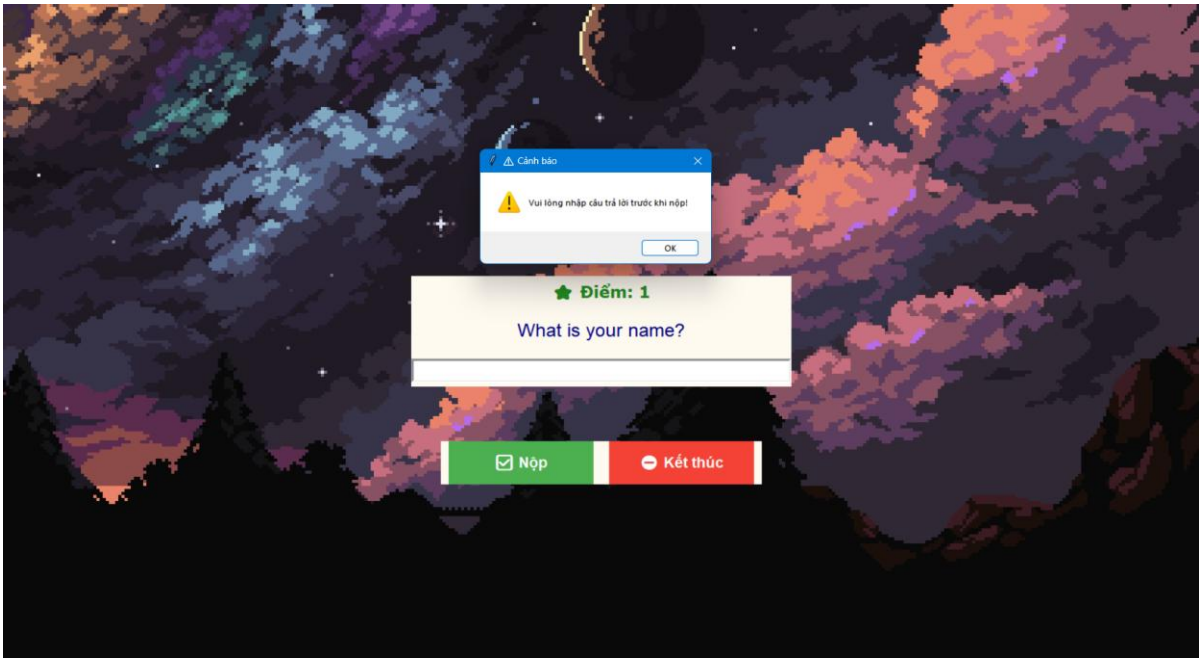

*Nếu nhập không đúng theo định dạng như đề tài sẽ xuất hiện giao diện báo lỗi không thể tìm thấy file*



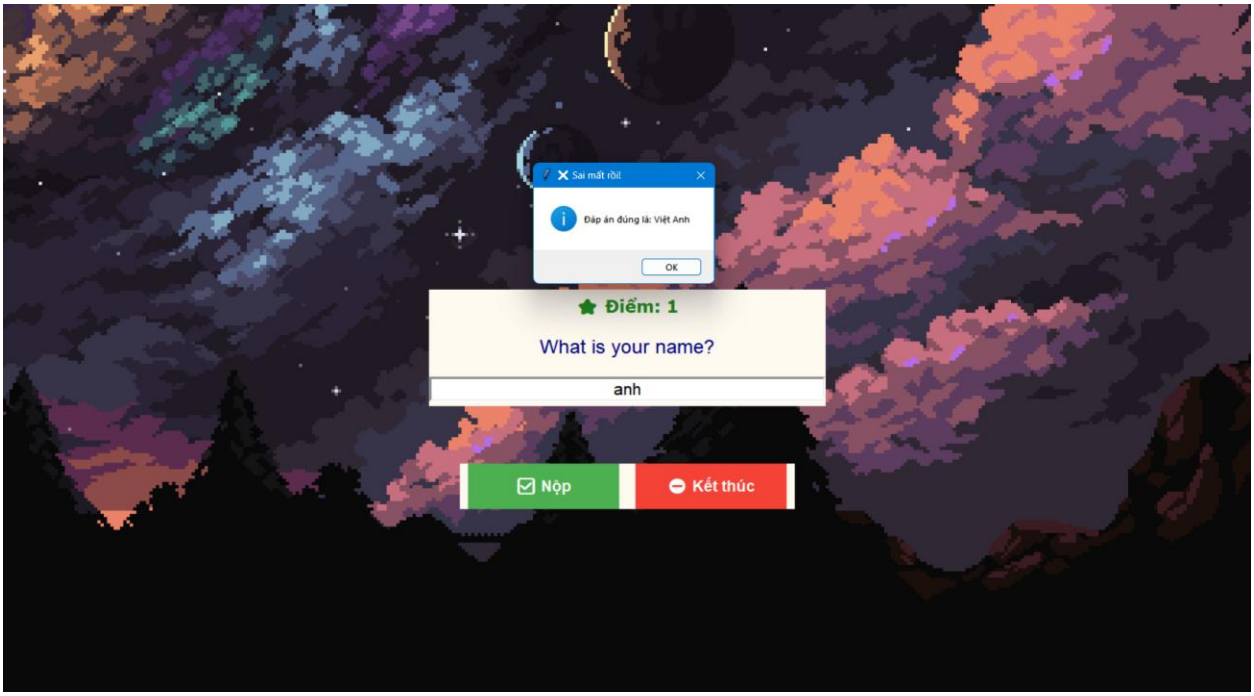

*Đây là giao diện sau khi nhập tên tệp đúng và sẽ hiện ra các mục: câu hỏi, phần nhập đáp án, nộp và kết thúc*



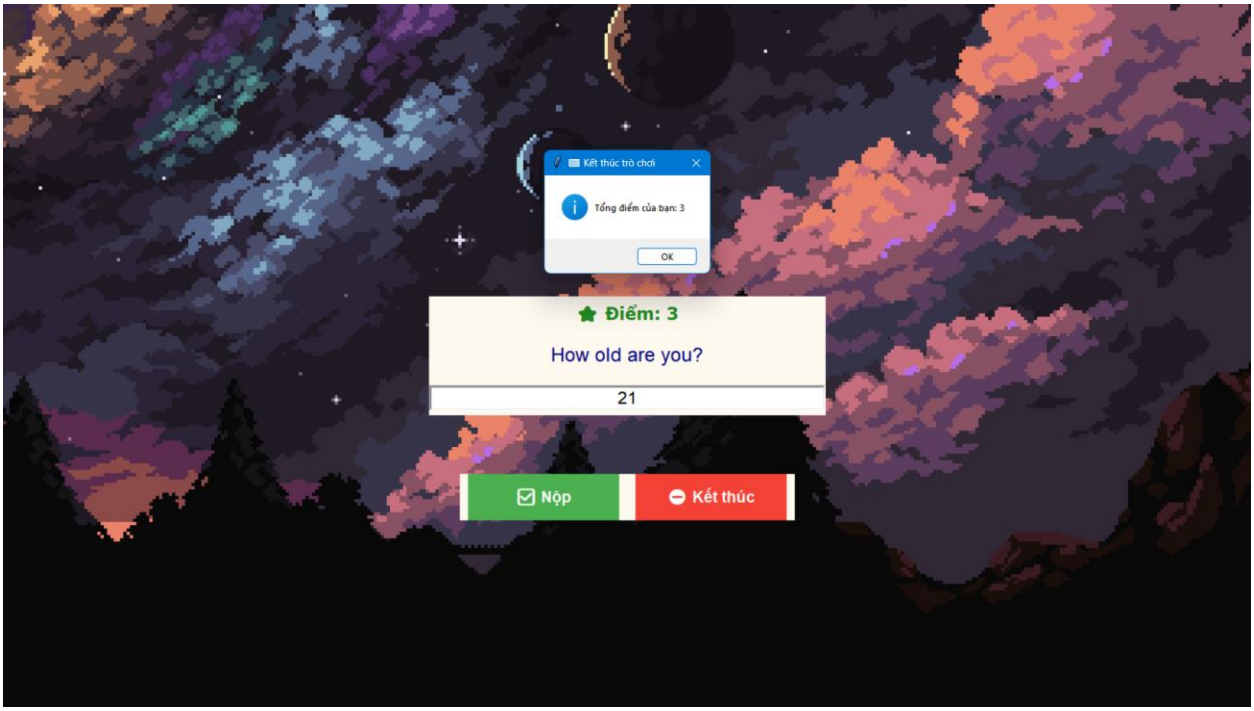

*Nhập đúng đáp án chương trình sẽ thông báo đã trả lời đúng và được cộng điểm*




*Nếu không nhập đáp án thì chương trình sẽ thông báo không được bỏ trống*




*Khi nhập sai đáp án chương trình sẽ thông báo đáp án đúng và sẽ không được cộng điểm*




*Sau khi hoàn thành hết các câu hỏi hoặc ấn nút kết "kết thúc" thì chương trình sẽ thông báo số điểm tổng kết và kết thúc chương trình*

## 4.2. Kết luận

➢ Sản phẩm đã thực hiện được:

- Thiết kế giao diện trực quan, dễ sử dụng bằng thư viện Tkinter, hỗ trợ nhập liệu và hiển thị câu hỏi rõ ràng.
- Nạp dữ liệu từ tệp văn bản và xử lý đầu vào từ người chơi một cách chính xác và ổn định.
- So sánh câu trả lời với đáp án đúng, đồng thời tính và hiển thị điểm số ngay sau mỗi lượt trả lời.
- Tự động chuyển sang câu hỏi kế tiếp và kiểm soát việc kết thúc trò chơi khi hết dữ liệu.
- Đảm bảo an toàn khi thao tác với tệp, đồng thời hiển thị thông báo tổng kết điểm sau cùng.

➢ Bài học rút ra:

- Hiểu rõ cách triển khai giao diện người dùng đơn giản với thư viện Tkinter, từ bố cục đến xử lý sự kiện.
- Rèn luyện kỹ năng làm việc với tệp văn bản trong Python thông qua các hàm như *open(), read(),* và *close().*
- Làm quen với cách chia chương trình thành các module nhỏ, dễ quản lý và tái sử dụng.
- Áp dụng được tư duy lập trình hướng đối tượng (OOP) để xây dựng các lớp điều khiển giao diện và logic xử lý.
- Nắm được cách sử dụng try–except để xử lý lỗi khi thao tác file hoặc khi nhập sai dữ liệu đầu vào.
- Cải thiện khả năng lập kế hoạch và phát triển phần mềm theo từng bước, từ đọc yêu cầu đến kiểm thử.

➢ Định hướng cải tiến trong tương lai:

- Bổ sung chức năng câu hỏi trắc nghiệm với nhiều lựa chọn (A, B, C, D) để người chơi dễ thao tác và giảm lỗi nhập liệu.
- Thiết kế giao diện riêng dành cho quản trị viên nhằm thêm, sửa hoặc xóa câu hỏi một cách trực quan.
- Ghi lại lịch sử điểm số của từng người chơi và xây dựng hệ thống bảng xếp hạng theo thời gian thực.
- Nâng cấp trải nghiệm người dùng bằng cách sử dụng màu sắc hợp lý, hiệu ứng chuyển động nhẹ và âm thanh khi trả lời đúng/sai.
- Mở rộng tính năng lựa chọn tệp câu hỏi từ máy người dùng để tăng tính linh hoạt.
- Phát triển phiên bản đa nền tảng hoặc có thể chạy trên trình duyệt thông qua công nghệ web (nếu mở rộng).

# TÀI LIỆU THAM KHẢO


- **Sweigart, A. (2015).** ***Automate the Boring Stuff with Python: Practical Programming for Total Beginners*****. No Starch Press.**
- **Python Software Foundation. (2023).** ***The Python Tutorial*****. Truy cập tại: https://docs.python.org/3/tutorial/**
- **Geeks for Geeks. (2024).** ***Python Tkinter Tutorial*****. Truy cập tại: https://www.geeksforgeeks.org/python-gui-tkinter/**
- **Zelle, J. (2010).** ***Python Programming: An Introduction to Computer Science*****. Franklin, Beedle & Associates.**
- **Real Python. (2024).** ***Working with Files in Python*****. Truy cập tại: https://realpython.com/working-with-files-in-python/**