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 : **LƯƠNG VĂN HỌC**

LỚP : **K58KTP**

GIÁO VIÊN GIẢNG DẠY : **TS.NGUYỄN VĂN HUY**

LINK GITHUB : **LINK GITHUB**

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

TÊN ĐỀ TÀI: XÂY GAME BLACKJACK VỚI GUI

SINH VIÊN	**:**	**LƯƠNG VĂN HỌC**
LỚP	**:**	**K58KTP**
GIÁO VIÊN GIẢNG DẠY	**:**	**TS.NGUYỄN VĂN HUY**
LINK GITHUB	**:**	**LINK GITHUB**

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 TÚC MÔN HỌC

MÔN HỌC : LẬP TRÌNH PYTHON

BỘ MÔN : CÔNG NGHỆ THÔNG TIN

Sinh viên: Lương Văn Học *MSSV:* *K225480106025*

Lớp: K58KTP *Khoá: K58*

Ngành học: Kỹ thuật phần mềm

Giáo viên hướng dẫn: Ts.Nguyễn Văn Huy

Ngày giao đề : *Ngày hoàn thành:*

Tên đề tài:

Xây game Blackjack với GUI

Yêu cầu:

Đầu vào – đầu ra:

- Đầu vào: Nút “Hit”, “Stand”.
- Đầu ra: Hiển thị lá bài (Label/Icon), điểm, kết quả.

Tính năng yêu cầu:

- Sử dụng class BJ_Card, BJ_Deck, BJ_Hand, BJ_Game.
- GUI cập nhật khi rút bài.
- Xử lý bust (>21) và so sánh với dealer.

Kiểm tra & kết quả mẫu:

- Rút A,10 → tài – Hiển thị “Blackjack!”
- Các bước triển khai:
- Import module cards, games.
- Class GUI: Frame người chơi và dealer.
- Button “Hit” và “Stand” gọi phương thức game.
- Cập nhật kết quả trên Label/Messagebox.

GIÁO VIÊN HƯỚNG DẪN

Ts.Nguyễn Văn Huy

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Thái Nguyên, ngày tháng 06 năm 2025

GIÁO VIÊN HƯỚNG DẪN

(Ký ghi rõ họ tên)

MỤC LỤC

LỜI CAM ĐOAN

Em xin cam đoan rằng toàn bộ nội dung bài tập lớn này là kết quả nghiên cứu và thực hiện cá nhân của em. Tất cả các mã nguồn, giải thuật, và phương pháp sử dụng trong bài đều được em tự tìm hiểu và áp dụng. Em cam kết không sao chép, gian lận hay sử dụng bất kỳ tài liệu nào không được phép trong quá trình thực hiện bài tập này.

Mọi tài liệu tham khảo, nếu có, đều được em trích dẫn và ghi rõ nguồn gốc đầy đủ. Em hiểu rằng việc vi phạm các quy định về đạo đức học thuật sẽ dẫn đến những hậu quả nghiêm trọng đối với kết quả học tập của mình.

Bài tập này hoàn toàn được thực hiện dưới sự hướng dẫn của thầy Nguyễn Văn Huy, và em chịu trách nhiệm hoàn toàn về nội dung của bài tập.

Kính đề nghị quý thầy cô xem xét và đánh giá.

Sinh viên thực hiện

DANH MỤC CÁC TỪ VIẾT TẮT

1. *BJ_Card: Blackjack Card (Lá bài Blackjack)*
2. *BJ_Deck: Blackjack Deck (Bộ bài Blackjack)*
3. *BJ_Hand: Blackjack Hand (Tay bài Blackjack)*
4. *BJ_Game: Blackjack Game (Trò chơi Blackjack)*
5. *GUI: Graphical User Interface (Giao diện đồ họa người dùng)*
6. *OOP: Object-Oriented Programming (Lập trình hướng đối tượng)*

DANH MỤC CÁC BẢNG VÀ HÌNH VẼ, ĐỒ THỊ

LỜI NÓI ĐẦU

Trong khuôn khổ môn học Python, bài tập lớn này được thiết kế nhằm giúp sinh viên áp dụng lý thuyết và kỹ năng lập trình đã học vào việc phát triển một trò chơi thực tế. Cụ thể, bài tập yêu cầu xây dựng một phiên bản đơn giản của trò chơi Blackjack với giao diện người dùng, cho phép người chơi tham gia vào một ván bài đầy thử thách.

Game Blackjack là một trong những trò chơi thẻ phổ biến, đòi hỏi người chơi tính toán điểm số và đưa ra các quyết định như "Hit" hoặc "Stand" để có được tổng điểm cao nhất mà không vượt quá 21. Trong bài này, em sẽ sử dụng các lớp như BJ_Card, BJ_Deck, BJ_Hand, BJ_Game để mô phỏng các chức năng cơ bản của trò chơi, đồng thời tích hợp GUI để tạo ra trải nghiệm người chơi trực quan và dễ dàng tương tác.

Các tính năng yêu cầu bao gồm việc cập nhật GUI khi người chơi rút bài, xử lý trường hợp "bust" khi tổng điểm vượt quá 21, và so sánh điểm của người chơi với dealer để xác định kết quả. Bài tập cũng yêu cầu sử dụng các button "Hit" và "Stand" để tương tác với trò chơi, đồng thời hiển thị kết quả ván bài qua các label hoặc messagebox.

Bài tập này không chỉ giúp em rèn luyện kỹ năng lập trình Python mà còn nâng cao khả năng sử dụng các thư viện hỗ trợ như tkinter để xây dựng giao diện người dùng. Em hy vọng qua bài tập này sẽ hiểu sâu hơn về cách lập trình game với Python, đặc biệt là việc kết hợp các lớp đối tượng và GUI.

Xin cảm ơn thầy Nguyễn Văn Huy đã hướng dẫn và tạo cơ hội để em áp dụng kiến thức vào thực tế.

Thái Nguyên, ngày … tháng … năm 2025
Nhóm sinh viên thực hiện
(Ký và ghi rõ họ tên)

LỜI CẢM ƠN

Em xin chân thành cảm ơn thầy Nguyễn Văn Huy đã tận tình hướng dẫn và hỗ trợ em trong suốt quá trình thực hiện bài tập lớn này. Sự hướng dẫn của thầy không chỉ giúp em hiểu sâu hơn về lập trình Python mà còn cung cấp cho em những kiến thức quý báu trong việc áp dụng lý thuyết vào thực tế.

Em cũng xin cảm ơn các bạn đồng môn và những người đã hỗ trợ, chia sẻ kiến thức và kinh nghiệm trong suốt thời gian qua. Những ý tưởng và góp ý của mọi người đã giúp em hoàn thiện bài tập này một cách tốt nhất.

Cuối cùng, em xin cảm ơn gia đình và bạn bè đã luôn động viên và ủng hộ em trong quá trình học tập và nghiên cứu.

CHƯƠNG 1. GIỚI THIỆU ĐẦU BÀI

1.1. Giới thiệu trò chơi Blackjack

Blackjack là một trò chơi bài phổ biến, được chơi với bộ bài tây 52 lá. Trò chơi này thường có hai đối thủ: người chơi và nhà cái (dealer). Mục tiêu của trò chơi là có điểm số càng gần 21 càng tốt, nhưng không được vượt quá 21 điểm. Người chơi sẽ phải đưa ra quyết định về việc rút thêm bài ("Hit") hoặc dừng lại ("Stand") tùy vào điểm số của mình.

Hình 1 Blackjack

1.2. Cách chơi

1.2.1. Mô hình trò chơi

- Trò chơi sẽ diễn ra giữa người chơi và nhà cái, với mỗi bên được chia 2 lá bài.
- Các lá bài có giá trị như sau:
  - Các lá bài từ 2 đến 9 có giá trị tương ứng với số trên lá bài.
  - Các lá 10, J, Q, K đều có giá trị 10 điểm.
  - Lá A có thể có giá trị là 1 hoặc 11 điểm, tùy vào tình huống. Nếu việc tính A là 11 giúp người chơi không bị vượt quá 21, A sẽ tính là 11, ngược lại sẽ tính là 1.

Hình 2 Tính điểm trong backjack

1.2.2 Quy trình chơi game

- **Chia bài**: Vào đầu mỗi ván bài, nhà cái sẽ chia cho người chơi và nhà cái mỗi người 2 lá bài. Những lá bài này sẽ được lật ngửa, nghĩa là người chơi sẽ nhìn thấy bài của mình nhưng chỉ thấy một lá bài của nhà cái.
- **Các lựa chọn của người chơi:**
  - Hit: Người chơi có thể rút thêm bài bằng cách nhấn nút "Hit". Khi rút bài, tổng điểm của người chơi sẽ tăng lên, và nếu tổng điểm vượt quá 21, người chơi sẽ thua ngay lập tức.
  - Stand: Nếu người chơi cảm thấy điểm hiện tại đủ cao, họ có thể dừng lại và chọn "Stand". Khi đó, lượt chơi của người chơi sẽ kết thúc và nhà cái sẽ bắt đầu hành động.
- **Tính điểm:** Sau khi người chơi dừng lại, nhà cái sẽ mở bài và tính điểm. Nếu tổng điểm của nhà cái <= 16, nhà cái sẽ phải rút thêm bài cho đến khi có tổng điểm >= 17. Nếu tổng điểm của nhà cái vượt quá 21, nhà cái sẽ thua.

1.2.3. So sánh điểm và xác định kết quả

- **Người chơi thắng:** Nếu điểm của người chơi cao hơn điểm của nhà cái mà không vượt quá 21, người chơi sẽ thắng.
- **Nhà cái thắng:** Nếu điểm của nhà cái cao hơn người chơi và không vượt quá 21, nhà cái sẽ thắng.

- **Hòa:** Nếu điểm của người chơi và nhà cái bằng nhau, kết quả sẽ là hòa.

Các tình huống đặc biệt trong trò chơi Blackjack:

1. **Blackjack cho nhà cái:**

Nếu lá bài lật ngửa của nhà cái là một lá 10 (10, J, Q, K) và lá bài úp của nhà cái là một lá A (tạo thành tổng điểm 21), thì nhà cái sẽ thắng ngay lập tức, gọi là "Blackjack".

2. **Blackjack cho người chơi:**

Nếu người chơi nhận được hai lá bài đầu tiên là một lá A và một lá 10 (hoặc 10, J, Q, K), thì người chơi sẽ thắng ngay lập tức mà không cần phải so điểm, gọi là "Blackjack".

3. **Bust (vượt quá 21 điểm):**

Nếu trong quá trình rút bài, tổng điểm của người chơi vượt quá 21, người chơi sẽ thua ngay lập tức (được gọi là "Bust").

4. **Kết quả hòa:**

Nếu người chơi và nhà cái có cùng tổng điểm và điểm số đó không vượt quá 21, kết quả sẽ là hòa.

1.3. Đầu vào – đầu ra của chương trình:

1. **Đầu vào:**

Người chơi sẽ tương tác với trò chơi thông qua các nút "Hit" và "Stand".

- Nút "Hit" cho phép người chơi rút thêm bài.
- Nút "Stand" cho phép người chơi dừng lại và kết thúc lượt chơi.

2. **Đầu ra:**

- Trò chơi sẽ hiển thị các lá bài đã rút dưới dạng hình ảnh hoặc icon.
- Điểm số của người chơi và nhà cái sẽ được hiển thị rõ ràng.
- Kết quả của trò chơi (thắng, thua, hòa) sẽ được hiển thị sau khi so sánh điểm của người chơi và nhà cái.

1.4. Tính năng của chương trình

- **Chia bài và rút bài**: Trò chơi sử dụng bộ bài tây 52 lá. Người chơi và nhà cái sẽ lần lượt nhận 2 lá bài, và người chơi có thể rút thêm bài bằng cách nhấn nút "Hit" hoặc dừng lại bằng nút "Stand".
- **Tính điểm**: Mỗi lá bài có một giá trị điểm cụ thể, và tổng điểm của người chơi hoặc nhà cái sẽ được tính dựa trên các lá bài hiện có.
- **Kết quả**: Trò chơi sẽ so sánh điểm giữa người chơi và nhà cái để xác định người chiến thắng. Nếu người chơi có điểm vượt quá 21, sẽ thua ngay lập tức. Nếu người chơi có điểm cao hơn nhà cái, người chơi thắng.
- **Giao diện người dùng (GUI)**: Sử dụng tkinter, giao diện sẽ hiển thị các lá bài của người chơi và nhà cái, điểm số và kết quả cuối cùng.

1.5. Thách thức trong bài tập

Một trong những khó khăn lớn nhất khi thực hiện bài tập lớn môn Python này là việc tích hợp hiệu quả giữa logic xử lý trò chơi và giao diện người dùng (GUI). Quá trình này đòi hỏi không chỉ sự khéo léo trong lập trình mà còn cần sự hiểu biết sâu sắc về cách các thành phần tương tác với nhau để tạo ra một trải nghiệm liền mạch. Cụ thể, các hành động quan trọng như chia bài, rút thêm bài, và tính điểm phải được cập nhật đồng bộ và chính xác trên giao diện theo thời gian thực. Điều này không chỉ đảm bảo rằng người chơi có thể theo dõi tiến trình trò chơi một cách dễ dàng mà còn giúp tăng cường tính hấp dẫn và thú vị, đặc biệt khi các thay đổi trên giao diện phản ánh đúng trạng thái logic của trò chơi. Việc xử lý sai sót trong bước tích hợp này có thể dẫn đến lỗi hiển thị hoặc trải nghiệm không mượt mà, làm giảm chất lượng tổng thể của sản phẩm.

Bên cạnh đó, việc quản lý trạng thái của trò chơi một cách chính xác cũng là một thách thức quan trọng cần được giải quyết kỹ lưỡng. Trong một trò chơi như Blackjack, trạng thái của trò chơi thay đổi liên tục dựa trên các quyết định của người chơi và nhà cái, đòi hỏi lập trình viên phải xây dựng một cơ chế mạnh mẽ để theo dõi và cập nhật những thay đổi này. Chẳng hạn,

cần nhận diện và phản ứng kịp thời với các tình huống đặc biệt như khi người chơi "Bust" (vượt quá 21 điểm) hoặc khi nhà cái đạt được "Blackjack" ngay từ đầu. Việc xử lý không đúng có thể dẫn đến kết quả không công bằng hoặc gây nhầm lẫn cho người chơi. Do đó, việc thiết kế một hệ thống quản lý trạng thái rõ ràng, bao gồm các điều kiện kiểm tra và phản hồi phù hợp, là yếu tố then chốt để đảm bảo tính liền mạch và công bằng trong suốt quá trình chơi, từ khi bắt đầu cho đến khi kết thúc ván bài.

Ngoài ra, xử lý các trường hợp đặc thù cũng đặt ra một yêu cầu cao trong quá trình phát triển phần mềm này. Một ví dụ điển hình là tình huống nhà cái có "Blackjack" ngay từ lá bài lật ngửa có giá trị 10 điểm, yêu cầu trò chơi phải kết thúc sớm mà không cần người chơi tiếp tục thực hiện các hành động như rút bài. Để thực hiện điều này, lập trình viên cần xây dựng một logic kiểm tra nhanh chóng và chính xác ngay từ giai đoạn đầu của ván bài, đồng thời đảm bảo rằng giao diện phản hồi ngay lập tức với kết quả này. Điều này không chỉ đòi hỏi kỹ năng lập trình mà còn cần sự hiểu biết về luật chơi để áp dụng đúng quy tắc, tránh những sai lệch không mong muốn. Việc xử lý tốt các trường hợp đặc biệt này không chỉ nâng cao tính chuyên nghiệp của trò chơi mà còn tạo nên sự bất ngờ và hứng thú cho người chơi, góp phần làm tăng giá trị trải nghiệm tổng thể.

1.6. Kiến thức vận dụng

Trong quá trình thực hiện bài tập lớn môn Python, một lượng kiến thức quan trọng đã được vận dụng một cách hiệu quả để xây dựng trò chơi Blackjack. Trước hết, lập trình hướng đối tượng (OOP) đóng vai trò cốt lõi trong việc thiết kế và tổ chức mã nguồn. Cụ thể, các lớp như BJ_Card, BJ_Deck, BJ_Hand, và BJ_Game được sử dụng để mô phỏng các thành phần chính của trò chơi, bao gồm quản lý lá bài, bộ bài, tay bài của người chơi và nhà cái, cũng như xử lý các hành động như chia bài, tính điểm, và phản hồi các quyết định của người chơi. Phương pháp này không chỉ giúp mã nguồn trở nên có cấu trúc rõ ràng mà còn dễ dàng mở rộng và bảo trì trong tương lai, đồng thời đảm bảo tính linh hoạt khi cần tích hợp thêm tính năng mới.

Bên cạnh đó, việc xử lý giao diện người dùng (GUI) là một phần không thể thiếu, được thực hiện thông qua thư viện tkinter. Thư viện này được tận

dụng để tạo ra các thành phần giao diện trực quan như các nút bấm "Hit" và "Stand", cho phép người chơi tương tác với trò chơi một cách dễ dàng. Ngoài ra, các label được sử dụng để hiển thị trực tiếp các lá bài đang nắm giữ và điểm số của cả người chơi lẫn nhà cái, giúp người dùng theo dõi tiến trình ván bài một cách tức thời. Việc thiết kế GUI đòi hỏi sự kết hợp chặt chẽ giữa giao diện và logic xử lý phía sau, đảm bảo rằng mọi thay đổi trong trạng thái trò chơi đều được phản ánh chính xác trên màn hình, từ đó nâng cao trải nghiệm người dùng.

Cuối cùng, các thuật toán trò chơi đã được áp dụng một cách hiệu quả để đảm bảo tính chính xác và công bằng trong quá trình chơi. Các thuật toán cơ bản được triển khai để tính toán điểm số dựa trên giá trị của từng lá bài, bao gồm xử lý trường hợp đặc biệt như lá bài "A" có thể được tính là 1 hoặc 11 điểm. Đồng thời, các quy tắc lượt chơi của người chơi và nhà cái, chẳng hạn như nhà cái phải rút bài khi dưới 17 điểm, cũng được lập trình một cách chi tiết. Quá trình này bao gồm cả việc xác định kết quả cuối cùng của ván bài, dựa trên sự so sánh điểm số giữa hai bên, từ đó đưa ra kết luận thắng, thua, hoặc hòa một cách minh bạch. Sự kết hợp giữa các thuật toán này không chỉ đảm bảo tính logic của trò chơi mà còn tạo nền tảng cho việc mở rộng thêm các tính năng phức tạp hơn trong tương lai.

Tóm tắt chương.

Chương này đã cung cấp cái nhìn tổng quan về các tính năng và thách thức trong việc xây dựng trò chơi Blackjack, đồng thời chỉ ra những kiến thức và kỹ năng cần thiết để hoàn thành bài tập.

CHƯƠNG 2. CƠ SỞ LÝ THUYẾT

2.1. Lập trình hướng đối tượng

Lập trình hướng đối tượng (OOP) là phương pháp lập trình dựa trên việc sử dụng các đối tượng và lớp. Mỗi đối tượng trong chương trình sẽ đại diện cho một thực thể của trò chơi. Trong trường hợp của trò chơi Blackjack, chúng ta có thể mô phỏng các thành phần như lá bài, bộ bài, tay bài của người chơi và nhà cái dưới dạng các đối tượng. Cách tiếp cận này giúp chia nhỏ các vấn đề và tổ chức mã nguồn một cách hợp lý, dễ bảo trì và mở rộng.

Cụ thể, trong trò chơi này, em sẽ sử dụng các lớp như BJ_Card, BJ_Deck, BJ_Hand và BJ_Game để mô phỏng các thành phần và hành động trong trò chơi. Lớp BJ_Card sẽ đại diện cho mỗi lá bài trong bộ bài, bao gồm các thuộc tính như giá trị và bộ bài. Lớp BJ_Deck sẽ quản lý bộ bài tây 52 lá, bao gồm các phương thức như xáo trộn bộ bài và rút bài. Lớp BJ_Hand đại diện cho tay bài của một người chơi, bao gồm các lá bài đã được chia cho người chơi hoặc nhà cái và các phương thức để tính tổng điểm của tay bài. Lớp BJ_Game là lớp chính, quản lý toàn bộ logic của trò chơi, bao gồm việc chia bài, xử lý các hành động của người chơi (hit/stand), tính điểm và xác định kết quả cuối cùng.

2.2. Cấu trúc dữ liệu

Trong trò chơi Blackjack, em sử dụng một số cấu trúc dữ liệu cơ bản để quản lý các lá bài và các tay bài của người chơi và nhà cái. Danh sách (list) trong Python sẽ là cấu trúc dữ liệu chính để lưu trữ các lá bài trong bộ bài và các tay bài của người chơi và nhà cái. Danh sách cho phép dễ dàng thêm mới các lá bài vào tay bài, cũng như rút lá bài từ bộ bài khi người chơi nhấn "Hit".

Ngoài ra, tuple sẽ được sử dụng để đại diện cho một lá bài, bao gồm giá trị và bộ bài (ví dụ: ("A", "♠")). Tuple là cấu trúc dữ liệu bất biến, giúp đảm bảo tính ổn định khi lưu trữ thông tin về các lá bài. Một cấu trúc dữ liệu khác có thể sử dụng là từ điển (dictionary), ví dụ như để lưu trữ điểm của các lá bài, như {"2": 2, "3": 3, ..., "A": 11}, giúp việc tra cứu giá trị điểm của mỗi lá bài trở nên dễ dàng và hiệu quả hơn.

2.3. Thuật toán xử lý trò chơi

Các thuật toán trong trò chơi Blackjack sẽ xử lý các bước quan trọng như tính điểm, xử lý các tình huống đặc biệt (Blackjack, Bust) và so sánh điểm giữa người chơi và nhà cái. Thuật toán tính điểm sẽ cộng tổng giá trị các lá bài trong tay người chơi và nhà cái, với quy tắc đặc biệt cho lá A (có thể tính là 1 hoặc 11 điểm tùy vào tình huống). Nếu tổng điểm của người chơi hoặc nhà cái vượt quá 21, thuật toán sẽ xác định tình huống "Bust" và người chơi hoặc nhà cái sẽ thua ngay lập tức.

Thuật toán so sánh điểm sẽ được thực hiện sau khi người chơi chọn "Stand". Nếu tổng điểm của người chơi cao hơn nhà cái mà không vượt quá 21, người chơi thắng. Nếu điểm của nhà cái cao hơn người chơi, nhà cái thắng. Nếu điểm của người chơi và nhà cái bằng nhau, kết quả sẽ là hòa. Thuật toán xử lý Blackjack sẽ kiểm tra nếu người chơi hoặc nhà cái có hai lá bài đầu tiên là A và 10 (hoặc 10, J, Q, K), đó gọi là "Blackjack" và người đó sẽ thắng ngay lập tức.

2.4. Giao diện người dùng (GUI)

Giao diện người dùng trong trò chơi Blackjack sẽ được xây dựng bằng thư viện tkinter, một thư viện GUI phổ biến trong Python. Các thành phần chính của giao diện bao gồm nút bấm "Hit" và "Stand", dùng để người chơi tương tác với trò chơi. Khi người chơi nhấn nút "Hit", lá bài mới sẽ được rút và hiển thị trên giao diện. Khi nhấn nút "Stand", lượt chơi của người chơi sẽ kết thúc và nhà cái sẽ bắt đầu hành động.

Ngoài các nút bấm, giao diện cũng sẽ hiển thị các lá bài của người chơi và nhà cái dưới dạng hình ảnh hoặc icon. Các label sẽ được sử dụng để hiển thị điểm số hiện tại của người chơi và nhà cái, giúp người chơi dễ dàng theo dõi quá trình chơi. Sau khi ván bài kết thúc, kết quả (thắng, thua, hòa) sẽ được hiển thị thông qua một messagebox, cho phép người chơi biết kết quả của ván bài.

2.5. Các quy tắc trong trò chơi Blackjack

Trong trò chơi Blackjack, có một số quy tắc quan trọng cần lưu ý. Một trong những quy tắc chính là việc tính điểm của các lá bài. Các lá bài từ 2 đến

9 có điểm tương ứng với giá trị trên lá bài, các lá bài 10, J, Q, K đều có giá trị 10 điểm, và lá A có thể tính là 1 hoặc 11 điểm tùy vào tình huống. Nếu tổng điểm vượt quá 21, người chơi hoặc nhà cái sẽ bị "Bust" và thua ngay lập tức.

Một quy tắc khác là khi nhà cái có lá bài lật ngửa là 10 điểm, nhà cái sẽ kiểm tra lá bài úp của mình. Nếu tổng điểm của nhà cái là 21, nhà cái sẽ thắng ngay lập tức, gọi là "Blackjack". Còn nếu người chơi có hai lá bài đầu tiên là A và 10 (hoặc 10, J, Q, K), người chơi sẽ thắng ngay lập tức mà không cần phải rút thêm bài.

Tóm tắt chương 2:

Chương 2 đã trình bày những kiến thức lý thuyết cần thiết để xây dựng trò chơi Blackjack, bao gồm lập trình hướng đối tượng, cấu trúc dữ liệu, thuật toán xử lý trò chơi, giao diện người dùng và các quy tắc cơ bản của trò chơi. Những kiến thức này là nền tảng vững chắc để em có thể triển khai trò chơi Blackjack một cách hiệu quả và dễ dàng quản lý các thành phần trong trò chơi.

CHƯƠNG III: THIẾT KẾ VÀ XÂY DỰNG CHƯƠNG TRÌNH

3.1. Sơ đồ khối hệ thống

3.1.1 Các mô-đun chính

Hệ thống trò chơi Blackjack được thiết kế theo mô hình mô-đun hóa, trong đó mỗi mô-đun đảm nhận một nhiệm vụ cụ thể nhằm tối ưu hóa việc điều khiển logic và giao diện của trò chơi, đảm bảo tính mạch lạc và dễ dàng bảo trì. Cụ thể, hệ thống được chia thành các mô-đun chính như sau:

Mô-đun BJ_Card đóng vai trò mô phỏng một lá bài trong bộ bài Blackjack, là thành phần cơ bản nhất của trò chơi. Mỗi lá bài được định nghĩa bởi hai thuộc tính chính: rank (giá trị của lá bài, chẳng hạn như 2, 3, 4, ..., A) và suit (bộ của lá bài, bao gồm Hearts, Spades, Diamonds, và Clubs). Để tính điểm, phương thức _get_value() được triển khai trong lớp này, quy định rằng các lá J, Q, K có giá trị 10 điểm, trong khi lá A có thể linh hoạt được tính là 1 hoặc 11 điểm tùy thuộc vào tình huống của tay bài, giúp xử lý các trường hợp đặc biệt một cách thông minh và chính xác.

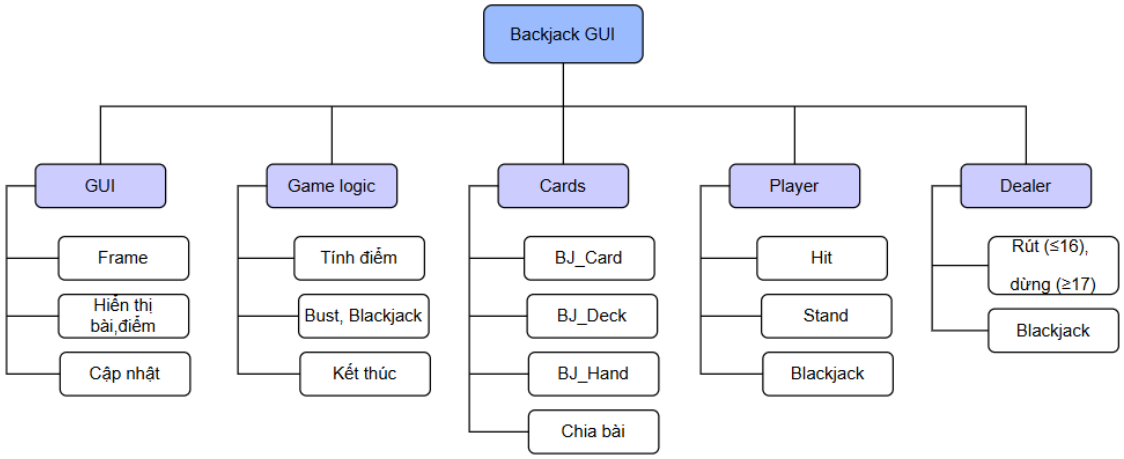
Mô-đun BJ_Deck được thiết kế để mô phỏng một bộ bài tiêu chuẩn gồm 52 lá, là nguồn cung cấp lá bài cho toàn bộ trò chơi. Bộ bài được khởi tạo bằng cách lặp qua tất cả các tổ hợp của rank và suit, sau đó được xáo trộn ngẫu nhiên để đảm bảo tính bất ngờ trong mỗi ván bài. Phương thức deal() của lớp này cho phép trả về một lá bài từ bộ bài mỗi khi có yêu cầu, đảm bảo rằng quá trình chia bài diễn ra suôn sẻ và không trùng lặp.

Mô-đun BJ_Hand chịu trách nhiệm quản lý tay bài của cả người chơi và nhà cái, đóng vai trò trung gian trong việc theo dõi các lá bài đang sở hữu. Mỗi tay bài được biểu diễn dưới dạng một danh sách các lá bài, với phương thức add_card() để thêm lá bài mới và phương thức get_value() để tính tổng điểm của tay bài. Đặc biệt, lớp này được thiết kế để xử lý trường hợp phức tạp với lá A, tự động điều chỉnh giá trị (1 hoặc 11 điểm) sao cho tổng điểm không vượt quá 21, từ đó đảm bảo tính logic và công bằng trong quá trình tính toán.

Mô-đun BJ_Game là trung tâm điều khiển toàn bộ quá trình chơi, đóng vai trò kết nối các mô-đun khác để tạo ra trải nghiệm trò chơi hoàn chỉnh. Mô-đun này bao gồm các phương thức quan trọng như chia bài ban đầu, xử lý các hành động "Hit" (rút thêm bài) và "Stand" (dừng rút) của người chơi, tính toán và so sánh điểm số giữa người chơi và nhà cái, cũng như hiển thị kết quả cuối cùng của ván bài. Nhờ đó, mô-đun này đảm bảo rằng logic trò chơi được thực thi chính xác và phản ánh đúng quy tắc Blackjack.

Mô-đun GUI được xây dựng dựa trên thư viện tkinter, đóng góp vào việc tạo ra một giao diện người dùng trực quan và thân thiện. Giao diện này bao gồm các frame để hiển thị tay bài của người chơi và nhà cái một cách rõ ràng, các nút bấm "Hit", "Stand", và "New Game" cho phép người chơi tương tác dễ dàng, cùng với các label để thể hiện điểm số hiện tại và thông báo kết quả của trò chơi. Sự kết hợp giữa GUI và logic phía sau đảm bảo rằng mọi thay đổi trong trạng thái trò chơi đều được phản ánh tức thời trên màn hình, nâng cao trải nghiệm người dùng.

3.1.2 Biểu đồ phân cấp chức năng



Hình 3 Biểu đồ phân cấp chức năng

1. **GUI (Giao diện người dùng)**
    - **Chức năng**: Quản lý giao diện đồ họa.
    - **Chi tiết**:
        - Hiển thị lá bài (Label/Icon), điểm số, kết quả (thắng/thua/hòa).
        - Tạo Frame riêng cho người chơi và nhà cái.
        - Cập nhật giao diện khi rút bài hoặc kết thúc ván.
        - Cung cấp nút "Hit" và "Stand" để người chơi thao tác.
2. **Logic (Logic trò chơi)**
    - **Chức năng**: Xử lý luật chơi.
    - **Chi tiết**:
        - Tính điểm: 2-9 theo số, 10/J/Q/K = 10, A = 1/11.
        - Kiểm tra Blackjack (A + 10) và bust (>21).
        - So sánh điểm để xác định thắng, thua, hoặc hòa.
3. **Cards (Quản lý bài)**
    - **Chức năng**: Quản lý và chia bài.
    - **Chi tiết**:
        - **BJ_Card**: Lá bài với chất và giá trị.
        - **BJ_Deck**: Bộ 52 lá, xáo và rút.
        - **BJ_Hand**: Tay bài, tính điểm.
        - Chia: 2 lá lật ngửa (người chơi), 1 lá lật + 1 úp (nhà cái).
4. **Player (Hành động người chơi)**
    - **Chức năng**: Thực hiện hành động người chơi.
    - **Chi tiết**:
        - **Hit**: Rút thêm lá bài, cập nhật điểm.
        - **Stand**: Dừng rút, chuyển lượt sang nhà cái.

- Kiểm tra Blackjack (A + 10) để thắng ngay.

5. **Dealer (Hành động nhà cái)**
    - **Chức năng**: Thực hiện hành động nhà cái.
    - **Chi tiết**:
        - **Rút**: Rút bài nếu điểm ≤16.
        - **Dừng**: Dừng nếu điểm ≥17.
        - Kiểm tra Blackjack nếu lá lật ngửa = 10.

Tương tác:

- GUI hiển thị và nhận lệnh từ Player (Hit/Stand).
- Cards cung cấp bài cho Player và Dealer.
- Logic tính toán kết quả và gửi lại GUI để cập nhật.

3.2. Sơ đồ khối các thuật toán chính

Các thuật toán chính trong trò chơi Blackjack sẽ bao gồm:

1. Thuật toán Khởi tạo ván mới (start_new_game)

- **Đầu vào**: Đối tượng BJ_Game.
- **Quy trình**:
    1. Tạo mới BJ_Deck và xóa sạch player_hand và dealer_hand.
    2. Xóa các label bài cũ trên giao diện.
    3. Chia 2 lá bài cho player_hand và 2 lá cho dealer_hand bằng deck.deal().
    4. Kiểm tra điều kiện đặc biệt:
        - Nếu dealer_hand.get_value() == 21 (với lá lên là 10 hoặc A), hiển thị "Dealer Blackjack! You lose!" và kết thúc.
        - Nếu player_hand.get_value() == 21, hiển thị "Blackjack! You win!" và kết thúc.
    5. Cập nhật giao diện bằng update_gui().
- **Đầu ra**: Trạng thái ván bài mới với lá bài hiển thị.

2. Thuật toán Xử lý "Hit" (hit)

- **Đầu vào**: Yêu cầu rút bài từ người chơi.
- **Quy trình**:
    1. Phát âm thanh hit.
    2. Thêm lá bài mới vào player_hand từ deck.deal().
    3. Cập nhật giao diện bằng update_gui().
    4. Kiểm tra:
        - Nếu player_hand.get_value() > 21, hiển thị "Bust! You lose!", vô hiệu hóa nút "Hit" và "Stand", kết thúc lượt.
- **Đầu ra**: Cập nhật tay bài và trạng thái (thắng/thua nếu Bust).

3. Thuật toán Xử lý "Stand" (stand)

- **Đầu vào**: Yêu cầu dừng từ người chơi.
- **Quy trình**:
    1. Vô hiệu hóa nút "Hit" và "Stand".
    2. Lượt nhà cái:
        - Lặp rút bài từ deck.deal() cho đến khi dealer_hand.get_value() >= 17.
    3. Cập nhật giao diện bằng update_gui().
    4. So sánh điểm:
        - Nếu dealer_score > 21, hiển thị "Dealer busts! You win!".
        - Nếu player_score > dealer_score, hiển thị "You win!".
        - Nếu dealer_score > player_score, hiển thị "Dealer wins!".
        - Nếu bằng nhau, hiển thị "It's a tie!".
    5. Gọi show_result() để hiển thị kết quả và phát âm thanh.
- **Đầu ra**: Kết quả ván bài (thắng, thua, hoặc hòa).

4. Thuật toán Tính điểm (get_value trong BJ_Hand)

- **Đầu vào**: Danh sách cards trong tay.
- **Quy trình**:
    1. Khởi tạo value = 0 và đếm số lá A (aces = 0).
    2. Lặp qua từng lá bài, cộng value với card.value và tăng aces nếu là A.

3. Nếu value > 21 và aces > 0, giảm value đi 10 (tính A là 1) và giảm aces đi 1, lặp lại đến khi value <= 21 hoặc hết A.

- **Đầu ra**: Tổng điểm của tay bài.

5. Thuật toán Cập nhật giao diện (update_gui)

- **Đầu vào**: Trạng thái tay bài của người chơi và nhà cái.
- **Quy trình**:
  1. Xóa các label bài cũ khỏi giao diện.
  2. Cập nhật tay bài người chơi:
     - Duyệt player_hand.cards, tải hình ảnh bằng load_card_image(), và hiển thị trên player_cards_frame.
  3. Cập nhật tay bài nhà cái:
     - Nếu hit_button còn hoạt động, hiển thị lá đầu tiên và lá bài úp (load_back_image()).
     - Nếu không, hiển thị toàn bộ lá bài.
  4. Cập nhật label điểm số cho người chơi và nhà cái.
- **Đầu ra**: Giao diện phản ánh trạng thái hiện tại của trò chơi.

3.3. Cấu trúc dữ liệu

Trong chương trình thực hiện trò chơi Blackjack, các cấu trúc dữ liệu được thiết kế một cách cẩn thận để hỗ trợ hiệu quả cho việc quản lý và xử lý logic của trò chơi. Các bảng dữ liệu chính bao gồm bộ bài, tay bài của người chơi và nhà cái, cùng với các giá trị điểm liên quan, được tổ chức và vận hành thông qua các lớp và phương thức cụ thể. Cụ thể:

Bộ bài (Deck) là một trong những thành phần cốt lõi, được biểu diễn dưới dạng một danh sách self.cards chứa 52 lá bài tiêu chuẩn. Mỗi lá bài là một đối tượng thuộc lớp BJ_Card, mang các thuộc tính quan trọng như suit (bộ của lá bài, chẳng hạn như Hearts, Spades, Diamonds, hoặc Clubs) và rank (giá trị của lá bài, từ 2 đến 10, hoặc J, Q, K, A). Quá trình quản lý bộ bài được thực hiện thông qua lớp BJ_Deck, trong đó phương thức deal() cho phép rút một lá bài từ danh sách này mỗi khi có yêu cầu, đảm bảo rằng quá trình chia bài diễn ra chính xác và không trùng lặp, đồng thời hỗ trợ việc xáo trộn ngẫu nhiên để tăng tính bất ngờ trong trò chơi.

Tay bài (Hand) được thiết kế để quản lý các lá bài mà người chơi và nhà cái đang sở hữu, được lưu trữ dưới dạng một đối tượng của lớp BJ_Hand. Mỗi tay bài chứa một danh sách các lá bài đã được chia, và phương thức get_value() được triển khai để tính toán tổng điểm của tay bài dựa trên giá trị của từng lá. Đặc biệt, lớp này xử lý linh hoạt trường hợp có lá A trong tay bài: nếu tổng điểm vượt quá 21 và có ít nhất một lá A, giá trị của lá A sẽ được điều chỉnh từ 11 xuống 1 để tránh tình trạng "Bust", từ đó duy trì tính logic và công bằng trong việc tính điểm, phù hợp với quy tắc của Blackjack.

Âm thanh là một yếu tố bổ sung nhằm tăng tính sinh động và hấp dẫn cho trò chơi, được quản lý thông qua thư viện pygame. Các hiệu ứng âm thanh được thiết kế cho các hành động quan trọng như "Hit" (rút thêm bài), "Win", "Lose", và "Tie", được lưu trữ dưới dạng các tệp âm thanh và phát trong các tình huống tương ứng. Sự tích hợp này không chỉ làm phong phú thêm trải nghiệm người dùng mà còn tạo ra một không khí chân thực, giúp người chơi cảm nhận rõ hơn các khoảnh khắc quan trọng trong ván bài, từ đó nâng cao giá trị giải trí của chương trình.

3.4. Chương trình

Chương trình Blackjack gồm các hàm chính sau:

1. deal_cards(): Hàm này chịu trách nhiệm chia bài cho người chơi và nhà cái, bao gồm việc rút hai lá bài cho mỗi bên từ bộ bài đã xáo trộn.
2. hit(): Hàm này sẽ rút thêm một lá bài cho người chơi. Nếu tổng điểm của người chơi vượt quá 21, trò chơi sẽ kết thúc và người chơi thua.
3. stand(): Khi người chơi nhấn "Stand", lượt chơi của người chơi sẽ kết thúc, và nhà cái sẽ bắt đầu mở bài và rút thêm nếu cần.
4. calculate_score(): Hàm này tính tổng điểm của tay bài của người chơi hoặc nhà cái, giúp xác định kết quả của trò chơi.
5. show_result(): Sau mỗi ván bài, hàm này hiển thị kết quả trò chơi (win/lose/tie) thông qua một messagebox và phát âm thanh tương ứng.
6. update_gui(): Cập nhật giao diện người dùng bằng cách hiển thị các lá bài của người chơi và nhà cái, cũng như điểm số và kết quả cuối cùng.

Tóm tắt chương 3:

Chương 3 đã trình bày chi tiết quá trình thiết kế và xây dựng chương trình trò chơi Blackjack. Việc chia nhỏ các chức năng thành các module riêng biệt, kết hợp với các thuật toán xử lý logic trò chơi, giúp cho chương trình hoạt động mượt mà và dễ bảo trì. Các thành phần giao diện người dùng được xây dựng rõ ràng và dễ sử dụng, mang đến cho người chơi trải nghiệm trực quan và thú vị.

CHƯƠNG IV. THỰC NGHIỆM VÀ KẾT LUẬN

4.1. Thực nghiệm

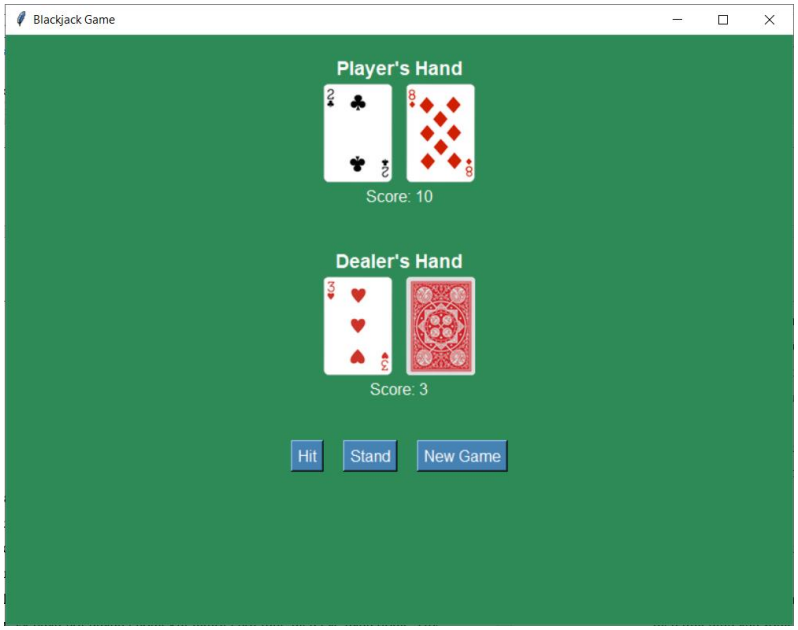
Trong quá trình phát triển trò chơi Blackjack, em đã thực hiện các bài test để kiểm tra và đảm bảo các tính năng của sản phẩm hoạt động chính xác. Các bài test chủ yếu bao gồm việc kiểm tra tính năng chia bài, rút bài (Hit), dừng bài (Stand), tính điểm, xác định kết quả trò chơi (win/lose/tie), và các tình huống đặc biệt như Blackjack và Bust.

Các bài test tính năng

1. Chia bài: Khi bắt đầu mỗi ván bài, chương trình phải chia hai lá bài cho người chơi và nhà cái. Em đã kiểm tra và đảm bảo rằng bộ bài được xáo trộn ngẫu nhiên và chia bài chính xác cho cả hai bên.
2. Tính điểm: Sau mỗi lần rút bài hoặc dừng lại, tổng điểm của người chơi và nhà cái sẽ được tính và hiển thị trên giao diện. Em đã kiểm tra các tình huống có lá A (tính là 1 hoặc 11 điểm) và đảm bảo tính toán điểm đúng trong các trường hợp này.
3. Xử lý Blackjack và Bust: Em đã kiểm tra các tình huống đặc biệt như người chơi hoặc nhà cái có Blackjack (21 điểm với hai lá bài đầu tiên) hoặc Bust (tổng điểm vượt quá 21). Các kết quả này đã được xử lý đúng, và kết quả sẽ được hiển thị ngay lập tức.
4. Kết quả trò chơi: Sau khi người chơi chọn "Stand", nhà cái sẽ mở bài và rút thêm nếu tổng điểm dưới 17. Sau đó, điểm của người chơi và nhà cái được so sánh để xác định người thắng cuộc. Em đã kiểm tra và đảm bảo rằng kết quả được tính đúng.
5. Giao diện người dùng: Các nút "Hit" và "Stand" hoạt động chính xác, và khi người chơi thực hiện các hành động, giao diện sẽ được cập nhật ngay lập tức để phản ánh kết quả (thẻ bài, điểm số, kết quả).

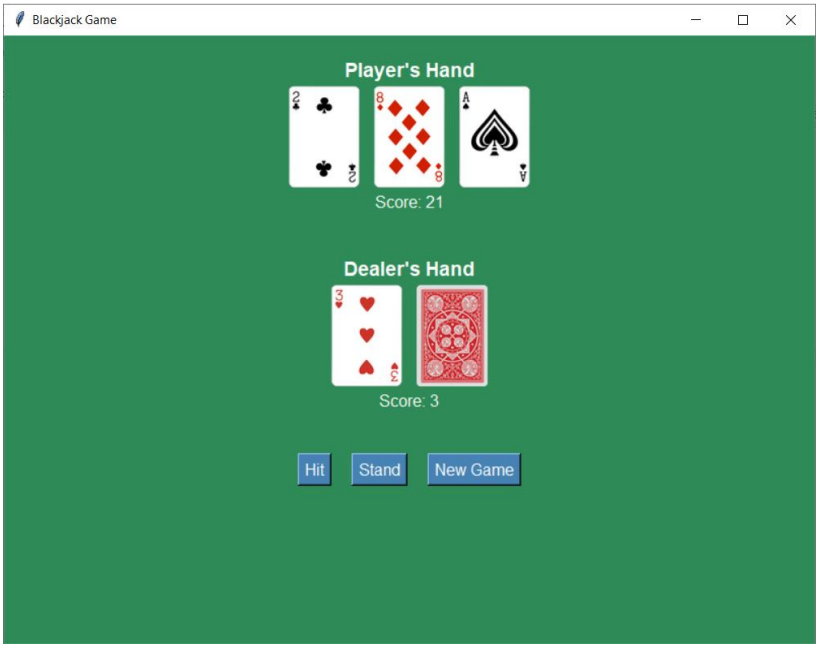
Em đã chụp lại các màn hình mô tả các tính năng của trò chơi như sau:

- Màn hình khi bắt đầu ván bài, hiển thị lá bài của người chơi và nhà cái.



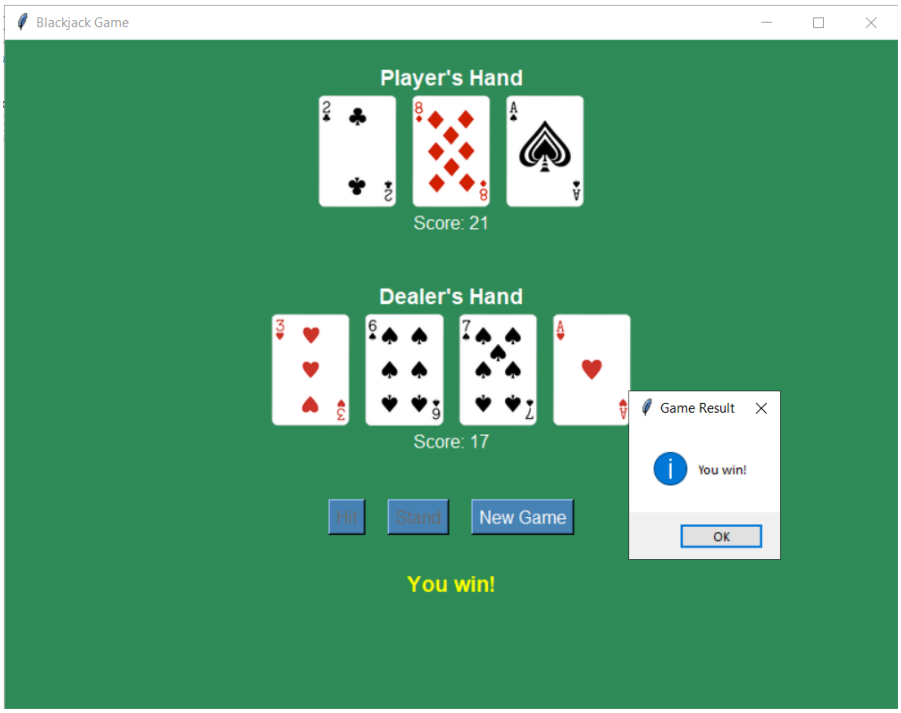
Hình 4 Màn hình khi bắt đầu ván bài

- Màn hình khi người chơi nhấn "Hit", lá bài mới được rút và hiển thị trong giao diện.



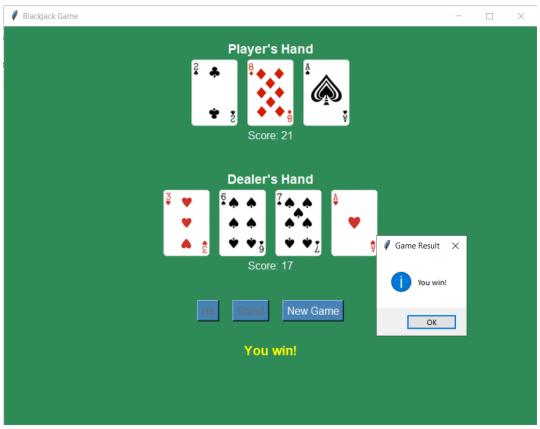
Hình 5 Màn hình khi người chơi nhấn "Hit"

- Màn hình khi người chơi nhấn "Stand", nhà cái mở bài và kết quả được so sánh.

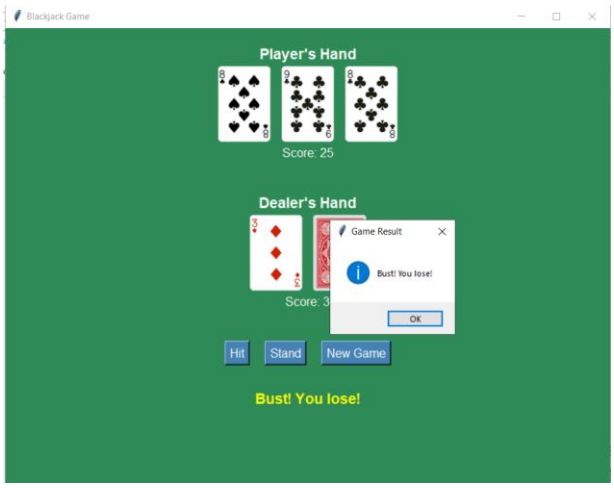


Hình 6 Màn hình khi người chơi nhấn "Stand"

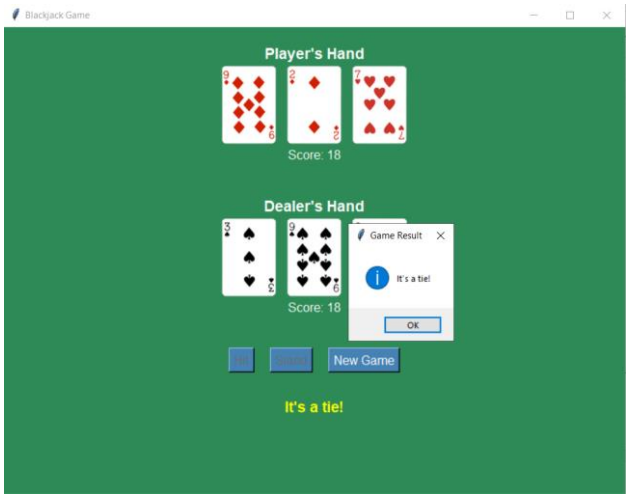
- Màn hình hiển thị kết quả trò chơi (win/lose/tie) sau mỗi ván bài.



Hình 7 Màn hình hiển thị kết quả trò chơi win



Hình 8 Màn hình hiển thị kết quả trò chơi lose



Hình 9 Màn hình hiển thị kết quả trò chơi tie

Đánh giá chất lượng

Chương trình đã hoạt động ổn định trong suốt quá trình kiểm thử. Tất cả các tính năng cơ bản của trò chơi như chia bài, tính điểm, xử lý Blackjack và Bust đều hoạt động chính xác. Giao diện người dùng được thiết kế rõ ràng, dễ sử dụng, và phản hồi nhanh chóng khi người chơi thực hiện các hành động. Tuy nhiên, vẫn có thể cải thiện một số yếu tố như tốc độ xử lý khi rút nhiều lá bài hoặc mở bài của nhà cái.

4.2. Kết luận

Sản phẩm cuối cùng của bài tập lớn môn Python đã hoàn thành một phiên bản trò chơi Blackjack với đầy đủ các tính năng cơ bản, đáp ứng tốt các yêu cầu ban đầu. Cụ thể, hệ thống được thiết kế để thực hiện chia bài ngẫu

nhiên cho cả người chơi và nhà cái, đảm bảo tính công bằng và bất ngờ trong mỗi ván bài. Quá trình tính điểm được thực hiện một cách chính xác, bao gồm việc xử lý các trường hợp đặc biệt như "Blackjack" (tổng điểm 21 ngay từ đầu) và "Bust" (vượt quá 21 điểm), giúp phản ánh đúng quy tắc của trò chơi. Kết quả ván bài được xác định thông qua việc so sánh điểm số giữa người chơi và nhà cái, sau đó hiển thị rõ ràng trên giao diện. Về mặt giao diện người dùng, hệ thống được xây dựng với thiết kế đơn giản nhưng hiệu quả, tích hợp các nút bấm "Hit", "Stand", và "New Game" để hỗ trợ tương tác, cùng với các label hiển thị điểm số và thông báo kết quả, tạo điều kiện thuận lợi cho người chơi theo dõi tiến trình.

Qua quá trình xây dựng trò chơi này, em đã tích lũy được nhiều kinh nghiệm quý báu trong lĩnh vực lập trình Python, đặc biệt là việc áp dụng thư viện tkinter để thiết kế và phát triển giao diện người dùng một cách trực quan và thân thiện. Đồng thời, em đã nắm vững cách sử dụng lập trình hướng đối tượng (OOP) thông qua việc tạo ra các lớp như BJ_Card, BJ_Deck, và BJ_Hand để mô phỏng các thành phần cốt lõi của trò chơi, từ đó nâng cao khả năng tổ chức và quản lý mã nguồn. Ngoài ra, việc tích hợp các yếu tố đa phương tiện như âm thanh (sử dụng thư viện pygame) và hình ảnh (sử dụng Pillow) đã giúp em cải thiện đáng kể kỹ năng xử lý các tài nguyên đa phương tiện trong Python, đồng thời làm phong phú thêm trải nghiệm người dùng.

Bên cạnh những kỹ năng kỹ thuật, em cũng học được cách quản lý bộ bài một cách hiệu quả, bao gồm việc xáo trộn và rút bài, cũng như cách tính toán điểm số một cách logic, đặc biệt là trong các tình huống phức tạp như xử lý lá bài "A". Em đã rèn luyện khả năng xử lý các tình huống đặc biệt trong trò chơi, chẳng hạn như khi nhà cái có "Blackjack" ngay từ đầu hoặc khi người chơi "Bust". Hơn nữa, quá trình kiểm tra và đảm bảo chất lượng sản phẩm thông qua các bài test đã giúp em hiểu rõ hơn về tầm quan trọng của việc debug và tối ưu hóa mã nguồn, từ đó nâng cao chất lượng sản phẩm cuối cùng.

Định hướng cải tiến

Dù trò chơi đã đạt được các tính năng cơ bản và hoạt động ổn định, em nhận thấy vẫn còn nhiều khía cạnh có thể được cải thiện để nâng cao chất

lượng và trải nghiệm người dùng. Thứ nhất, về hiệu ứng đồ họa và âm thanh, giao diện hiện tại có thể được nâng cấp với các hiệu ứng hình ảnh sống động hơn, chẳng hạn như thay đổi nền (background) khi trò chơi kết thúc để tạo cảm giác mới mẻ, hoặc thêm các hiệu ứng chuyển động mượt mà khi rút bài, nhằm tăng tính hấp dẫn và chuyên nghiệp. Thứ hai, tốc độ xử lý của chương trình có thể được tối ưu hóa, đặc biệt trong các trường hợp người chơi thực hiện nhiều hành động "Hit" liên tiếp hoặc khi nhà cái mở toàn bộ bài. Điều này có thể đạt được bằng cách tối ưu hóa mã nguồn, loại bỏ các đoạn mã dư thừa, hoặc áp dụng các kỹ thuật xử lý đa luồng để cải thiện hiệu suất.

Ngoài ra, một hướng cải tiến quan trọng là mở rộng tính năng để hỗ trợ chế độ đa người chơi, cho phép nhiều người tham gia cùng lúc trong một ván bài thay vì chỉ giới hạn ở cuộc đối đầu giữa người chơi và nhà cái. Điều này không chỉ làm tăng tính tương tác mà còn mở ra cơ hội phát triển trò chơi theo hướng cộng đồng. Cuối cùng, em dự định bổ sung tính năng lưu điểm và thống kê các ván bài đã chơi, cho phép người chơi theo dõi kết quả, lịch sử thắng thua, và các thành tích cá nhân. Những cải tiến này không chỉ nâng cao giá trị giải trí mà còn tạo tiền đề để phát triển sản phẩm thành một phiên bản hoàn thiện hơn trong tương lai.

KẾT LUẬN

Bài tập lớn này đã giúp em có cái nhìn tổng thể về quá trình phát triển một ứng dụng hoàn chỉnh, từ việc thiết kế trò chơi Blackjack cho đến việc triển khai giao diện người dùng và xử lý các sự kiện trong trò chơi. Sử dụng Python và các thư viện như tkinter và pygame, em đã xây dựng một trò chơi Blackjack với đầy đủ các tính năng cơ bản như chia bài, tính điểm, xử lý các tình huống đặc biệt như Blackjack và Bust, và cuối cùng là xác định kết quả giữa người chơi và nhà cái.

Quá trình phát triển không chỉ giúp em củng cố kiến thức lập trình hướng đối tượng (OOP) mà còn rèn luyện kỹ năng sử dụng các thư viện hỗ trợ để xây dựng giao diện người dùng và tích hợp âm thanh vào ứng dụng. Việc áp dụng các thuật toán tính toán điểm, xử lý sự kiện và quản lý bộ bài cũng đã giúp em hiểu sâu hơn về cách tổ chức và xử lý dữ liệu trong một ứng dụng game.

Trong quá trình thực nghiệm, em đã kiểm tra và đảm bảo tất cả các tính năng hoạt động chính xác, bao gồm việc chia bài, tính điểm, xử lý các tình huống đặc biệt, và hiển thị kết quả. Giao diện người dùng đơn giản nhưng rõ ràng đã giúp người chơi dễ dàng tương tác với trò chơi.

Dù đã hoàn thành các tính năng cơ bản, em nhận thấy vẫn còn một số điểm có thể cải tiến như nâng cao hiệu ứng đồ họa, tối ưu tốc độ xử lý và mở rộng tính năng để hỗ trợ chế độ chơi nhiều người. Những cải tiến này sẽ giúp trò chơi trở nên sinh động hơn và mang lại trải nghiệm người dùng tốt hơn trong tương lai.

Cuối cùng, bài tập lớn này không chỉ giúp em ứng dụng các kiến thức lập trình vào thực tế mà còn là bước đệm quan trọng để em phát triển các ứng dụng phức tạp hơn trong lĩnh vực phần mềm. Em tin rằng những kỹ năng và kinh nghiệm thu được từ bài tập này sẽ hỗ trợ em rất nhiều trong các dự án phần mềm sau này.

TÀI LIỆU THAM KHẢO

1. Python® Programming for the Absolute Beginner, Third Edition. Truy cập:tại đây
2. Blackjack là gì? Hướng dẫn cách chơi bài Blackjack. Truy cập: tại đây
3. Giới thiệu về blackjack và luật chơi cơ bản. Truy cập:tại đây
4. Luật chơi BLACKJACK. Truy cập:tại đây
5. Hình ảnh quân bài. Truy cập: tại đây