TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH

KHOA CÔNG NGHỆ THÔNG TIN

MÔN HỌC LẬP TRÌNH PYTHON

BÁO CÁO ĐỒ ÁN CUỐI KÌ

LẬP TRÌNH TRÒ CHƠI THÁP HÀ NỘI
BẰNG NGÔN NGỮ LẬP TRÌNH PYTHON

GVHD:	**PHAN THỊ THỂ**
NHÓM SVTH:	**NHÓM 18**
MSSV	**HỌ VÀ TÊN**
23162021	**HUỲNH THIÊN HẠO**
23162022	**TRƯƠNG NGUYỄN MINH HẬU**
23162027	**VÕ GIA HUÂN**
23162028	**CAO ĐĂNG HUY**

TP. HỒ CHÍ MINH - NĂM 2024

BÁO CÁO PHÂN CÔNG NHIỆM VỤ

STT	MSSV	Họ và tên	Nhiệm vụ	Hoàn thành
1	23162021	Huỳnh Thiên Hạo	Viết code, tổng hợp source code, chỉnh sửa phát triển tính năng, thuyết trình.	100%
2	23162022	Trương Nguyễn Minh Hậu	Viết báo cáo, làm bài thuyết trình, viết code, đóng góp ý kiến phát triển.	100%
3	23162027	Võ Gia Huân	Viết báo cáo, làm bài thuyết trình, viết code, đóng góp ý kiến phát triển.	100%
4	23162028	Cao Đăng Huy	Code, sửa lỗi, đóng góp ý kiến phát triển, thêm tính năng âm thanh, giao diện.	100%

Ghi chú:

- Tỉ lệ % = 100% : Mức độ phần trăm của từng sinh viên tham gia.

Nhận xét của giáo viên: .....

.....

.....

.....

.....

.....

Ngày tháng năm 2024

LỜI CẢM ƠN

Trước hết, em xin gửi lời cảm ơn chân thành và sâu sắc nhất đến Cô Phan Thị Thể vì đã tận tình hướng dẫn, chỉ bảo nhóm chúng em trong suốt quá trình thực hiện đồ án. Nhờ sự chỉ dẫn quý báu và sự động viên của cô, đã tiếp thêm động lực cho chúng em kịp thời hoàn thành được đồ án này.

Em cũng xin gửi lời cảm ơn đến Ban giám hiệu và các thầy cô trong Khoa Công Nghệ Thông Tin đã tạo điều kiện và cung cấp nguồn tài liệu, kiến thức cần thiết để nhóm em có thể thực hiện đồ án một cách thuận lợi, suôn sẻ.

Em xin cảm ơn các bạn bè trong lớp đã hỗ trợ, chia sẻ kinh nghiệm và giúp đỡ nhóm em trong quá trình nghiên cứu và hoàn thiện đồ án kết thúc môn học.

MỤC LỤC

Contents

PHẦN MỞ ĐẦU

1. Lý do chọn đề tài

Trò chơi Tháp Hà Nội là một trò chơi toán học cổ điển, nổi tiếng với tính thử thách về tư duy logic và khả năng giải quyết vấn đề. Người chơi phải di chuyển các đĩa từ cột này sang cột khác, tuân theo quy tắc rằng đĩa trên phải nhỏ hơn đĩa dưới. Đặc biệt, trò chơi này không chỉ giúp rèn luyện khả năng tư duy logic mà còn là một cơ hội tuyệt vời để áp dụng thuật toán đệ quy trong lập trình, giúp người lập trình phát triển kỹ năng giải quyết các bài toán phức tạp qua việc chia nhỏ bài toán thành các phần đơn giản hơn.

Việc xây dựng trò chơi Tháp Hà Nội bằng Python sẽ giúp người học hiểu rõ hơn về cách tối ưu hóa các thuật toán, đồng thời nâng cao khả năng quản lý tài nguyên bộ nhớ và cải thiện hiệu suất xử lý. Hơn nữa, khi kết hợp với thư viện Pygame để phát triển đồ họa, người lập trình sẽ không chỉ làm quen với các nguyên lý cơ bản của lập trình đồ họa như vẽ hình, xử lý sự kiện người dùng, mà còn học được cách tạo hiệu ứng chuyển động và xây dựng một giao diện tương tác hấp dẫn. Điều này không chỉ giúp tạo ra những sản phẩm trực quan, dễ tiếp cận mà còn là một công cụ tuyệt vời để người chơi trải nghiệm và phát triển kỹ năng giải quyết vấn đề.

2. Mục đích nghiên cứu

Trò chơi Tháp Hà Nội không chỉ là một thử thách tư duy logic mà còn là một công cụ học tập tuyệt vời để phát triển và ứng dụng các kỹ thuật lập trình. Mục tiêu chính của nghiên cứu này bao gồm:

Áp dụng thuật toán đệ quy: Nâng cao hiểu biết về thuật toán đệ quy thông qua việc giải quyết bài toán Tháp Hà Nội. Đây là cơ hội để hiểu rõ cách chia nhỏ bài toán lớn thành các phần nhỏ hơn, tối ưu hóa hiệu suất xử lý và sử dụng tài nguyên bộ nhớ hợp lý.

Phát triển kỹ năng lập trình đồ họa: Sử dụng thư viện Pygame để thiết kế và triển khai giao diện người dùng trực quan. Điều này giúp rèn luyện khả năng quản lý giao diện đồ họa, xử lý sự kiện, và tạo hiệu ứng tương tác hấp dẫn.

Xây dựng tư duy giải quyết vấn đề: Thông qua việc phân tích yêu cầu, thiết kế và kiểm thử trò chơi, nhóm nghiên cứu phát triển tư duy phân tích, kỹ năng giải quyết vấn đề và khả năng làm việc nhóm.

Tạo sản phẩm thực tế: Kết quả nghiên cứu là một trò chơi hoàn chỉnh, trực quan, dễ sử dụng, có thể trở thành công cụ học tập hoặc giải trí hữu ích, giúp người chơi rèn luyện kỹ năng logic và tư duy sáng tạo.

3. Phương pháp nghiên cứu

Nghiên cứu lý thuyết: Tìm hiểu và nghiên cứu tài liệu về lập trình Python, đặc biệt là các thư viện cần thiết như Pygame để phát triển ứng dụng đồ họa. Việc nắm vững lý thuyết sẽ giúp nhóm có cái nhìn sâu sắc về cách thức hoạt động của các thuật toán và công cụ đồ họa.

Phân tích yêu cầu: Xác định các yêu cầu cơ bản của trò chơi, bao gồm các chức năng chính như di chuyển đĩa, điều khiển giao diện đồ họa, xử lý sự kiện người dùng và tính toán điểm số. Phân tích kỹ các yêu cầu này giúp nhóm hiểu rõ mục tiêu và phạm vi của trò chơi.

Thiết kế và phát triển: Tiến hành thiết kế cấu trúc trò chơi, lập trình các chức năng theo các bước rõ ràng, từ việc xử lý logic cho đến xây dựng giao diện đồ họa. Các chức năng sẽ được tổ chức trong một cấu trúc mã nguồn chặt chẽ và dễ bảo trì.

Kiểm thử và đánh giá: Sau khi hoàn thành, nhóm tiến hành chạy thử trò chơi để kiểm tra lỗi và phản hồi từ người chơi. Quá trình này giúp nhận diện các điểm cần cải tiến, đảm bảo trò chơi hoạt động ổn định và mang lại trải nghiệm người dùng tốt.

PHẦN NỘI DUNG

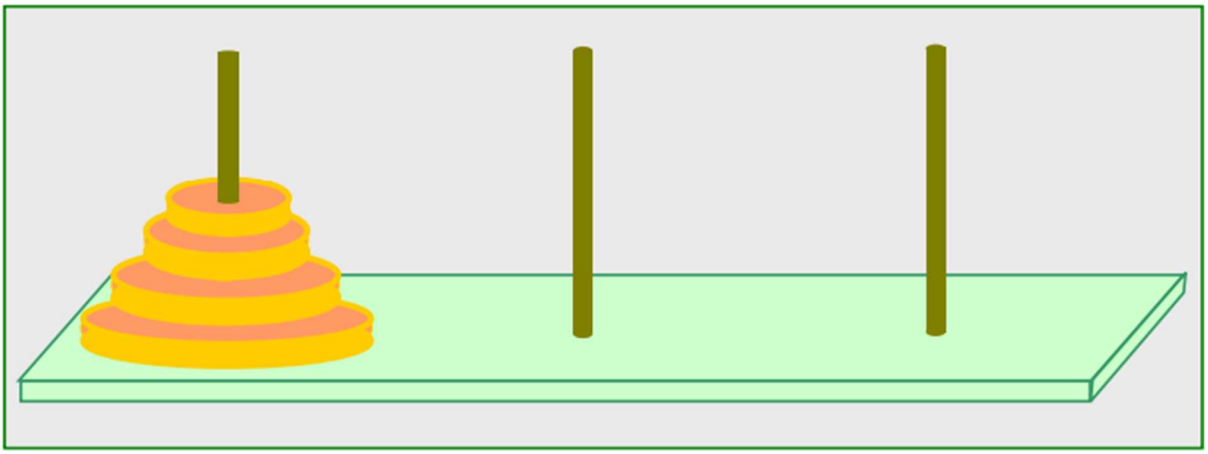
CHƯƠNG 1: CƠ SỞ LÝ THUYẾT

1.1. Sơ lược về tháp Hà Nội và tháp Hà Nội từ tính

Trò chơi Tháp Hà Nội có thể đã xuất hiện ở Đông Á từ thế kỷ 19 hoặc trước đó. Các đĩa được làm bằng sứ ở Trung Quốc, Nhật Bản và Việt Nam. Trò chơi này được đưa sang phương Tây lần đầu bởi nhà toán học người Pháp Edouard Lucas vào năm 1883. Trò chơi này nhanh chóng được các nhà toán học nghiên cứu sau đó, và trở thành ví dụ về phương pháp giải đệ quy kinh điển trong dạy học và tin học. Số lượng nước đi tối thiểu để giải Tháp Hà Nội là 2^n-1 nước đi, với n là số đĩa. Cấu trúc của trò chơi Tháp Hà Nội bao gồm ba trụ và n số đĩa. Quá trình giải câu đố tháp Hà Nội yêu cầu di chuyển từng đĩa một ở đĩa nguồn sao cho các đĩa di chuyển sang hết đĩa đích mà vẫn đúng luật. Ngoài ra còn có một biến thể của trò chơi tháp Hà Nội mang tính tư duy cao, số bước cần giải tăng đáng kể được gọi là Tháp Hà Nội từ tính.

1.1.1. Tháp Hà Nội

Câu đố tháp cổ điển Hà Nội bao gồm ba trụ và n đĩa. Quá trình giải câu đố ("trò chơi") yêu cầu di chuyển từng đĩa một bị giới hạn bởi một "quy tắc kích thước". Câu đố được giải quyết khi tất cả các đĩa được chuyển từ cột "nguồn" sang cột "đích".



Thành phần câu đố: Ba trụ bằng nhau, bộ N đĩa có đường kính khác nhau.

Thiết lập bắt đầu câu đố: N đĩa được sắp xếp theo thứ tự kích thước giảm dần từ trên xuống trên một trụ "Nguồn".

Di chuyển: Nâng một đĩa ra khỏi một cột và đặt nó vào một cột khác. Mỗi lần chỉ di chuyển được duy nhất 1 đĩa.

Quy tắc đặt đĩa: Không được đặt một đĩa lớn lên một đĩa nhỏ hơn.

Trạng thái kết thúc câu đố : N đĩa được sắp xếp thành một thứ tự kích thước giảm dần từ dưới lên trên trên cột "Đích".

1.1.2. Tháp Hà Nội từ tính

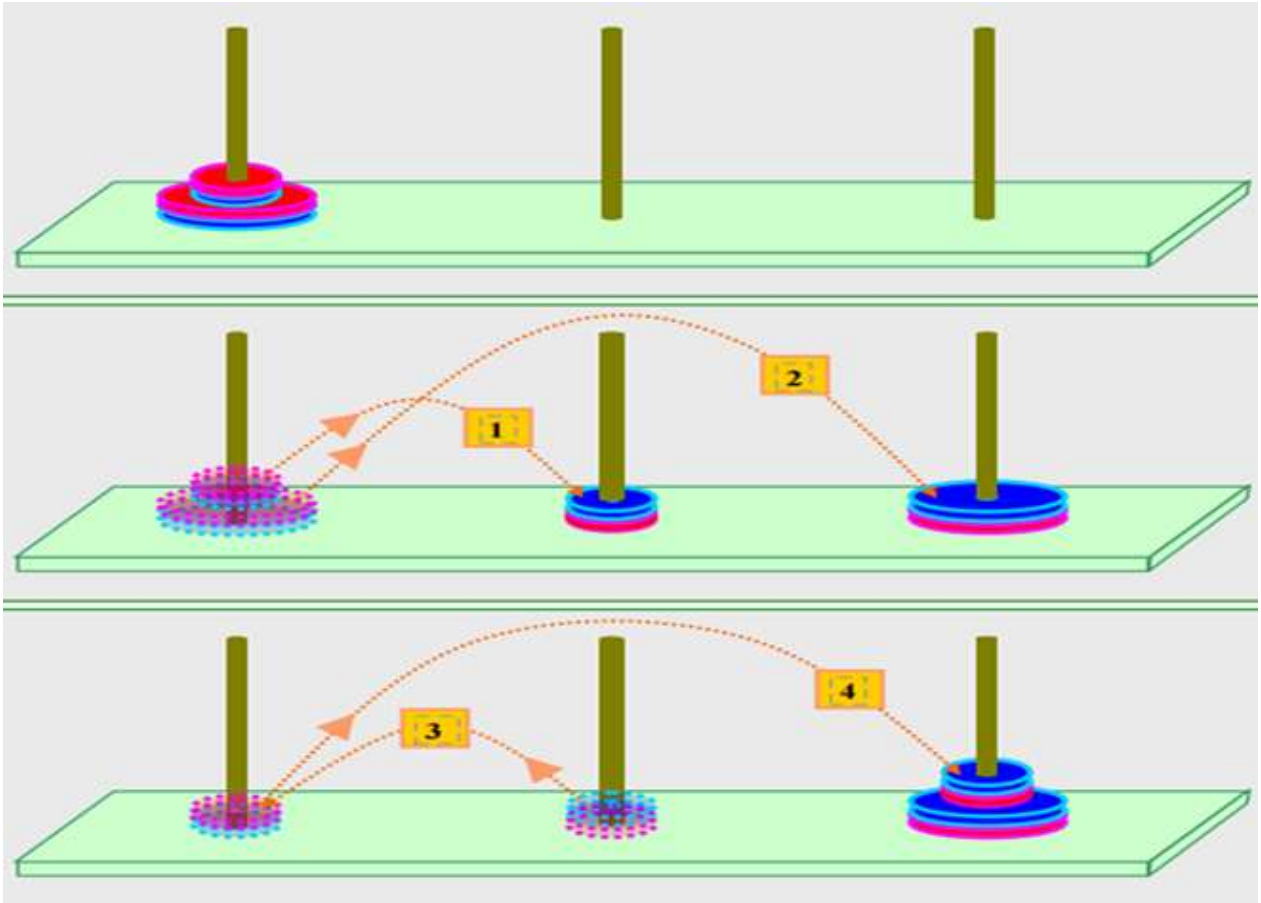
Trong bài toán Tháp Hà Nội từ tính, chúng ta vẫn sử dụng 3 trụ và N đĩa. Tuy nhiên, bản thân đĩa, định nghĩa nước đi và luật chơi đều được sửa đổi (mở rộng). Thành phần câu đố: Ba trụ bằng nhau**, M**ột bộ N đĩa có đường kính khác nhau**, M**ặt "dưới" của mỗi đĩa có màu Xanh và mặt "trên" của nó có màu Đỏ

Thiết lập bắt đầu câu đố: N đĩa được sắp xếp theo thứ tự kích thước giảm dần từ dưới lên trên trên cột "Nguồn". Mặt đỏ của mọi đĩa trong ngăn xếp hướng lên trên. Lưu ý rằng cài đặt bắt đầu câu đố đáp ứng "Quy tắc nam châm"(cùng màu thì đẩy, khác màu thì hút).

Di chuyển: Nhấc đĩa ra khỏi một cột, lật ngược đĩa và đặt nó lên một cột khác.

Quy tắt đặt đĩa: Đĩa nhỏ không thể "chở" đĩa lớn hơn (Không bao giờ đặt đĩa lớn lên một đĩa cái nhỏ hơn). Sự loại bỏ xảy ra giữa hai màu giống nhau (Không bao giờ đặt một đĩa sao cho mặt dưới của nó chạm vào mặt trên cùng màu).

Trạng thái kết thúc câu đố: N đĩa được sắp xếp theo thứ tự kích thước giảm dần từ trên xuống tuân theo quy tắt nam châm.



1.2. Sơ lược về thư viện Pygame trong Python

1.2.1. Thư viện Pygame là gì?

Pygame là một thư viện của ngôn ngữ lập trình Python và là một tập hợp các mô-đun Python được thiết kế riêng để lập trình trò chơi. Pygame được viết bởi Pete Shinners thay thế cho chương trình PySDL sau khi quá trình phát triển dự án này bị đình trệ. Chính thức phát hành từ năm 2000, Pygame được phát hành theo phần mềm miễn phí GNU Lesser General Public License.

Pygame có thể chạy trên nhiều nền tảng và hệ điều hành khác nhau. Với thư viện pygame trong Python, các nhà phát triển có thể sử dụng công cụ và chức năng mở rộng để tạo ra các trò chơi nhập vai ấn tượng. Bởi vậy, Pygame đang ngày càng phổ biến với nhà phát triển vì tính đơn giản, linh hoạt, dễ sử dụng.

1.2.2. Đặt điểm nổi bật của Pygame

Dưới đây là một số đặc điểm nổi bật của Pygame:

Pygame sử dụng Simple DirectMedia Layer (SDL), một thư viện phát triển đa nền tảng cho phép các nhà phát triển có thể truy cập vào phần cứng máy tính như đồ họa, âm thanh và thiết bị đầu vào.

Xây dựng các trò chơi trên nhiều nền tảng khác nhau như Windows, Mac, Linux thậm chí là cả các thiết bị di động

Nhà phát triển có thể quản lý tất cả các yếu tố trong quá trình phát triển trò chơi. Đó có thể là các chức năng như xuất đồ họa, xử lý sự kiện, hoạt ảnh, hiệu ứng âm thanh và phát lại nhạc

Cung cấp nhiều chức năng mở rộng hỗ trợ nhà phát triển tập trung phát triển trò chơi

API trực quan và dễ hiểu, hỗ trợ người mới sử dụng hay cả những nhà phát triển có kinh nghiệm đều có thể truy cập được

Nguồn tài nguyên và tài liệu phong phú, các nhà phát triển có thể sử dụng các mã nguồn mở miễn phí để phát triển dự án của mình

Tính đa phương tiện giúp nhà phát triển có thể ứng dụng để xử lí hình ảnh hay video, mô phỏng, công cụ giáo dục….

1.3. Áp dụng các cơ sở lý thuyết đã học trong python

+ Lập trình hướng đối tượng (OOP) trong python : Lớp và đối tượng.

+ Cấu trúc dữ liệu: Danh sách, tuple, từ điển.

+ Hàm và phương thức: Tổ chức mã nguồn và tái sử dụng mã.

+ Câu lệnh điều kiện: Điều khiển luồng chương trình dựa trên các điều kiện.

+ Vòng lặp: Thực hiện các thao tác lặp đi lặp lại.

+ Xử lý sự kiện trong Pygame: Xử lý tương tác của người dùng.

+ Thư viện Pygame: Vẽ đồ họa và quản lý thời gian.

+ Quản lý tệp: Đọc và ghi dữ liệu vào tệp.

+ Trì hoãn và thời gian: Kiểm soát thời gian trong trò chơi.

+ Xử lý và sắp xếp dữ liệu: Quản lý bảng xếp hạng và điểm số

1.4. Cấu trúc dữ liệu sử dụng trong Tháp Hà Nội và Tháp Hà Nội từ tính

1.4.1. Cấu trúc dữ liệu Stack

**Khái niệm*

Stack là một danh sách có thứ tự trong đó việc chèn và xoá được thực hiện ở một đầu được gọi là top - đỉnh. Phần tử cuối cùng được chèn là phần tử đầu tiên sẽ bị xoá. Do đó, nó được gọi là Last in First out (LIFO) hoặc First in Last Out (FILO) list.

**Đặc điểm*

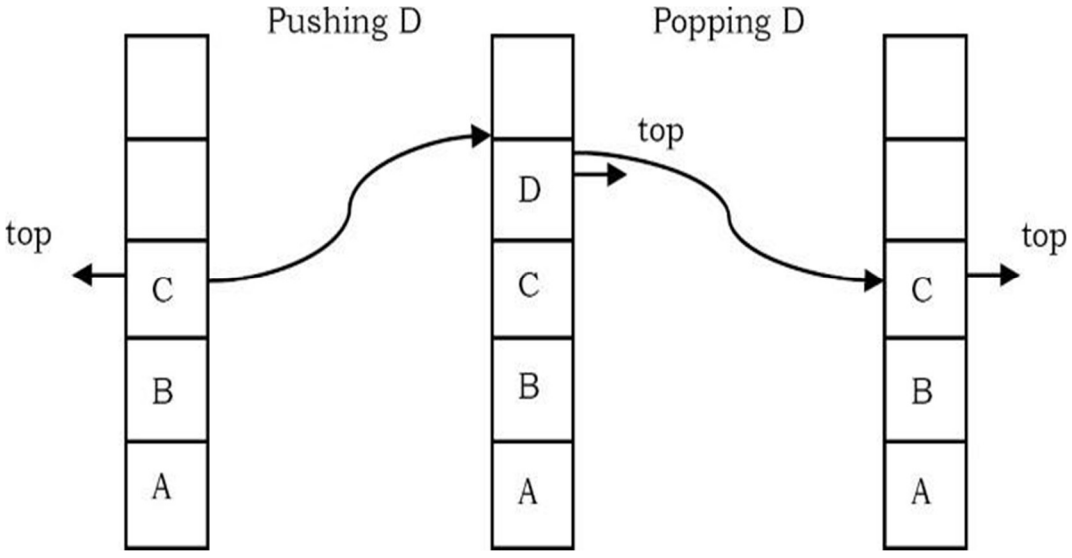
Stack có 2 thao tác cơ bản: thêm phần tử vào được gọi là push, xoá phần tử ra được gọi là pop. Việc cố gắng pop một stack trống được gọi là underflow và cố gắng đẩy một phần tử trong một stack đầy được gọi là overflow. Khi những điều trên xảy ra, chúng ta gọi chúng là exceptions (ngoại lệ).

Ví dụ: Hình ảnh sau minh họa cách hoạt động của ngăn xếp (stack) với các thao tác cơ bản "push" (đẩy) và "pop" (lấy ra).

Bước 1: Trạng thái ban đầu của ngăn xếp bao gồm các phần tử lần lượt từ dưới lên là A, B, và C. Phần tử trên cùng là C và được đánh dấu bằng top.

Bước 2: Pushing D, ở bước này ta thực hiện thao tác "push" phần tử D vào ngăn xếp. Phần tử D sẽ được thêm vào vị trí trên cùng, và con trỏ top sẽ được cập nhật để chỉ phần tử D. Bây giờ, ngăn xếp có thứ tự từ dưới lên là A, B, C, và D.

Bước 3: Popping , ở bước này ta thực hiện thao tác "pop", nghĩa là lấy phần tử D ra khỏi ngăn xếp. Sau khi "pop" phần tử D, con trỏ top sẽ quay về chỉ phần tử C, và phần tử D không còn trong ngăn xếp nữa. Hiện tại, ngăn xếp trở về trạng thái ban đầu với các phần tử A, B, và C.



1.4.2. Lý do chọn Stack

Việc sử dụng cấu trúc dữ liệu ngăn xếp Stack trong trò chơi Tháp Hà Nội là một lựa chọn lý tưởng bởi vì stack có thể quản lý số đĩa ở mỗi cột dễ dàng hơn và cơ chế hoạt động của stack (Last in, first out) cũng phù hợp với quy tắc di chuyển đĩa của trò chơi.

1.4.3. Nguyên lí hoạt động

Mỗi cột trong tháp Hà Nội tương ứng với một ngăn xếp (stack) độc lập, các đĩa được đặt lên các stack theo thứ tự từ lớn đến nhỏ với đĩa lớn nhất ở dưới và đĩa nhỏ nhất ở trên cùng. Ngăn xếp "source" chứa tất cả các đĩa ban đầu, "temp" là cột giữa, "target" là cột đích. Mỗi thao tác di chuyển đĩa giữa các cột thực chất là các thao tác push và pop trên các ngăn xếp này.

Ví dụ: Khi di chuyển một đĩa từ cột A sang cột B thì đĩa ở đỉnh cột A là đĩa nhỏ nhất được lấy ra (pop) khỏi cột A và được đưa vào (push) cột B trở thành đĩa đỉnh ở cột B. Thực hiện tương tự giữa cột A và C, cột B và cột C cho đến khi các đĩa ở cột A di chuyển hết đến cột C.

1.4.5. Cách sử dụng Stack trong chương trình

Khởi tạo các tháp (towers):

Trong trò chơi, có ba tháp được khởi tạo dưới dạng các danh sách (mỗi tháp là một danh sách). Các đĩa ban đầu được lưu trữ trong tháp đầu tiên.

```
towers = [[], [], []]
```

Ở đây, mỗi phần tử trong towers là một danh sách đại diện cho một tháp. Các đĩa được lưu trữ trong các tháp này.

Thao tác pop() (Lấy đĩa từ đỉnh tháp):

Khi người chơi muốn di chuyển một đĩa, họ phải chọn một tháp. Nếu tháp có đĩa, người chơi sẽ lấy đĩa từ đỉnh tháp (thực hiện thao tác pop()). Điều này tương ứng với thao tác lấy phần tử từ đỉnh ngăn xếp.

```
if selected_disk is None and towers[i]:
    selected_disk = towers[i].pop()
    selected_tower = i
```

Nếu không có đĩa nào được chọn (biến selected_disk là None) và tháp đã chọn (towers[i]) có đĩa, đĩa cuối cùng sẽ bị lấy ra từ tháp (pop()) và gán vào biến selected_disk.

Đĩa bị lấy ra từ đỉnh tháp (tương tự như việc pop từ ngăn xếp).

Thao tác append() (Thêm đĩa vào đỉnh tháp):

Sau khi người chơi đã chọn một tháp và di chuyển đĩa đến tháp mới, đĩa được thêm vào đỉnh của tháp đích bằng cách sử dụng append(). Đây là thao tác tương ứng với push trong ngăn xếp — thêm phần tử vào đỉnh của ngăn xếp.

```
if not towers[i] or towers[i][-1] > selected_disk:
    towers[i].append(selected_disk)
```

Nếu tháp đích (towers[i]) chưa có đĩa (tháp trống) hoặc đĩa trên đỉnh của tháp đích lớn hơn đĩa đã chọn, đĩa sẽ được thêm vào tháp đó (append()).

Đĩa được thêm vào đỉnh tháp (tương tự như thao tác push trong ngăn xếp).

Quy tắc di chuyển đĩa:

Trong trò chơi Towers of Hanoi, có một quy tắc quan trọng là không thể đặt một đĩa lớn lên đĩa nhỏ hơn. Điều này được kiểm tra trong mã bằng cách so sánh kích thước của đĩa đang di chuyển với đĩa ở đỉnh tháp đích. Nếu điều kiện này không được thỏa mãn, đĩa sẽ không được di chuyển và phải quay lại tháp ban đầu.

```
else:
    towers[selected_tower].append(selected_disk)
```

Nếu điều kiện di chuyển không hợp lệ (đĩa lớn hơn đĩa ở tháp đích), đĩa sẽ được đưa lại tháp cũ.

Các thao tác di chuyển đĩa:

```
    elif selected_disk is not None:
        if i != selected_tower:
            if not towers[i] or towers[i][-1] > selected_disk:
                towers[i].append(selected_disk)
                selected_disk = None
                selected_tower = None
                move_count += 1
            else:
                towers[selected_tower].append(selected_disk)
                selected_disk = None
                selected_tower = None
```

Chọn đĩa và di chuyển: Khi người chơi chọn một tháp và di chuyển đĩa, quy trình này có thể được xem như một chuỗi thao tác pop (lấy đĩa từ tháp) và append (thêm đĩa vào tháp).

1.5. Giải thuật Tháp Hà Nội

```
def tower_of_hanoi(towers, source, target, temp, n):
    if n == 1:
        target.append(source.pop())
        draw_towers(towers)
        pygame.display.update()
        time.sleep(0.5)
        return
    tower_of_hanoi(towers, source, temp, target, n - 1)
    target.append(source.pop())
    draw_towers(towers)
    pygame.display.update()
    time.sleep(0.5)
    clock.tick(framerate)
    tower_of_hanoi(towers, temp, target, source, n - 1)
```

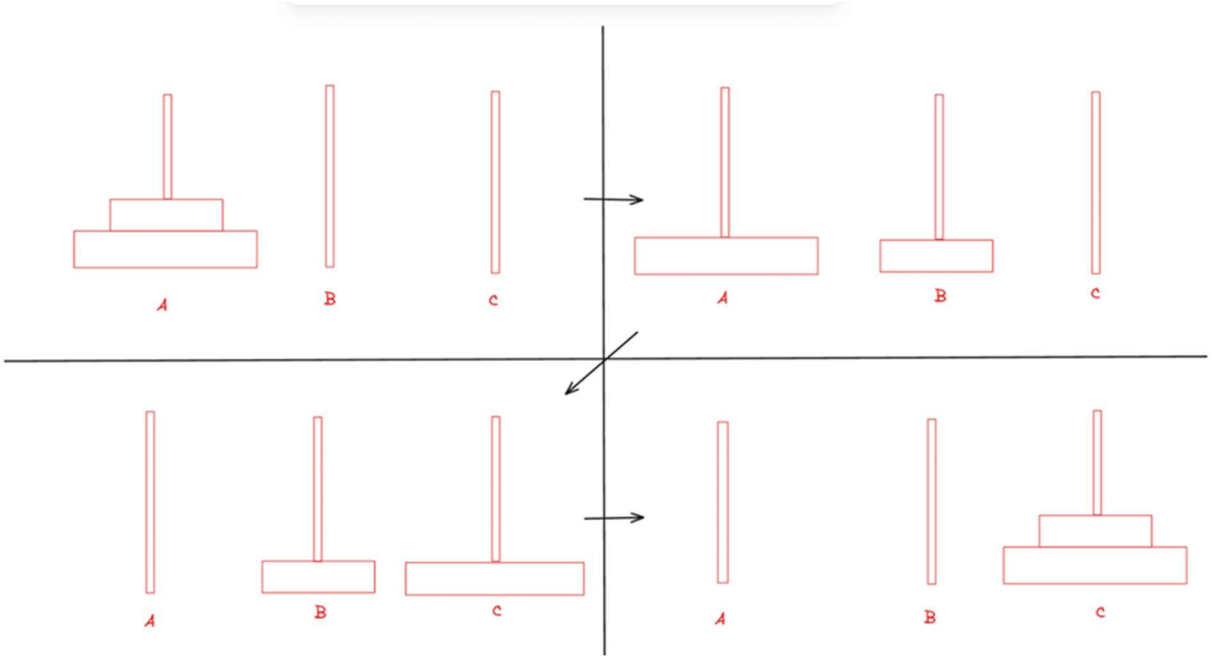
Để giải được bài toán này, chúng ta có ba bước :

Di chuyển n-1 đĩa ở trên cùng ở chiếc đinh ban đầu đến chiếc đinh trung gian.

Di chuyển chiếc đĩa lớn nhất ở dưới cùng của chiếc đinh đầu tiên đến chiếc đinh đích

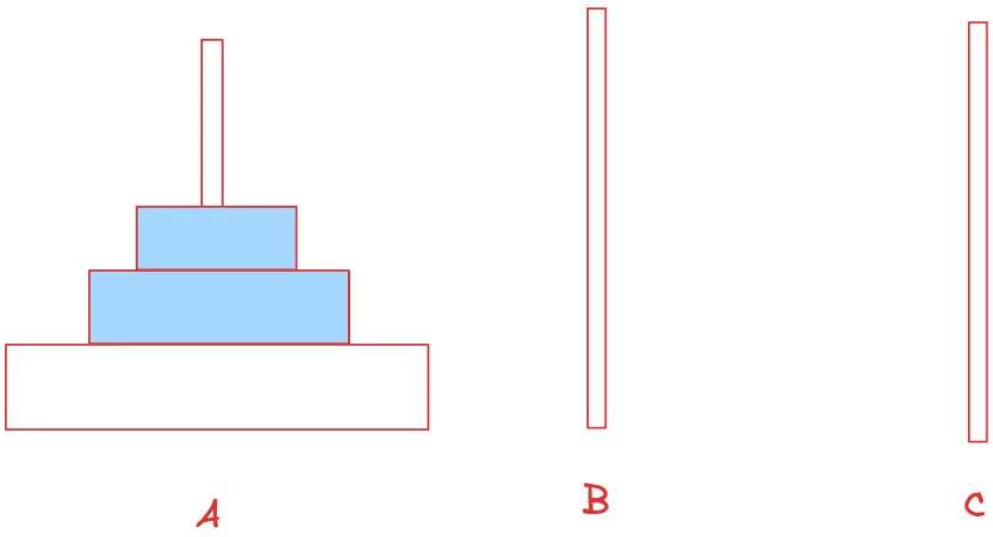
Di chuyển n-1 chiếc đĩa ở đinh trung gian sang đinh đích

Dưới đây là cách giải bài toán tháp Hà Nội với số đĩa là hai.



Với số đĩa là 2, công việc của chúng ta rất đơn giản, là di chuyển từng chiếc đĩa một với các bước giải đã nêu trên.

Vậy, với số đĩa lớn hơn 2, làm thế nào để có thể chuyển n-1 chiếc đĩa sang chiếc đinh trung gian.



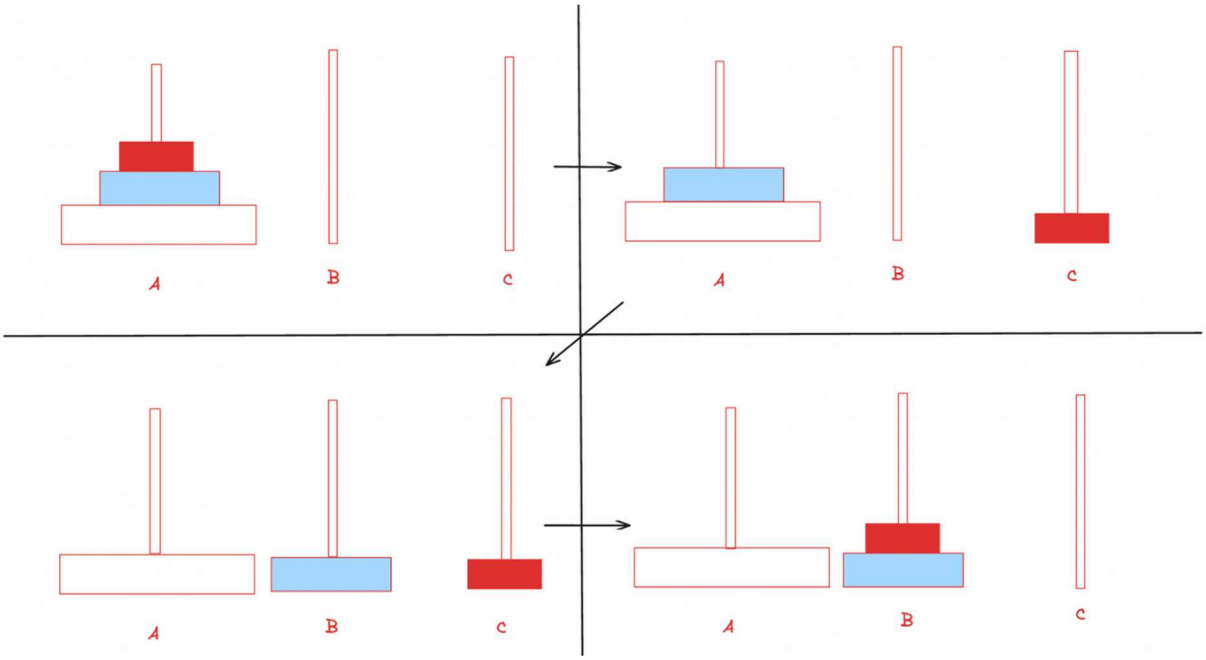
Từ hình ảnh trên, vấn đề của chúng ta là cần di chuyển cả 2 chiếc đĩa trên cùng ở chiếc đinh A sang chiếc đinh B.

Để có thể di chuyển 2 chiếc đinh này từ A sang B, ta sử dụng lại bài toán đã làm ở trường hợp 2 chiếc đĩa, khi đó, chúng ta có :

Di chuyển chiếc đĩa trên cùng ở đinh A sang đinh C

Di chuyển chiếc đĩa thứ hai sang cột B

Di chuyển chiếc đĩa ở cột C, là chiếc đĩa nhỏ nhất, sang cột B



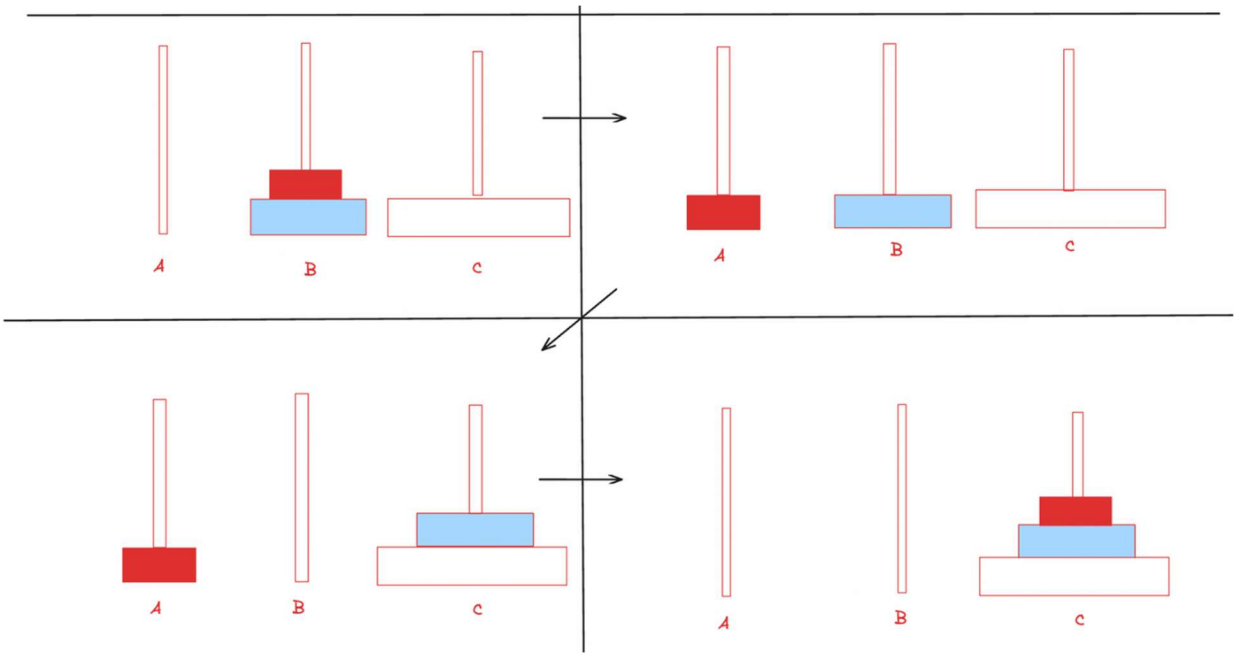
Các bước tiếp theo diễn ra như sau :

Di chuyển chiếc đĩa cuối cùng ở đinh A sang đinh C

Di chuyển chiếc đĩa trên cùng ở đinh B sang đinh A

Di chuyển chiếc đĩa còn lại ở đinh B sang đinh C

Cuối cùng, di chuyển chiếc đinh nhỏ nhất ở đinh A sang đinh C



Giải quyết bài toán lớn bằng một bài toán nhỏ hơn là biểu hiện rất rõ ràng cho việc chúng ta có thể xử lý bài toán này bằng đệ quy.

Chúng ta sẽ đi sâu hơn về cách giải bài toán tháp Hà Nội ở chương tiếp theo.

CHƯƠNG 2: TỔNG QUAN SƠ LƯỢC ĐỒ ÁN VÀ CHỨC NĂNG

2.1. Tổng quan sơ lược đồ án

Quá trình thực hiện đồ án gồm các bước chính:

Phân tích yêu cầu:

Xác định các chế độ chơi cần có (luyện tập, tự giải, bảng xếp hạng, máy giải).

Thiết kế giao diện tương tác đơn giản, trực quan thân thiện với người chơi.

Thêm phần âm thanh khi bắt đầu chơi giúp trò chơi thêm phần sinh động thú vị.

Lập trình các chức năng cốt lõi:

Xây dựng thuật toán đệ quy giải bài toán Tháp Hà Nội.

Thiết kế chức năng luyện tập, cho phép người chơi tự di chuyển các đĩa.

Tích hợp bảng xếp hạng để lưu trữ và hiển thị thành tích người chơi.

Xây dựng giao diện đồ họa:

Tạo menu chính, màn hình chơi, và các giao diện kết thúc.

Hiển thị hình ảnh đĩa và cọc trong Tháp Hà Nội, cùng các hiệu ứng hình ảnh.

Kiểm tra và tinh chỉnh:

Thử nghiệm các chế độ chơi, sửa lỗi và tối ưu hóa giao diện.

2.2. Chức năng của trò chơi

Chương trình tháp Hà Nội ba chế độ chính và một số chế độ phụ:

2.2.1. Chế độ chơi (PLAY)

Đấu xếp hạng (RANKING): người chơi sau khi hoàn thành trò chơi thì sẽ được cập nhập các thông tin về người chơi trên bảng SCOREBOARD

Chế độ thường (NORMAL): chế độ này người chơi chỉ cần di chuyển các đĩa từ cột đầu tiên sang cột cuối là hoàn thành

Chế độ nâng cao (ADVANCED): chế độ này người chơi không chỉ phải di chuyển các đĩa từ cột đầu sang cột cuối mà còn phải di chuyển dựa vào quy tắc nam châm, tức là khi di chuyển 1 đĩa từ cột này sang cột khác thì đĩa đổi màu và các đĩa trên một cột phải có màu giống nhau.

Luyện tập (PRACTICE): Ở chế độ này các đĩa sẽ được đặt ngẫu nhiên trên các cột và người chơi phải có tư duy nhạy bén để hoàn thành trò chơi nhanh chóng.

Giải tự động (AUTO SOLVE): chế độ này sẽ tự động giải dựa vào thuật toán đệ quy và hiện thị trực quan từng bước di chuyển giúp người chơi dễ dàng hình dung cách chơi.

2.2.2. Bảng điểm (SCOREBOARD)

Chế độ thường (NORMAL): Hiển thị các người chơi chế độ thường theo thứ tự giảm dần từ trên xuống dựa vào số lượng đĩa và số điểm.

Chế độ nâng cao (ADVANCED): Hiển thị các người chơi chế độ nâng cao theo thứ tự giảm dần từ trên xuống dựa vào số lượng đĩa và số điểm.

Ngoài ra ở mỗi chế độ còn có thêm chức năng tìm kiếm người chơi.

2.3. Các hàm chính và chức năng của chúng trong game Tháp Hà Nội

- **chonSoDia()**

```
def chonSoDia():
    global NUM_DISKS
    while True:
        SCREEN.blit(BG, (0, 0))

        USER_MOUSE_POS = pygame.mouse.get_pos()
        MENU_TEXT = get_font(30).render("CHOOSE NUMBER OF DISKS", True, "#b68f40")
        MENU_RECT = MENU_TEXT.get_rect(center=(360, 100))

        PLUS_BUTTON = Button(image=None, pos=(600, 250),
                             text_input="+", font=get_font(TEXT_SIZE), base_color="#d7fcd4", hovering_color="Red")
        MINUS_BUTTON = Button(image=None, pos=(120, 250),
                              text_input="-", font=get_font(TEXT_SIZE), base_color="#d7fcd4", hovering_color="Red")
        PLAY_BUTTON = Button(image=None, pos=(360, 350),
                             text_input="PLAY", font=get_font(TEXT_SIZE), base_color="#d7fcd4", hovering_color="Red")
        BACK_BUTTON = Button(image=None, pos=(360, 400),
                             text_input="BACK", font=get_font(TEXT_SIZE), base_color="#d7fcd4", hovering_color="Red")

        SCREEN.blit(MENU_TEXT, MENU_RECT)

        for button in [PLUS_BUTTON, MINUS_BUTTON, PLAY_BUTTON, BACK_BUTTON]:
            button.changeColor(USER_MOUSE_POS)
            button.update(SCREEN)

        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                pygame.quit()
                sys.exit()
            if event.type == pygame.MOUSEBUTTONDOWN:
                if PLUS_BUTTON.checkForInput(USER_MOUSE_POS):
                    NUM_DISKS += 1
                    if NUM_DISKS > 10:
                        NUM_DISKS = 10
                if MINUS_BUTTON.checkForInput(USER_MOUSE_POS):
                    NUM_DISKS -= 1
                    if NUM_DISKS < 1:
                        NUM_DISKS = 1
                if PLAY_BUTTON.checkForInput(USER_MOUSE_POS):
                    return
                if BACK_BUTTON.checkForInput(USER_MOUSE_POS):
                    play()

        font = pygame.font.Font("font/font.ttf", 40)
        disks_text = font.render(f"{NUM_DISKS}", True, (255, 255, 255))
        SCREEN.blit(disks_text, (360, 200))

        pygame.display.update()
        clock.tick(framerate)
```

Chức năng: Cho phép người chơi chọn số lượng đĩa cho trò chơi.

Thuật toán:

Người chơi được yêu cầu nhập số lượng đĩa.

Nếu số đĩa hợp lệ (theo một giới hạn nào đó, ví dụ 3 đến 10 đĩa), lưu giá trị đó để dùng trong việc khởi tạo tháp.

Sau khi chọn xong số lượng đĩa, số đĩa này sẽ được sử dụng trong các chế độ Tower of Hanoi để xác định bài toán cần giải.

- **draw_towers(towers)**

```python
def draw_towers(towers):
    SCREEN.blit(BG, (0, 0))
    tower_width = 10
    disk_height = 20
    tower_height = 150
    tower_positions = [(150, 300), (360, 300), (570, 300)]

    disk_colors = [
        (255, 0, 0), (0, 255, 0), (0, 0, 255), (255, 255, 0),
        (255, 165, 0), (128, 0, 128), (255, 192, 203), (0, 255, 255),
        (255, 0, 255), (128, 128, 128), (255, 255, 255)
    ]
    disk_color_mapping = []
    for i in range(3):
        tower_colors = []
        for j, disk in enumerate(towers[i]):
            tower_colors.append(disk_colors[disk % len(disk_colors)])
        disk_color_mapping.append(tower_colors)

    for pos in tower_positions:
        pygame.draw.rect(SCREEN, (0, 0, 0), (pos[0] - tower_width // 2, pos[1] - tower_height, tower_width, tower_height))

    for i in range(3):
        x, y = tower_positions[i]
        for j, disk in enumerate(towers[i]):
            disk_width = 20+ disk * 20
            disk_rect = pygame.Rect(x - disk_width // 2, y - (j + 1) * disk_height, disk_width, disk_height)
            disk_color = disk_color_mapping[i][j]
            pygame.draw.rect(SCREEN, disk_color, disk_rect)
            border_color = (0, 0, 0)
            border_width = 2
            pygame.draw.rect(SCREEN, border_color, disk_rect, border_width)
```

Chức năng: Vẽ ba tháp và các đĩa lên màn hình.

Thuật toán:

- Nhận vào tham số towers (một danh sách chứa các tháp, mỗi tháp là một danh sách các đĩa).

- Duyệt qua từng tháp trong danh sách towers.

- Với mỗi tháp, duyệt qua tất cả các đĩa trong tháp đó.

- Vẽ mỗi đĩa bằng cách sử dụng pygame.draw.rect() với chiều rộng và vị trí tương ứng với giá trị của đĩa.

- Các tháp sẽ được vẽ cạnh nhau, mỗi tháp cách nhau một khoảng cố định.

- Đĩa sẽ được xếp chồng lên nhau từ dưới lên, với kích thước giảm dần.

- **draw_towers_magnatic(towers)**

```
def draw_towers_magnatic(towers):
    SCREEN.blit(BG, (0, 0))
    tower_width = 10
    disk_height = 20
    tower_height = 150
    tower_positions = [(150, 300), (360, 300), (570, 300)]

    for pos in tower_positions:
        pygame.draw.rect(SCREEN, (0, 0, 0), (pos[0] - tower_width // 2, pos[1] - tower_height, tower_width, tower_height))

    for i, tower in enumerate(towers):
        x, y = tower_positions[i]
        for j, disk in enumerate(tower):
            disk_width = 20 + disk.size * 20
            disk_rect = pygame.Rect(x - disk_width // 2, y - (j + 1) * disk_height, disk_width, disk_height)

            disk_color = (255, 0, 0) if disk.isRedTop else (0, 0, 255)
            pygame.draw.rect(SCREEN, disk_color, disk_rect, border_radius=10)
            border_color = (0, 0, 0)
            border_width = 2
            pygame.draw.rect(SCREEN, border_color, disk_rect, border_width, border_radius=10)
```

Chức năng: Vẽ ba tháp với các đĩa có thuộc tính từ tính, đỉnh đĩa có màu đỏ (isRedTop).

Thuật toán:

- Nhận vào tham số towers (giống như draw_towers()).
- Duyệt qua từng tháp trong danh sách towers.
- Với mỗi tháp, duyệt qua tất cả các đĩa trong tháp đó.
- Kiểm tra thuộc tính isRedTop của từng đĩa:
    + Nếu đĩa có thuộc tính isRedTop, vẽ nó bằng màu đỏ.
    + Nếu không, vẽ đĩa bằng màu mặc định (màu xanh lam).
- Vị trí và kích thước của đĩa được tính như trong draw_towers().
- Lượt di chuyển.

- tower_of_hanoi(towers, source, target, temp, n)

```
def tower_of_hanoi(towers, source, target, temp, n):
    if n == 1:
        target.append(source.pop())
        draw_towers(towers)
        pygame.display.update()
        time.sleep(0.5)
        return
    tower_of_hanoi(towers, source, temp, target, n - 1)
    target.append(source.pop())
    draw_towers(towers)
    pygame.display.update()
    time.sleep(0.5)
    clock.tick(framerate)
    tower_of_hanoi(towers, temp, target, source, n - 1)
```

Chức năng: Giải bài toán Tower of Hanoi tự động bằng thuật toán đệ quy.

Thuật toán:

- Đệ quy di chuyển các đĩa từ tháp nguồn (source) sang tháp đích (target) thông qua tháp tạm (temp).

- Nếu chỉ còn một đĩa, di chuyển đĩa đó từ tháp nguồn sang tháp đích.

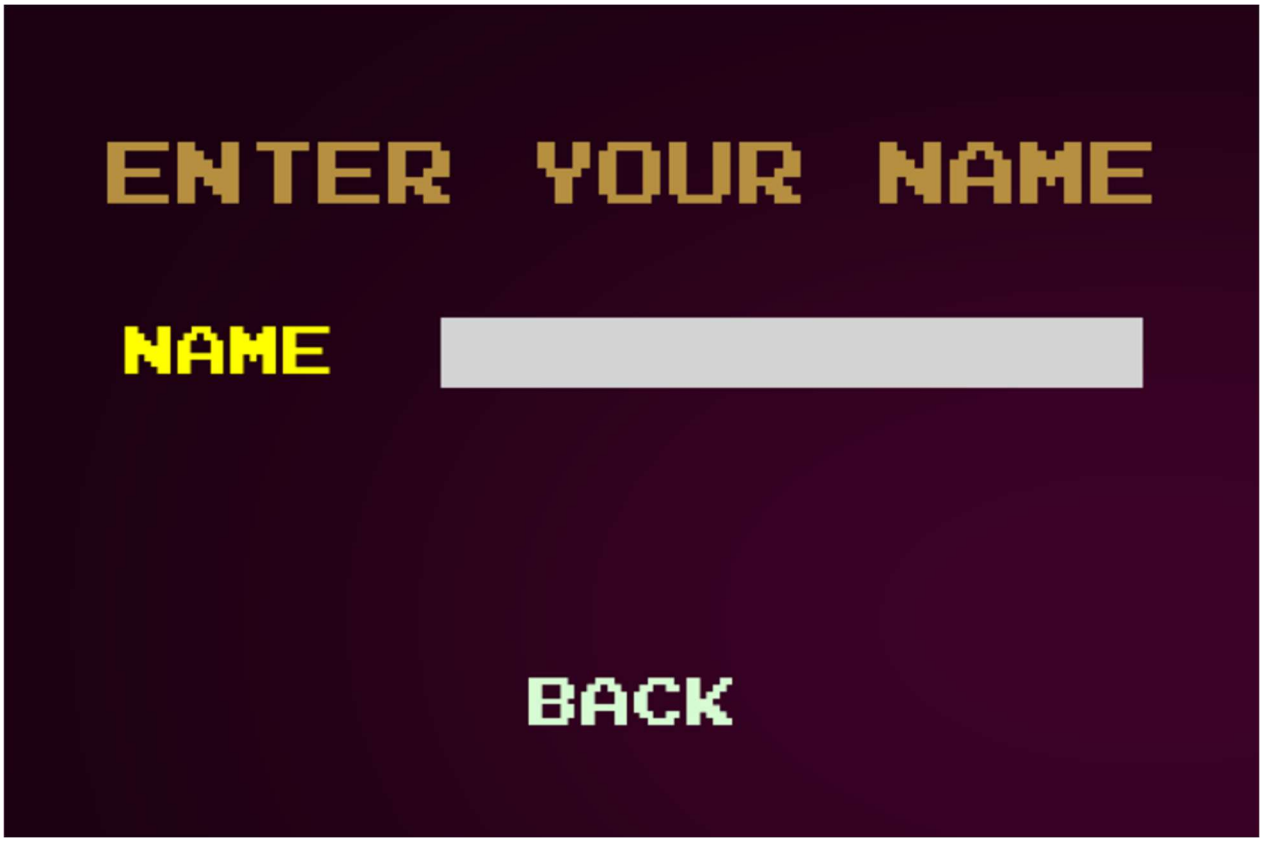
- Nếu có nhiều hơn một đĩa, di chuyển n-1 đĩa từ tháp nguồn sang tháp tạm, sau đó di chuyển đĩa còn lại từ tháp nguồn sang tháp đích, cuối cùng di chuyển n-1 đĩa từ tháp tạm sang tháp đích.

2.4. Giao diện của trò chơi

Giao diện trang chủ chính của trò chơi

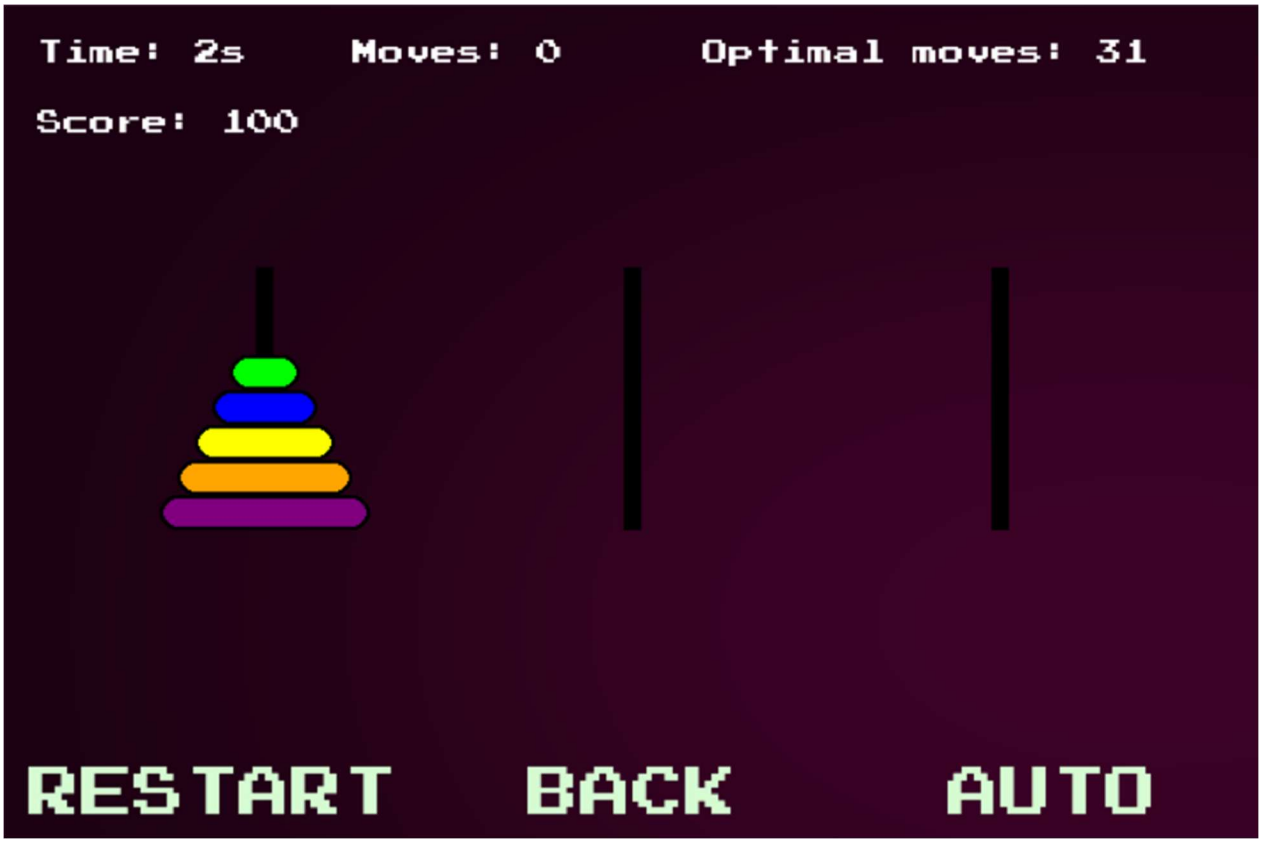
Giao diện khi ấn vào nút play

Giao diện chọn chế độ chơi khi ấn Ranking

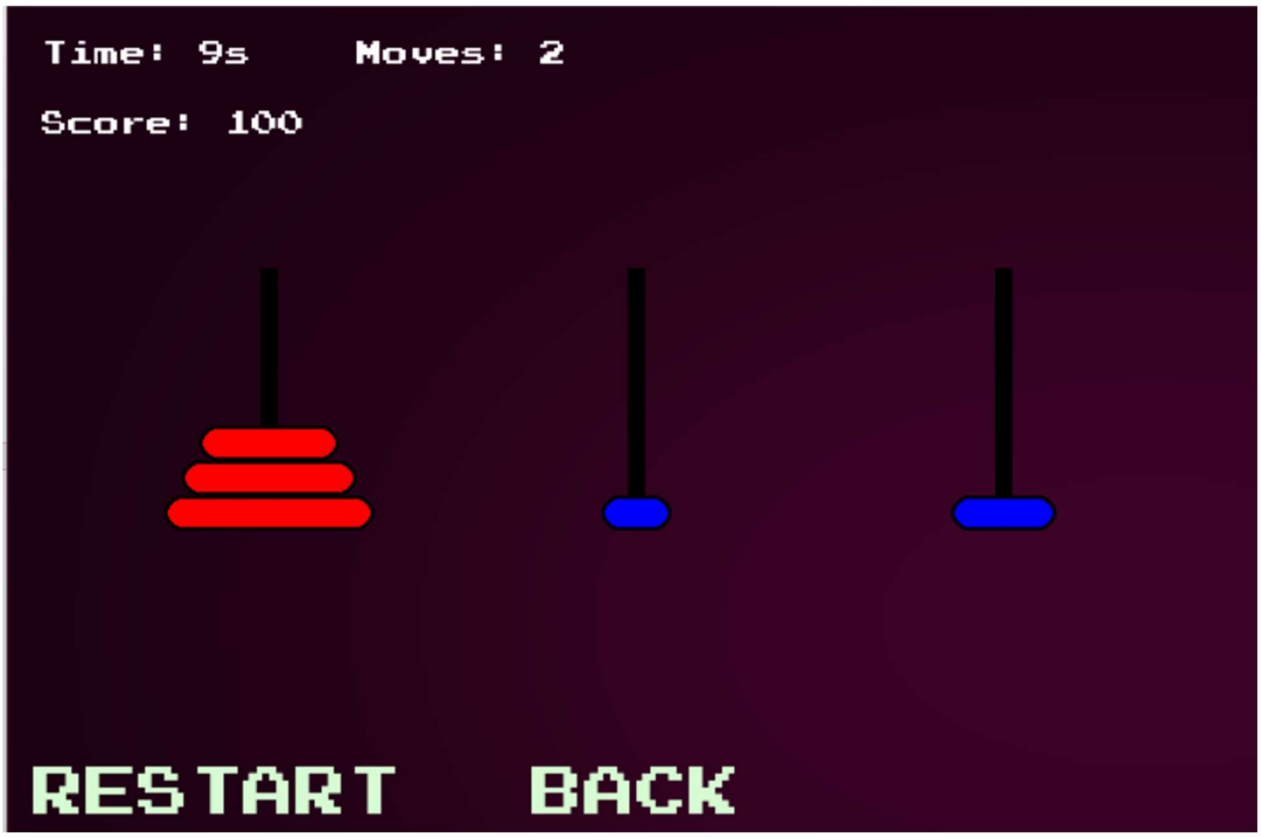


Giao diện nhập tên người dùng

Giao diện chọn số đĩa

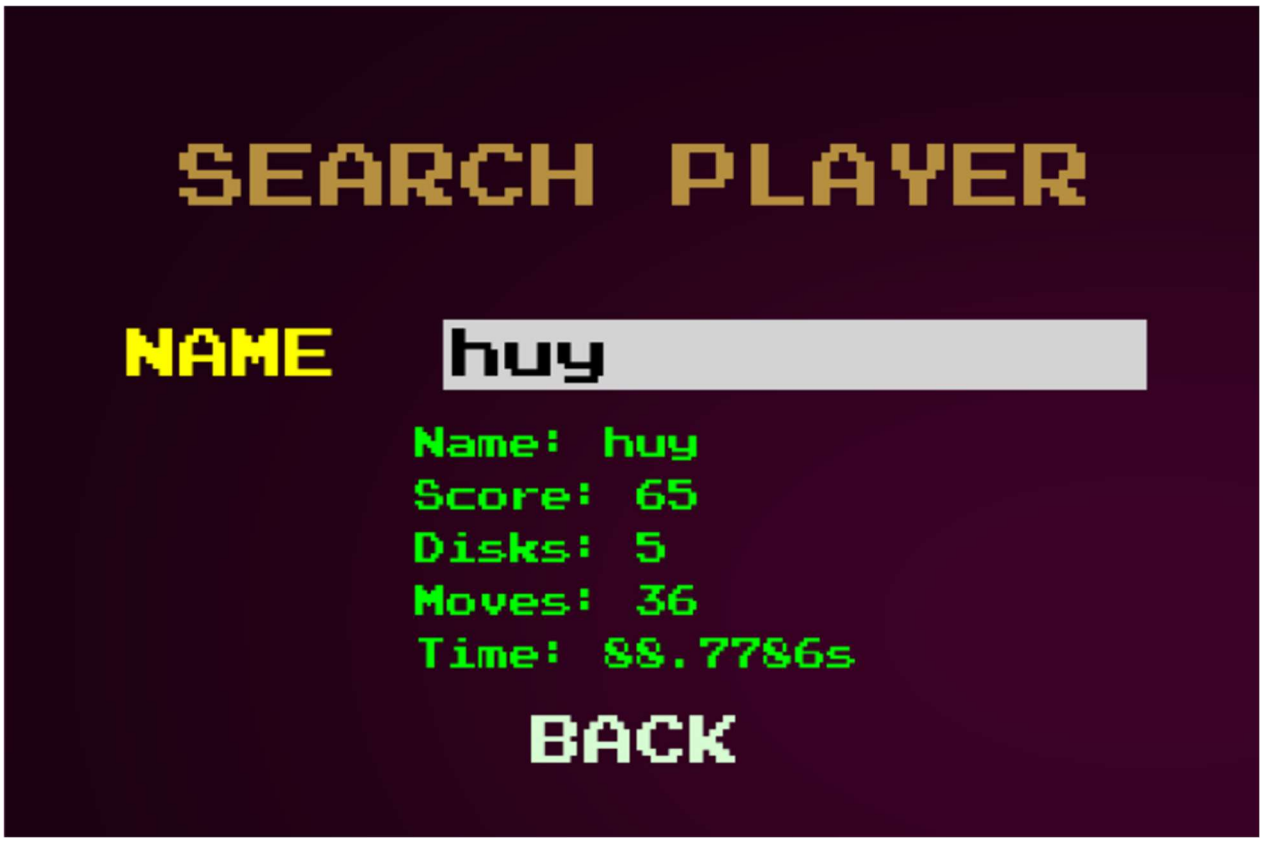


Giao diện khi chơi tháp Hà Nội thường



Giao diện khi chơi tháp Hà Nội nâng cao

Giao diện bảng điểm người chơi



Giao diện tìm kiếm người chơi

Giao diện chúc mừng chiến thắng

KẾT LUẬN

1. Kết quả đạt được

Trò chơi "Towers of Hanoi" đã được hoàn thiện về cơ bản, đáp ứng các yêu cầu của một trò chơi điển hình. Đồng thời, các chức năng bổ sung đã được triển khai nhằm nâng cao trải nghiệm người chơi và tính tối ưu của trò chơi.

Các điểm nổi bật đạt được:

Chạy ổn định: Trò chơi hoạt động ổn định qua các thử nghiệm trên máy, không xảy ra lỗi nghiêm trọng trong quá trình vận hành.

Giao diện rõ ràng: Thiết kế giao diện thân thiện, dễ sử dụng, giúp người chơi nhanh chóng làm quen với các chức năng và lối chơi.

Còn có âm thành nền của trò chơi và khi nhấp chuột vào các nút. Cuối game sau khi hoàn thành trò chơi còn có âm thanh chúc mừng chiến thắng kèm với hiệu ứng pháo hoa

Nhiều chức năng bổ sung:

Hệ thống bảng xếp hạng cho phép người chơi so sánh kết quả.

Chế độ luyện tập với các bài toán ngẫu nhiên.

Chế độ tự động giải, giúp người chơi tham khảo cách giải tối ưu.

Chế độ nâng cao giúp tăng tư duy nhạy bén

Chức năng nhập tên, theo dõi điểm số và thời gian.

Chức năng tìm kiếm người chơi dựa trên tên người chơi và hiển thị thông tin của người chơi đó

Tối ưu hóa trải nghiệm: Các tính năng được tích hợp mượt mà, từ giao diện menu chính đến các chế độ chơi và bảng xếp hạng.

2. Hướng phát triển trong tương lai

Mở rộng các chế độ chơi:

Chế độ nhiều người chơi (Multiplayer): Cho phép hai hoặc nhiều người chơi cạnh tranh giải bài toán Tower of Hanoi, theo dõi ai hoàn thành nhanh nhất hoặc ít bước nhất.

Chế độ thử thách (Challenge Mode): Tạo các thử thách với điều kiện đặc biệt, ví dụ: giới hạn số bước hoặc thời gian giải.

Nâng cao giao diện và đồ họa:

Sử dụng đồ họa 3D để tạo cảm giác chân thực hơn cho các tháp và đĩa.

Phát triển trí tuệ nhân tạo (AI):

Tạo đối thủ ảo để người chơi có thể cạnh tranh, ví dụ: AI giải bài toán với tốc độ cao và tối ưu nhất.

Hỗ trợ hướng dẫn thông minh, gợi ý các bước đi cho người chơi khi gặp khó khăn.

Mở rộng tính năng bảng xếp hạng:

Xây dựng hệ thống xếp hạng trực tuyến để so sánh thành tích với người chơi toàn cầu.

Thêm các danh hiệu hoặc huy hiệu (achievements) để khuyến khích người chơi.

Tích hợp trên các nền tảng khác:

Phát triển phiên bản dành cho thiết bị di động (Android/iOS) với giao diện cảm ứng tối ưu.

Đưa trò chơi lên các nền tảng web hoặc console để mở rộng đối tượng người dùng.

Tùy biến mức độ khó:

Cho phép người chơi tự thiết lập số lượng tháp hoặc quy tắc riêng để tăng tính thử thách.

Cung cấp các bài toán Tower of Hanoi phức tạp hơn với số lượng đĩa lớn hơn hoặc điều kiện bổ sung.

Hỗ trợ ngôn ngữ và giáo dục:

Hỗ trợ nhiều ngôn ngữ để tiếp cận người chơi toàn cầu.

Thêm phần hướng dẫn lý thuyết và bài học về thuật toán Tower of Hanoi để hỗ trợ học sinh/sinh viên.

TÀI LIỆU THAM KHẢO

1. Jain, S. (2024, September 4). Python Program for Tower of Hanoi. GeeksforGeeks. Retrieved November 20, 2024, from https://www.geeksforgeeks.org/python-program-for-tower-of-hanoi/

2. ICANTECH. (2023, 10 10). Pygame là gì? Tất cả những gì bạn cần biết về lập trình Pygame trong Python. Retrieved 11 15, 2024, from https://www.icantech.vn/kham-pha/lap-trinh-pygame

3. Tower of Hanoi. (n.d.). Wikipedia. Retrieved November 20, 2024, from https://en.wikipedia.org/wiki/Tower_of_Hanoi

4. Nick Koumaris. (2024, 6). Pygame-buttons-tutorial. Pygame-buttons-tutorial. Retrieved 11 1, 2024, from https://github.com/educ8s/Pygame-buttons-tutorial

5. GEEKSFORGEEKS. (2023, September 28). Bài toán tháp Hà Nội và cách giải sử dụng Đệ Quy. 200Lab. Retrieved November 26, 2024, from https://200lab.io/blog/bai-toan-thap-ha-noi/