

TRƯỜNG ĐẠI HỌC SÀI GÒN
KHOA CÔNG NGHỆ THÔNG TIN



PHÁT TRIỂN PHẦN MỀM MÃ NGUỒN MỞ

xây dựng game

Bingo Bằng Python

GVHD: Từ Lãng Phiêu

SV:

Trần Dương Dắc Lộc - 3121410304

Nguyễn Hữu Nam - 3121410333

Trần An Mẫn - 3121410318

TP. HỒ CHÍ MINH, THÁNG 5/2024

Mục lục

1	Tổng Quan Về Đề Tài	4
1.1	Lý do chọn đề tài	4
1.2	Mục tiêu của đề tài	4
2	Giới Thiệu Công Cụ Phát Triển	5
2.1	Giới thiệu Python	5
2.2	Giới thiệu thư viện Pygame của Python	5
2.3	Giới thiệu thư viện Pickle của Python	6
2.4	Giới thiệu thư viện Socket của Python	6
3	Hướng Dẫn Cài Đặt Game	7
3.1	Cài đặt môi trường	7
3.2	Cài đặt ứng dụng	7
4	Xây Dựng Game Bingo Bằng Python	8
4.1	Giới thiệu game Bingo	8
4.2	Quy tắc chơi game Bingo	8
4.3	Các thành phần xây dựng game Bingo	9
4.3.1	Giao diện	9
4.3.2	Các lớp chính	16
4.3.3	Các hàm xử lý chính	20
5	Kết Luận	25
5.1	Dánh giá	25
5.2	Hướng phát triển	25
6	Tài Liệu Tham Khảo	26



LỜI MỞ ĐẦU

Trong những năm gần đây, Python đã trở thành một trong những ngôn ngữ lập trình phổ biến và được ưa chuộng nhất trong cộng đồng lập trình viên bởi tính dễ học, dễ sử dụng và khả năng ứng dụng rộng rãi của nó. Đặc biệt, Python còn cung cấp nhiều thư viện hỗ trợ phát triển game như Pygame, Arcade, Panda3D, giúp cho việc xây dựng các trò chơi trở nên đơn giản và hiệu quả hơn.

Bingo là một trò chơi dân gian quen thuộc với nhiều người, đặc biệt là trong các buổi họp mặt gia đình hoặc các sự kiện cộng đồng. Trò chơi không chỉ mang lại niềm vui, tiếng cười mà còn giúp rèn luyện trí nhớ và kỹ năng quan sát. Vì vậy, việc phát triển một game Bingo trên nền tảng kỹ thuật số sẽ là một sự kết hợp hoàn hảo giữa truyền thống và hiện đại.

Dề tài "Xây dựng game Bingo bằng Python" sẽ hướng dẫn chi tiết từng bước từ việc thiết kế, lập trình, kiểm thử đến phát hành game. Qua đó, người đọc không chỉ nắm bắt được quy trình phát triển một ứng dụng game mà còn có thể áp dụng những kiến thức học được vào các dự án khác trong tương lai.

Mong rằng, đề tài này sẽ mang lại những thông tin hữu ích và góp phần khơi dậy niềm đam mê lập trình trong mỗi người đọc. Cảm ơn các thầy cô, bạn bè và gia đình đã động viên và hỗ trợ để đề tài này được hoàn thành.



LỜI CẢM ƠN

Chúng em xin chân thành cảm ơn Khoa Công nghệ thông tin, trường Đại học Sài Gòn đã tạo điều kiện cho chúng em được thực hiện đồ án môn học "Phát triển phần mềm mã nguồn mở". Chúng em cũng xin chân thành cảm ơn thầy Từ Lãng Phiêu đã giảng dạy cho chúng em những kiến thức cần thiết cho môn học này. Những kiến thức đó đã giúp cho chúng em rất nhiều trong quá trình làm đồ án báo cáo môn học. Chúng em xin chân thành cảm ơn quý Thầy cô trong Khoa đã tận tình giảng dạy và trang bị cho chúng em những kiến thức cần thiết trong thời gian qua. Mặc dù nhóm đã cố gắng hoàn thành đồ án môn học với tất cả nỗ lực của từng thành viên trong nhóm, nhưng đồ án chắc chắn không tránh khỏi những thiếu sót nhất định, rất mong nhận được sự cảm thông, chia sẻ và tận tình đóng góp chỉ bảo của quý Thầy Cô.



1 Tổng Quan Về Đề Tài

1.1 Lý do chọn đề tài

- Học hỏi và nâng cao kỹ năng lập trình Python: Việc xây dựng một game như Bingo giúp người học củng cố kiến thức về lập trình Python thông qua việc triển khai các khái niệm cơ bản và nâng cao như xử lý danh sách, điều kiện, vòng lặp, và các hàm.
- Tính ứng dụng thực tế: Game Bingo có thể được sử dụng trong nhiều tình huống khác nhau, từ giải trí cá nhân đến tổ chức sự kiện hoặc các buổi học nhóm, giúp tăng tính tương tác và hứng thú.
- Phát triển tư duy logic và giải quyết vấn đề: Khi xây dựng game Bingo, lập trình viên sẽ phải đổi mới với nhiều vấn đề cần giải quyết như kiểm tra sự trùng lặp của số, xác định kết quả thắng/thua, và tối ưu hóa mã nguồn.
- Dễ dàng triển khai và chia sẻ: Một game như Bingo viết bằng Python có thể dễ dàng được triển khai và chia sẻ với người khác, cho phép lập trình viên thu thập phản hồi và cải thiện kỹ năng thông qua thực tế sử dụng.

1.2 Mục tiêu của đề tài

- Phát triển một phiên bản game Bingo hoàn chỉnh với các chức năng cơ bản, kiểm tra trùng thưởng, và thông báo kết quả.
- Phát triển khả năng giải quyết vấn đề và tư duy logic thông qua việc giải quyết các thách thức trong quá trình xây dựng game, chẳng hạn như kiểm tra sự trùng lặp của số, xác định kết quả thắng/thua, và tối ưu hóa mã nguồn.
- Mở rộng game với các tính năng nâng cao như chế độ chơi nhiều người, các biến thể của luật chơi, hoặc tích hợp mạng để chơi trực tuyến.



2 Giới Thiệu Công Cụ Phát Triển

2.1 Giới thiệu Python

Python là một ngôn ngữ lập trình mã nguồn mở, hướng đối tượng cấp cao, mạnh mẽ và thông dịch. Được Guido van Rossum tạo ra và lần đầu ra mắt vào năm 1991. Ban đầu, Python được phát triển để chạy trên nền Unix. Nhưng rồi theo thời gian, Python dần mở rộng sang mọi hệ điều hành từ MS-DOS đến Mac OS, OS/2, Windows, Linux và các hệ điều hành khác thuộc họ Unix. Mặc dù sự phát triển của Python có sự đóng góp của rất nhiều cá nhân, nhưng Guido van Rossum hiện nay vẫn là tác giả chủ yếu của Python. Ông giữ vai trò chủ chốt trong việc quyết định hướng phát triển của Python.

Python luôn được xếp hạng vào những ngôn ngữ lập trình phổ biến nhất.

Ưu điểm

- Là ngôn ngữ có hình thức và cấu trúc rõ ràng, cú pháp ngắn gọn.
- Có trên tất cả các nền tảng hệ điều hành.
- Tương thích mạnh mẽ với số lượng thư viện khổng lồ.
- Tốc độ xử lý cực nhanh, có thể tạo ra những chương trình từ những script siêu nhỏ tới những phần mềm cực lớn.

Nhược điểm

- Không có các thuộc tính như: *protected*, *private*, *public*, không có vòng lặp *do...while* và *switch...case*.
- Tốc độ xử lý vẫn kém hơn Java và C++.

2.2 Giới thiệu thư viện Pygame của Python

Pygame là một bộ mô-đun Python đa nền tảng được thiết kế để viết trò chơi điện tử. Nó bao gồm đồ họa máy tính và thư viện âm thanh được thiết kế để sử dụng với ngôn ngữ lập trình Python.

Pygame sử dụng thư viện Simple DirectMedia Layer (SDL), với mục đích cho phép phát triển trò chơi máy tính trong thời gian thực mà không cần cơ chế bậc thấp của ngôn ngữ lập trình C và các dẫn xuất của nó. Điều này dựa trên giả định rằng các chức năng đắt tiền nhất bên trong trò chơi có thể được trừu tượng hóa khỏi logic trò chơi, do đó có thể sử dụng ngôn ngữ lập trình bậc cao, chẳng hạn như Python, để cấu trúc trò chơi.

Các tính năng khác mà SDL không có bao gồm toán học vectơ, phát hiện va chạm, quản lý đồ họa 2D, hỗ trợ MIDI, camera, thao tác mảng pixel, chuyển đổi, lọc, hỗ trợ phông chữ freetype nâng cao và v.v.

Các ứng dụng sử dụng Pygame có thể chạy trên điện thoại và máy tính bảng Android với việc sử dụng Bộ phụ Pygame cho Android (pgs4a). Âm thanh, rung, bàn phím và gia tốc kế được hỗ trợ trên Android.



2.3 Giới thiệu thư viện Pickle của Python

`pickle` là một thư viện chuẩn của Python được sử dụng để tuần tự hóa và giải tuần tự hóa các đối tượng Python. Quá trình tuần tự hóa (serialization) là việc chuyển đổi các đối tượng trong bộ nhớ thành một định dạng có thể lưu trữ hoặc truyền qua mạng, trong khi giải tuần tự hóa (deserialization) là quá trình ngược lại, chuyển đổi dữ liệu tuần tự hóa trở lại thành các đối tượng Python.

Thư viện `pickle` cho phép lưu trữ các cấu trúc dữ liệu phức tạp, bao gồm các đối tượng tùy chỉnh, vào các tệp hoặc truyền qua mạng để sử dụng sau này. Đây là một công cụ mạnh mẽ cho việc bảo quản trạng thái của chương trình hoặc trao đổi dữ liệu giữa các tiến trình Python.

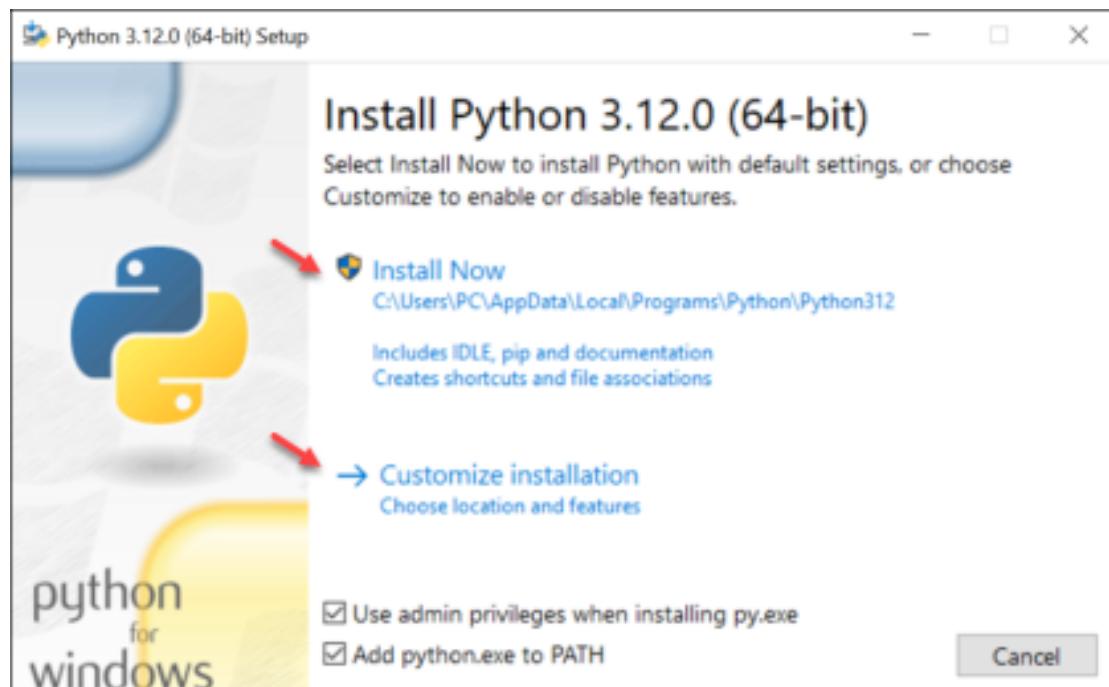
2.4 Giới thiệu thư viện Socket của Python

Thư viện `socket` của Python cung cấp giao diện cấp thấp để tương tác với mạng máy tính. Nó cho phép các lập trình viên tạo ra và quản lý các kết nối mạng, truyền và nhận dữ liệu qua các giao thức mạng khác nhau như TCP, UDP. Thư viện này rất quan trọng cho việc phát triển các ứng dụng mạng, từ các ứng dụng chat đơn giản đến các hệ thống phân tán phức tạp.

3 Hướng Dẫn Cài Đặt Game

3.1 Cài đặt môi trường

Truy cập vào trang web [python.org](https://www.python.org) và chọn phiên bản tương ứng với hệ điều hành của máy. Chọn tùy chọn Add Python 3.12 to PATH và nhấn Install Now. Bạn mở cmd từ thanh tìm



Hình 1: Cài đặt Python

kiểm trên window -> gõ: python --version -> enter nếu nó ra Python version là cài thành công

3.2 Cài đặt ứng dụng

Xem file README.md trên github ứng dụng



4 Xây Dựng Game Bingo Bằng Python

4.1 Giới thiệu game Bingo

Bingo là một trò chơi may rủi phổ biến, được chơi rộng rãi trên khắp thế giới. Trò chơi này thường được tổ chức trong các sự kiện xã hội, câu lạc bộ và đặc biệt là các hội trường lớn, nơi nhiều người có thể tham gia cùng một lúc. Bingo không chỉ là một trò chơi giải trí mà còn mang tính cộng đồng cao, kết nối mọi người qua những phút giây thư giãn và vui vẻ.

4.2 Quy tắc chơi game Bingo

Khi game khởi động, bạn cần phải nhập tên (đây cũng là tên phòng khi bạn tạo phòng) và nhấn nút Enter để tiếp tục

Sau đó màn hình menu sẽ xuất hiện gồm 3 lựa chọn:

1. Host Room: Bạn sẽ tạo phòng để chơi
2. Join Room: Tìm các phòng của các người chơi khác trong cùng mạng nội bộ
3. Exit: Thoát game

Host Room: Khi bạn tạo phòng bạn có thể đợi người khác tham gia (1 phòng có tối đa 10 người tham gia), hoặc có thể chơi ngay bằng cách bấm nút Ready.

Khi tất cả người chơi ready phòng sẽ bắt đầu sau 5s. Tiếp theo đó sẽ là 15s để quyết định đặt cược con số may mắn của bạn bằng cách chọn và nhập số tiền muốn cược sau đó bấm nút để xác nhận. Bạn có thể chọn tối đa 4 số

Sau khi hết 15s chọn số may mắn. Xúc xắc sẽ bắt đầu quay ngẫu nhiên. Có 3 xúc xắc được quay. Con số may mắn là tổng của 3 xúc xắc cộng lại

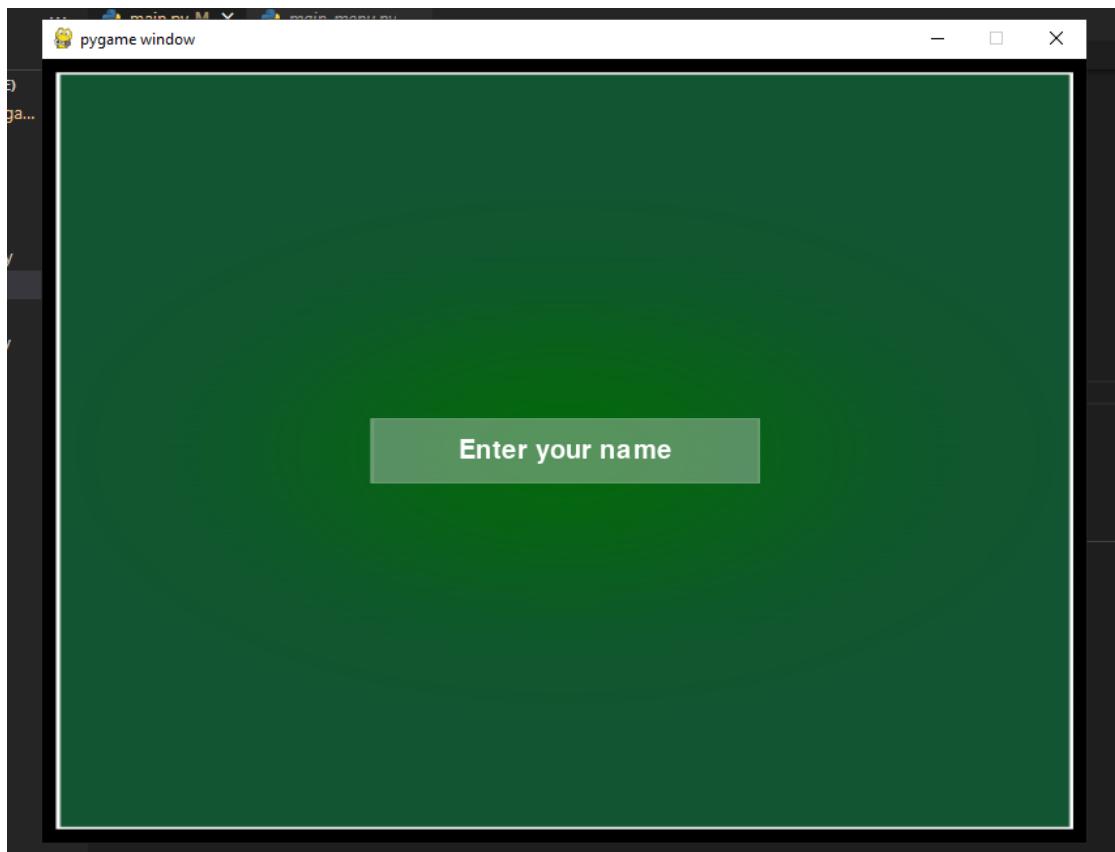
Khi bạn đoán đúng số may mắn, phần thưởng sẽ là số tiền bạn đặt cược nhân với tỉ lệ thưởng của mỗi số được ghi trên màn hình

Join Room: Bạn có thể tìm phòng của các người chơi trong cùng mạng nội bộ của thiết bị, màn hình sẽ hiển thị danh sách các phòng, nhấn chuột để tham gia. Lưu ý: chỉ có thể tham gia phòng khi phòng đang ở trạng thái chờ; khi muốn tìm lại phòng vui lòng back lại màn hình chính và nhấn join room một lần nữa

4.3 Các thành phần xây dựng game Bingo

4.3.1 Giao diện

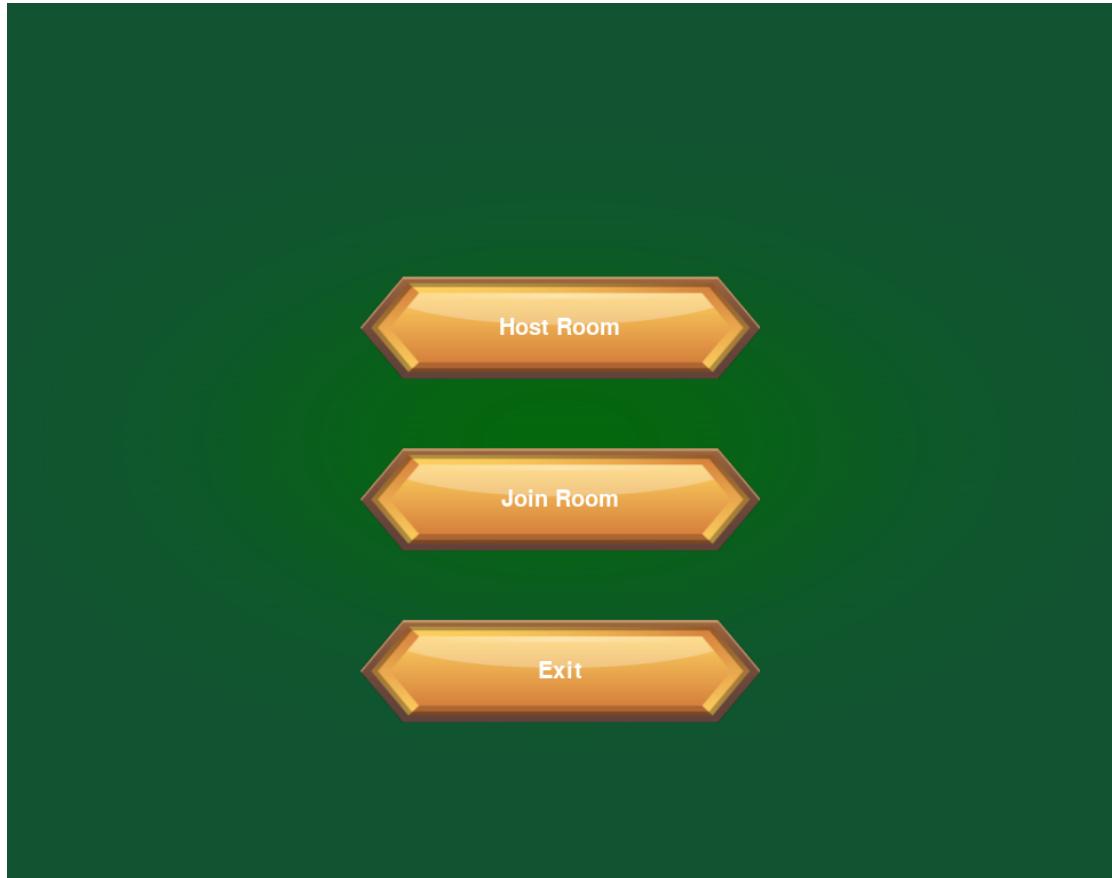
- Giao diện lúc khởi động game sẽ yêu cầu người chơi nhập tên của mình



Hình 2: Giao diện nhập tên



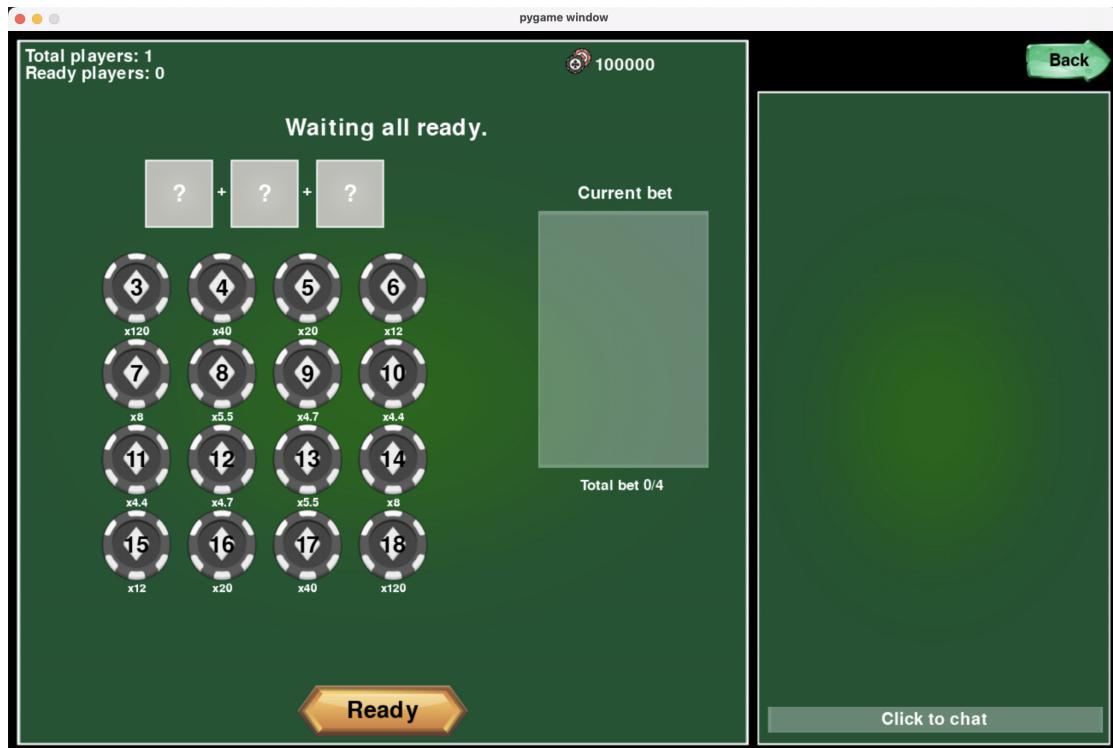
- Giao diện tạo hoặc vào một phòng nào đó



Hình 3: Giao diện lựa chọn



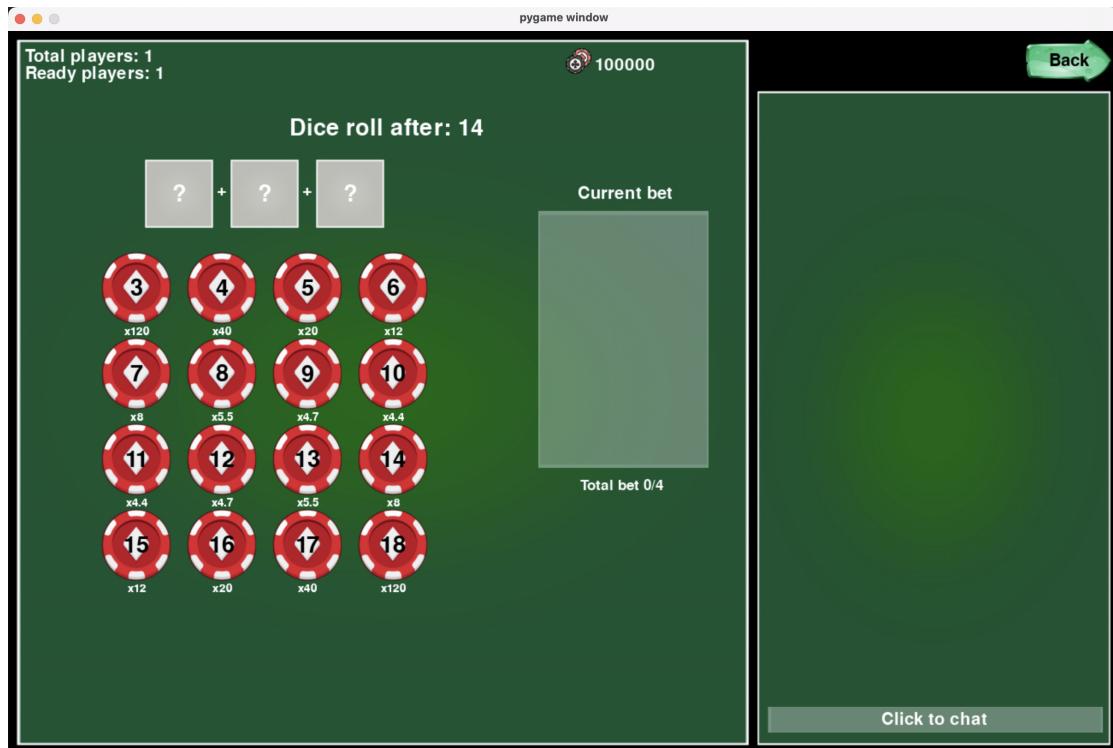
- Giao diện khi mình là chủ room



Hình 4: Giao diện chủ room



- Giao diện khi đặt cược



Hình 5: Giao diện đặt cược

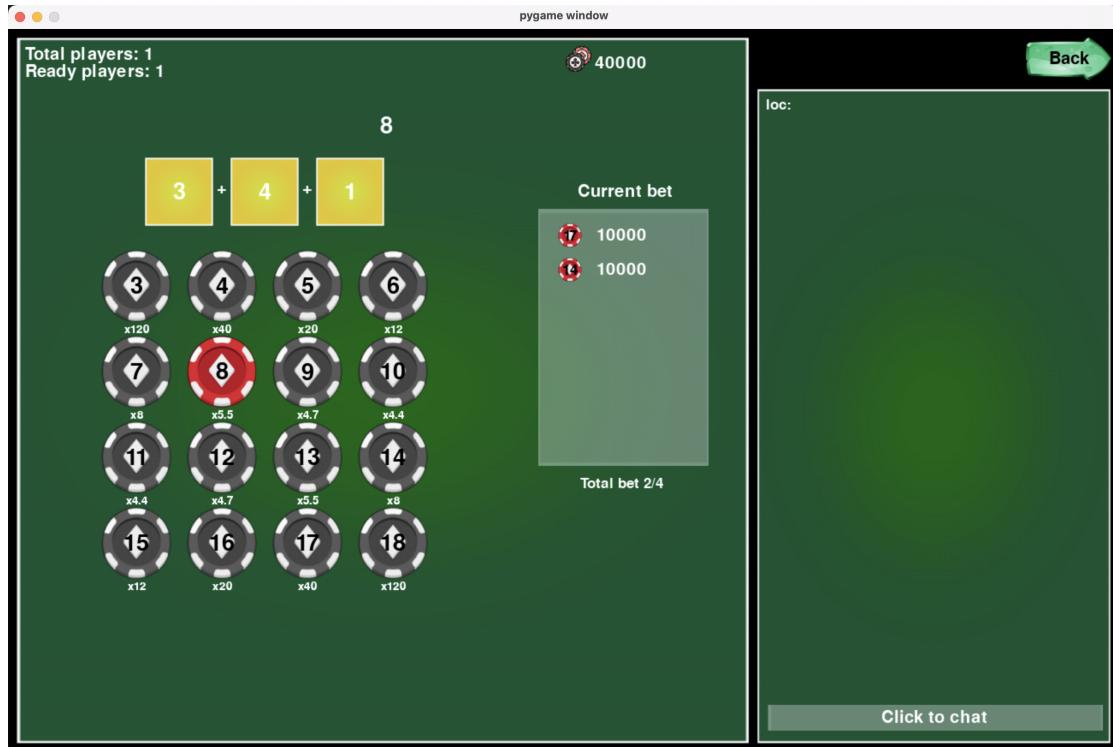


- Giao diện khi nhắn tin



Hình 6: Giao diện nhắn tin

- Giao diện kết quả



Hình 7: Giao diện kết quả



- Giao diện khi mình tìm phòng để do



Hình 8: Giao diện tìm phòng



4.3.2 Các lớp chính

- lớp room chứa các phương thức để tạo hay vào một phòng

```
class Room():
    def __init__(self, host, room_name, player_name, client = None) -> None:
        self.listBtn = []
        self.diceHistory = []
        self.ready = 0
        self.playerCount = 1
        self.txtTitleStr = "Waiting all ready"
        self.readyCountTime = 5
        self.gameEndCountTime = 15
        self.diceRollCountTime = 3
        self.animationTime = 1000
        self.time = 0
        self.user_txt = ""
        self.state = 0
        self.close = False

        self.diceRollTime = self.diceRollCountTime * 1000

        self.messHistory = []

        self.playerReady = False

        self.host = host
        self.client = client

        # current bet curformed
        self.currentConfirm = []
        # self.currentConfirm.append([3, 1000])
```

Hình 9: lớp room



- lớp menu load lên cho người dùng sự lựa chọn là tạo hay vào phòng

```
class MainMenu:  
    def __init__(self, screen_width, screen_height) -> None:  
  
        self.screen_width = screen_width  
        self.screen_height = screen_height  
  
        imgBackground = pygame.image.load('./assets/room_background.png')  
        self.imgBackground = pygame.transform.smoothscale(imgBackground, (screen_width - 20, screen_height - 20))  
  
        imgButton = pygame.image.load('./assets/button2.png')  
        imgButton = pygame.transform.smoothscale(imgButton, (350, 90))  
  
        self.host = Button(imgButton, (screen_width / 2, 300), "Host Room", get_font(30), "white", "black")  
        self.join = Button(imgButton, (screen_width / 2, 450), "Join Room", get_font(30), "white", "black")  
        self.exit = Button(imgButton, (screen_width / 2, 600), "Exit", get_font(30), "white", "black")  
        backImg = pygame.image.load('./assets/back.png')  
        backImg = pygame.transform.smoothscale(backImg, (100, 50))  
        self.btnExit = Button(image=backImg, pos=(screen_width - 60, 35), text_input="Back", font=get_font(30),  
                             color="white", border_color="black")  
  
        # self.imgBorder = pygame.image.load('./assets/chatbox.png')  
        self.state = 0  
        self.playerName = None  
  
        imgBorder = pygame.image.load('./assets/chatbox.png')  
        imgInput = pygame.transform.smoothscale(imgBorder, (300, 50))  
        self.imgRoomSelect = pygame.transform.smoothscale(imgBorder, (200, 30))  
  
        self.inputName = TextInput(imgInput, (screen_width / 2, screen_height / 2), "white", "yellow", "Enter your name",  
#         self.inputName.set_active(True)  
        self.txtEnterToPlay = get_font(30).render("Enter to play", True, "white")  
  
    def update(self, delta, screen):  
        screen.fill("black")
```

Hình 10: lớp menu



- lớp button xử lý load hình ảnh và màu chữ của các nút

```
class Button():      Dak Lod, 2 weeks ago • first init
    def __init__(self, image, pos, text_input, font, base_color, hovering_color):
        self.y_pos = pos[1]
        self.font = font
        self.base_color, self.hovering_color = base_color, hovering_color
        self.text_input = text_input
        self.text = self.font.render(self.text_input, True, self.base_color)
        if self.image is None:
            self.image = self.text
        self.rect = self.image.get_rect(center=(self.x_pos, self.y_pos))
        self.text_rect = self.text.get_rect(center=(self.x_pos, self.y_pos))
        self.current_color = base_color

    def update(self, screen):
        self.rect = self.image.get_rect(center=(self.x_pos, self.y_pos))
        self.text_rect = self.text.get_rect(center=(self.x_pos, self.y_pos))
        if self.image is not None:
            screen.blit(self.image, self.rect)

        self.text = self.font.render(self.text_input, True, self.current_color)
        screen.blit(self.text, self.text_rect)

    def checkForInput(self, position):
        if position[0] in range(self.rect.left, self.rect.right) and position[1] in range(self.rect.top, self.rect.bottom):
            return True
        return False

    def changeColor(self, position):
        if position[0] in range(self.rect.left, self.rect.right) and position[1] in range(self.rect.top, self.rect.bottom):
            self.current_color = self.hovering_color
        else:
            self.current_color = self.base_color
```

Hình 11: lớp button



- lớp TextInput xử lý load các form nhập liệu để chat giữa các người chơi

```
class Button():      Dak Lod, 2 weeks ago • first init
    def __init__(self, image, pos, text_input, font, base_color, hovering_color):
        self.y_pos = pos[1]
        self.font = font
        self.base_color, self.hovering_color = base_color, hovering_color
        self.text_input = text_input
        self.text = self.font.render(self.text_input, True, self.base_color)
        if self.image is None:
            self.image = self.text
        self.rect = self.image.get_rect(center=(self.x_pos, self.y_pos))
        self.text_rect = self.text.get_rect(center=(self.x_pos, self.y_pos))
        self.current_color = base_color

    def update(self, screen):
        self.rect = self.image.get_rect(center=(self.x_pos, self.y_pos))
        self.text_rect = self.text.get_rect(center=(self.x_pos, self.y_pos))
        if self.image is not None:
            screen.blit(self.image, self.rect)

        self.text = self.font.render(self.text_input, True, self.current_color)
        screen.blit(self.text, self.text_rect)

    def checkForInput(self, position):
        if position[0] in range(self.rect.left, self.rect.right) and position[1] in range(self.rect.top, self.rect.bottom):
            return True
        return False

    def changeColor(self, position):
        if position[0] in range(self.rect.left, self.rect.right) and position[1] in range(self.rect.top, self.rect.bottom):
            self.current_color = self.hovering_color
        else:
            self.current_color = self.base_color
```

Hình 12: lớp Textinput



4.3.3 Các hàm xử lý chính

- Xử lý tạo room

```
def server_init(self):
    # create a socket object
    server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    global host, port
    try:
        server.bind((host, port))
        server.listen(10)
        print('listen on port: ' + str(port))
        self.server = server
        self.clients = []
        global t1
        t1 = threading.Thread(target=self.server_add_connect, daemon=True)
        t1.daemon = True
        t1.start()
        self.openPort = True
        self.thread = threading.Thread(target=self.start_broadcast_server, daemon=True)
        self.thread.start()
        return
    except:
        self.openPort = False
        self.openPort = False
```

Hình 13: tạo phòng



- Xử lý tìm phòng

```
def discover_server(self):
    with socket.socket(socket.AF_INET, socket.SOCK_DGRAM, socket.IPPROTO_UDP) as client_socket:
        client_socket.setsockopt(socket.SOL_SOCKET, socket.SO_BROADCAST, 1)
        client_socket.settimeout(5)
        client_socket.sendto(b'discover', ('<broadcast>', 65432))
        try:
            data, server = client_socket.recvfrom(1024)
            print(f"Server discovered at {server}: {data}")
            self.listBtn.append((Button(self.imgRoomSelect, (self.screen_width / 2, 200 + 40 * len(self.serverList)
                - 1), data.decode(), get_font(25), "white", "black"), server))
        except socket.timeout:
            print("No server response")
```

Hình 14: tìm phòng



- Xử lý gửi tin nhắn

```
def server_send(self, type, value = None):
    value = str(value)
    match type:
        case "player_count":
            self.server_broadcast("#player_count/" + value)
        case "state":
            self.server_broadcast("#state/" + value)
        case "ready":
            self.server_broadcast("#ready/" + value)
        case "join":
            self.server_broadcast("#join/" + value)
        case "mess":
            self.server_broadcast("#mess/" + value)
        case "result":
            self.server_broadcast("#result/" + value)
        case "bet":
            self.server_broadcast("#bet/" + pickle.dumps(value))

def server_broadcast(self, string):
    for client in self.clients:
        try:
            client.send(string.encode())
        except:
            continue
```

Hình 15: gửi tin nhắn



- Xử lý nhận tin nhắn

```
def client_rev(self):
    while True:
        try:
            print("client listen")
            # self.client.settimeout(1)
            request = self.client.recv(1024)
            print(request)

            if request is None:
                global screen_id, room
                screen_id = 1
                room = MainMenu(screen_width, screen_height)

            request = request.decode().split("/")
            match request[0]:
                case "#mess":
                    self.messHistory.append(request[1])
                case "#player_count":
                    self.playerCount = int(request[1])
                case "#state":
                    self.initState(int(request[1]))
                case "#ready":
                    self.ready = int(request[1])
                case "#result":
                    result = request[1].split(',')
                    self.dice1.text_input = result[0]
                    self.dice2.text_input = result[1]
                    self.dice3.text_input = result[2]
                    self.txtTitleStr = str(int(result[0]) + int(result[1]) + int(result[2]))
                    self.initState(4)

                case "bet":
                    self.roomBet = pickle.loads(request[1])
            except:
                return
                pass
```

Hình 16: nhận tin nhắn



- Xử lý trả thưởng

```
pass
case 4:

    for btn in self.listBtn:
        btn.image = self.imgBtnNumberDisable

    if self.host:
        result = (self.dice1.text_input, self.dice2.text_input, self.dice3.text_input)
        self.server_send("result", ',' .join(result))
        self.state_time = 5000

    self.listBtn[int(self.txtTitleStr) - 3].image = self.imgBtnNumber

    for bet in self.currentConfirm:
        print(bet)
        global player_money
        if (bet[0] == int(self.txtTitleStr)):
            price = int(bet[1]) * self.price_scale(bet[0])
            player_money += price
            print(price)
```

Hình 17: trả thưởng



5 Kết Luận

5.1 Đánh giá

- Ưu điểm:
 - + Có đầy đủ các hoạt động cơ bản của trò chơi.
 - + Giao diện đơn giản và dễ hiểu.
 - + Dung lượng cài đặt chương trình nhỏ.
- Nhược điểm:
 - + Phải thực hiện đúng trình tự các bước trong hướng dẫn mới có thể chơi. Thực hiện sai có thể dẫn đến đóng băng ứng dụng.
 - + Hiệu năng có thể bị giới hạn.
 - + Giới hạn nhiều người chơi cùng một lúc ít.

5.2 Hướng phát triển

- thêm nhạc nền và các hiệu ứng
- tăng giới hạn người chơi cùng lúc



6 Tài Liệu Tham Khảo

- 1 [Bingo \(American version\) Wikipedia.](#)
- 2 [Python \(ngôn ngữ lập trình\) – Wikipedia tiếng Việt.](#)
- 3 Slide bài giảng của thầy Từ Lãng Phiêu.