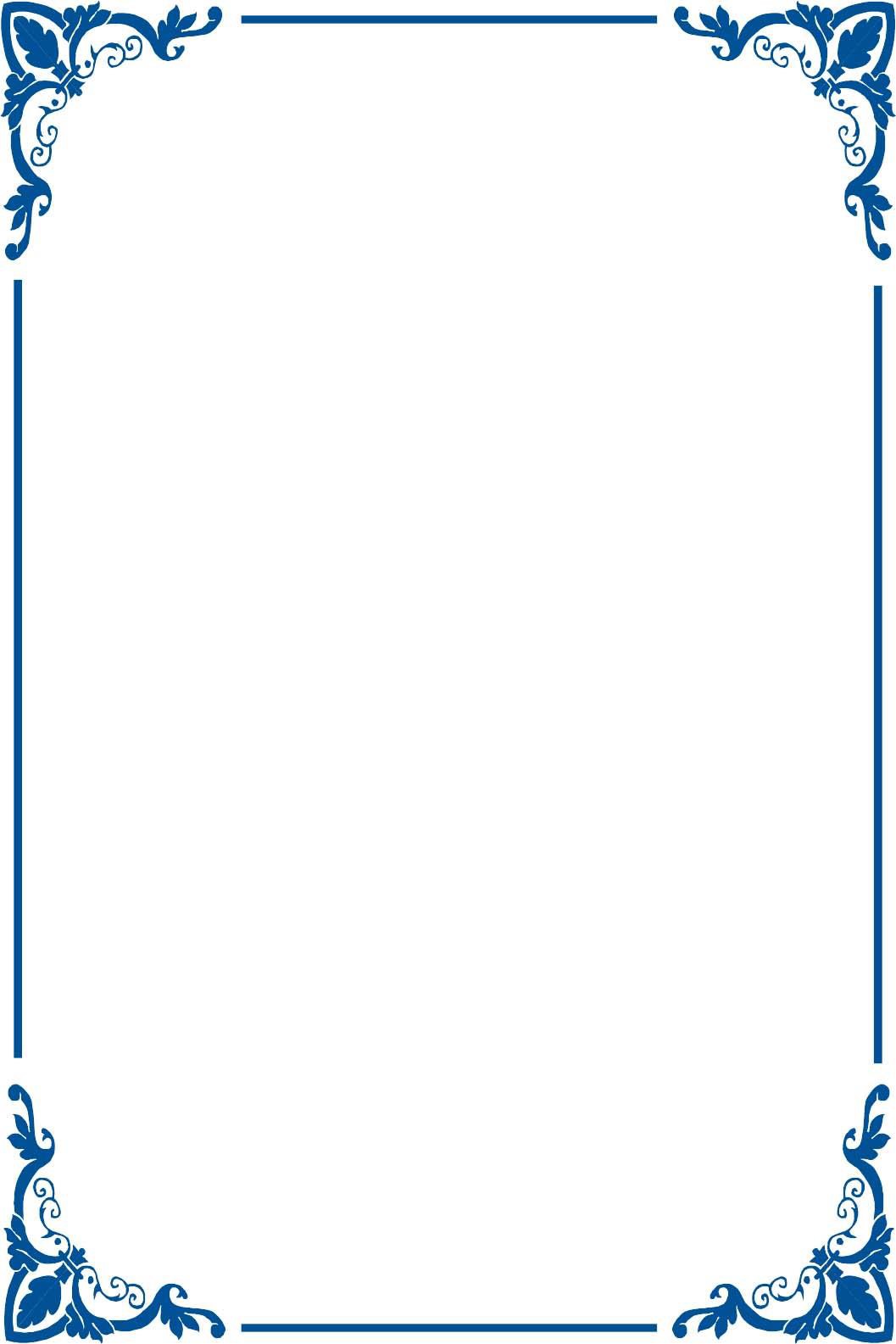
**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HCM KHOA CÔNG NGHỆ THÔNG TIN**





**ĐỒ ÁN CUỐI KỲ**

**MÔN TRÍ TUỆ NHÂN TẠO**

*Đề tài:*

XÂY DỰNG CÁC DỰ ÁN TRÍ TUỆ NHÂN TẠO

**GVHD: ThS.Trần Tiến Đức**

**Mã LHP: 241ARIN330585\_07CLC**

**Sinh viên thực hiện:**

|  |  |  |
| --- | --- | --- |
| **Phan Văn Quân** | **-** | **22110214** |

**Thành phố Hồ Chí Minh Năm 2024**

***Nhận xét của giáo viên***

………………………………………………………………………………………...

………………………………………………………………………………………...

………………………………………………………………………………………...

………………………………………………………………………………………...

………………………………………………………………………………………...

………………………………………………………………………………………..

………………………………………………………………………………………...

………………………………………………………………………………………...

………………………………………………………………………………………...

………………………………………………………………………………………...

………………………………………………………………………………………...

………………………………………………………………………………………..

………………………………………………………………………………………...

………………………………………………………………………………………...

………………………………………………………………………………………...

………………………………………………………………………………………...

………………………………………………………………………………………...

………………………………………………………………………………………..

*Ngày 1 tháng 12 năm 2024*

**KẾ HOẠCH THỰC HIỆN**

|  |  |
| --- | --- |
| **Tuần** | **Công việc** |
| 13 | Tìm đề tài  Lên ý tưởng  Tìm hiểu kiến thức về Simpleai, AIMA và Streamlit, các thuật toán |
| 14 | Viết source code  Tiến hành triển khai console code với GUI Tkinte |
| 15 | Viết source code  Triển khai code trên Streamlit |
| 16 | Viết source code  Kiểm thử và chỉnh sửa lỗi |

Mục Lục

[PHẦN MỞ ĐẦU 1](#_Toc183982358)

[1. Lời cảm ơn 1](#_Toc183982359)

[2. Lời mở đầu 1](#_Toc183982360)

[3. Các thư viện sử dụng 1](#_Toc183982361)

[CHƯƠNG 1: CHƯƠNG TRÌNH TÌM KIẾM ĐƯỜNG ĐI TRONG MÊ CUNG 2](#_Toc183982362)

[1.1 Xây dựng chương trình tìm đường trong mê cung với giao diện bằng Tkinter 2](#_Toc183982363)

[1.2 Triển khai chương trinh tìm đường trong mê cung bằng Streamlit 3](#_Toc183982364)

[CHƯƠNG 2: CHƯƠNG TRÌNH LẬP LỊCH THỜI KHÓA BIỂU 6](#_Toc183982365)

[2.1 Xây dựng chương trình lập lịch thời khóa biểu với console 6](#_Toc183982366)

[2.2 Triển khai chương trình lập lịch thời khóa biểu bằng Streamlit 7](#_Toc183982367)

[CHƯƠNG 3: CHƯƠNG TRÌNH GIẢI QUYẾT BÀI TOÁN 8-PUZZLE 9](#_Toc183982368)

[3.1 Xây dựng chương trình 8\_Puzzle với giao diện bằng Tkinter 9](#_Toc183982369)

[3.2 Xây dựng chương trình 8\_Puzzle với giao diện bằng Tkinter 13](#_Toc183982370)

[KẾT LUẬN 15](#_Toc183982371)

# PHẦN MỞ ĐẦU

## Lời cảm ơn

Trước tiên, chúng em xin gửi lời cảm ơn chân thành đến thầy Trần Tiến Đức vì đã tận tâm giảng dạy và truyền đạt những kiến thức quý báu về Trí tuệ nhân tạo trong suốt học kỳ vừa qua. Những kiến thức mà thầy mang lại không chỉ giúp chúng em hiểu rõ lý thuyết mà còn ứng dụng vào thực. Chúng em cũng xin cảm ơn nhà trường đã tạo điều kiện học tập, cung cấp tài liệu, cơ sở vật chất, và môi trường học tập thuận lợi để chúng em có thể hoàn thành tốt môn học này.Cuối cùng, chúng em xin bày tỏ lòng biết ơn đến các bạn và thầy đã luôn đồng hành, động viên, và hỗ trợ em trong quá trình thực hiện bài báo cáo này, qua đó khơi dậy niềm đam mê và động lực học tập trong lĩnh vực này.

## Lời mở đầu

Trong thời đại cách mạng công nghệ 4.0, Trí tuệ nhân tạo (AI) đã và đang trở thành một trong những lĩnh vực then chốt, tác động sâu sắc đến mọi khía cạnh của đời sống, từ kinh tế, giáo dục, y tế cho đến công nghiệp và giải trí. Với khả năng mô phỏng trí tuệ con người, AI mang đến những giải pháp tối ưu, giúp cải thiện hiệu suất và mở ra những cơ hội mới cho sự phát triển bền vững.

Môn học Trí tuệ nhân tạo không chỉ cung cấp cho chúng em những kiến thức nền tảng về thuật toán, hệ chuyên gia, học máy và xử lý dữ liệu, mà còn khơi dậy tư duy sáng tạo và khả năng áp dụng vào thực tế. Trong bài báo cáo này, chúng em sẽ trình bày những nội dung đã học, cùng với kết quả nghiên cứu và ứng dụng trong một bài toán thực tiễn cụ thể.

## Các thư viện sử dụng

Chương trình đường thực hiện bằng một số thư viện hỗ trợ như: Matplotlib, numpy, PIL, math, simpleai, streamlit, streamlit\_drawable\_canvas, cv2, tkinter, time, os, constraint, heapq, queue, ramdom,…

# CHƯƠNG 1: CHƯƠNG TRÌNH TÌM KIẾM ĐƯỜNG ĐI TRONG MÊ CUNG

### Xây dựng chương trình tìm đường trong mê cung với giao diện bằng Tkinter

Chương trình tìm đường đi trong mê cung là một chương trình sử dụng thuật toán A\* giúp người dùng có thể tìm thấy đường đi tối ưu nhất từ điểm bắt đầu tới điểm kết thúc trong mê cung.

Chương trình được triển khai bằng cách xây dựng mê cung từ ảnh cung cấp, dựa vào môt số thư việc có sẵn, chương trình sẽ thực hiện đọc ảnh (mecung.png) và chuyển đổi ảnh sang mê cung dạng nhị phân và hiện thị nó dưới dạng 2D

image = cv2.imread(os.path.join(os.getcwd(), "assets\mecung.png"))

gray = cv2.cvtColor(image, cv2.COLOR\_BGR2GRAY)

\_, thresh = cv2.threshold(gray, 128, 255, cv2.THRESH\_BINARY)

MAP = []

for row in thresh:

    MAP.append([' ' if pixel == 255 else '#' for pixel in row])

for row in MAP:

    print(''.join(row))

Sau khi hình ảnh được chuyển đổi thành dạng nhị phân, sau đó chương trình sẽ triển khai và xây dựng đồ họa dựa vào thư viện Tkinter

self.title('Tìm đường trong mê cung')

        self.cvs\_me\_cung = tk.Canvas(self, width=N \* W, height=M \* W,

                                      relief=tk.SUNKEN, border=1)

        image\_tk = ImageTk.PhotoImage(pil\_image\_resized)

        self.image\_tk = ImageTk.PhotoImage(pil\_image)

        self.cvs\_me\_cung.create\_image(0, 0, anchor=tk.NW, image=self.image\_tk)

        self.cvs\_me\_cung.bind("<Button-1>", self.xu\_ly\_mouse)

        lbl\_frm\_menu = tk.LabelFrame(self)

        btn\_start = tk.Button(lbl\_frm\_menu, text='Start', width=7,

                              command=self.btn\_start\_click)

        btn\_reset = tk.Button(lbl\_frm\_menu, text='Reset', width=7,

                              command=self.btn\_reset\_click)

        btn\_start.grid(row=0, column=0, padx=5, pady=5, sticky=tk.N)

        btn\_reset.grid(row=1, column=0, padx=5, pady=5, sticky=tk.N

        self.cvs\_me\_cung.grid(row=0, column=0, padx=5, pady=5)

        lbl\_frm\_menu.grid(row=0, column=1, padx=5, pady=7, sticky=tk.NW)

Thuật toán A\* sử dụng hàm Heuristic để ước tính khoảng cách từ trạng thái hiện tại đến trạng thái đích và công thức khoảng cách Euclid.

def heuristic(self, state):

        x, y = state

        gx, gy = self.goal

        return math.sqrt((x - gx) \*\* 2 + (y - gy) \*\* 2)

Thuật toán sẽ ước lượng chi phí dựa trên chi phí cố định mà chung ta khởi tạo ban đầu

COSTS = {

    "up": cost\_regular,

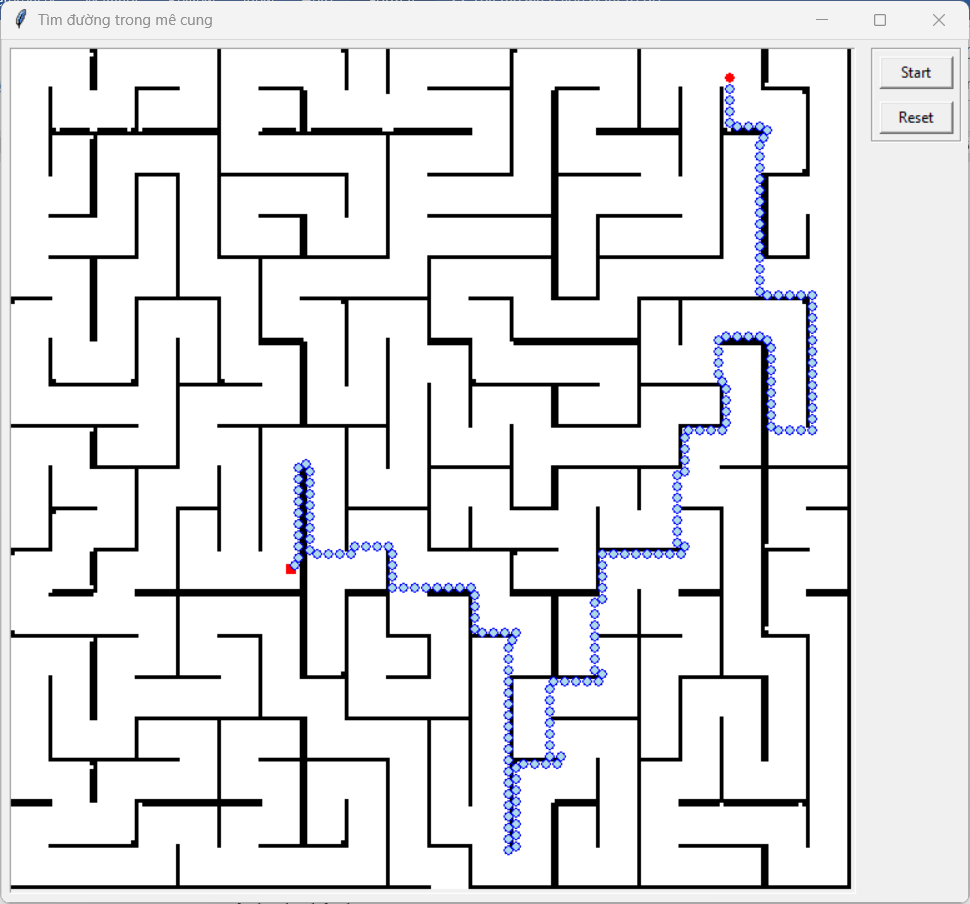
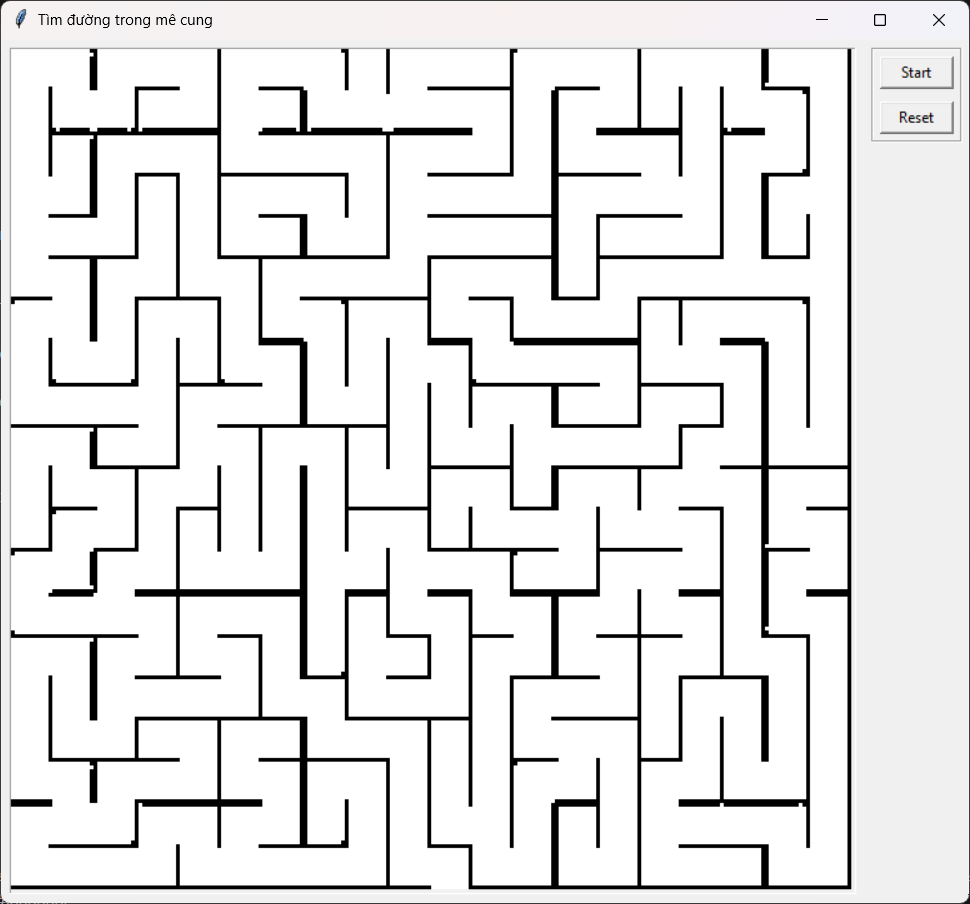
    "down": cost\_regular,

    "left": cost\_regular,

    "right": cost\_regular

}

Tkinter sẽ xây dựng và tạo các nút, người dùng sẽ thực hiện chọn điểm bắt đầu và điểm kết thúc trên mê cung và tiến hành nhấn nút Start, sau đó chương trình sẽ thực hiện tính toán dựa trên thuật toán A\* và hiện thị kết quả ra màn hình



Hình 1 Chương trình tìm kiếm đường đi bằng Tkinter

Người dùng có thể thực hiện lại bằng cách nhấn nút Reset và bắt đầu lại với điểm bắt đầu và kết thúc mới

### Triển khai chương trinh tìm đường trong mê cung bằng Streamlit

Bằng cách sử dụng thư viện Streamlit, chương trình sẽ sử dụng thuật toán A\* để tìm kiếm đường đi và triển khai với giao diện đồ họ bằng Streamlit.

Thư viện streamlit sẽ giúp chúng ta tạo các nút giúp hỗ trợ trong quá trình tương tác với mê cung

st.title('Tìm đường trong mê cung')

if col2.button('Đường dẫn'):

        if "path\_finding\_done" not in st.session\_state:

            st.session\_state["path\_finding\_done"] = True

            x1, y1 = st.session\_state["points"][0]

            x2, y2 = st.session\_state["points"][1]

            maze\_map[y1][x1] = 'o'

            maze\_map[y2][x2] = 'x'

            problem = MazeSolver(maze\_map)

            result = astar(problem, graph\_search=True)

            path = [x[1] for x in result.path()]

            st.session\_state["path"] = path

            draw\_frame = st.session\_state["bg\_image"].copy()

            for p in path:

                x, y = p

                cell\_x = x \* cell\_size + 2

                cell\_y = y \* cell\_size + 2

                frame\_temp = ImageDraw.Draw(draw\_frame)

                frame\_temp.ellipse([cell\_x, cell\_y, cell\_x + cell\_size - 4, cell\_y + cell\_size - 4], fill="#AEC6CF", outline="#AEC6CF")

            st.session\_state["bg\_image"] = draw\_frame

            st.rerun()

    if col4.button('Chuyển động'):

        if "path\_finding\_done" in st.session\_state:

            path = st.session\_state["path"]

            n = len(path)

            px = []

            py = []

            for p in path:

                px.append(p[0] \* cell\_size + cell\_size // 2 - 2)

                py.append(p[1] \* cell\_size + cell\_size // 2 - 2)

            im = plt.imread("assets/mecung.bmp")

            fig, ax = plt.subplots()

            image = ax.imshow(im)

            dest, = ax.plot(px[-1:], py[-1:], "ro", markersize=10)

            red\_square, = ax.plot([], [], "ro", markersize=10)

            def init():

                return image, dest, red\_square

            def animate(i):

                red\_square.set\_data(px[:i + 1], py[:i + 1])

                return image, dest, red\_square

            anim = FuncAnimation(fig, animate, frames=n, interval=500, init\_func=init, repeat=False, blit=True)

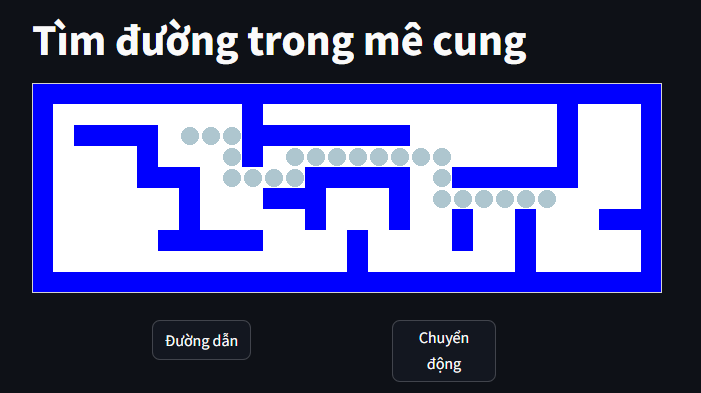
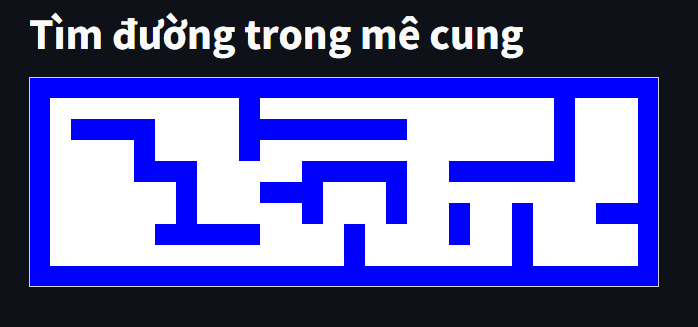
            components.html(anim.to\_jshtml(), height=550)

    \_, col2, \_ = st.columns(3)

    col2.text('Nhấn Ctrl-R để chạy lại')

Chương trình sẽ được chạy bằng lệnh: ***streamlit run findmyway\_streamlit.py***

Người dùng cũng sẽ tiến hành chọn điểm bắt đầu và kết thúc, sau đó chương trình sẽ thực hiện tính toán và đưa ra kết quả



Hình 2 Tìm kiếm đường đi trong mê cung bằng Streamlit

# CHƯƠNG 2: CHƯƠNG TRÌNH LẬP LỊCH THỜI KHÓA BIỂU

### Xây dựng chương trình lập lịch thời khóa biểu với console

Chương trình lập lịch thời khóa biểu hoạt động dựa trên thuật toán Backtracking(Quay lui) để phân tích, tính toán và phân bổ thời khóa biểu của một trường học. Thuật toán được hỗ trợ bởi thư viện constraint để quản lí các biến, miền giá trị và kiểm tra ràng buộc một cách hiểu quả.

Chương trình thực hiện nhóm các môn học lại và chia các khoảng thời gian thành các miền giá trị để dễ dàng tính toán và sắp xếp.

bien\_ma\_mon\_hoc = list(set([

    item for sublist in [P\_6352, P\_7084, P\_9079, P\_9831, P\_0623, P\_3995, P\_2151, P\_0562,

                         P\_3984, P\_0309, P\_7094, P\_2148, P\_9153, P\_1352, P\_6452, P\_9999,

                         P\_6920, P\_0561, P\_9983] for item in sublist

]))

morning\_slots = list(range(1, 25))

afternoon\_slots = list(range(25, 49))

evening\_slots = list(range(49, 73))

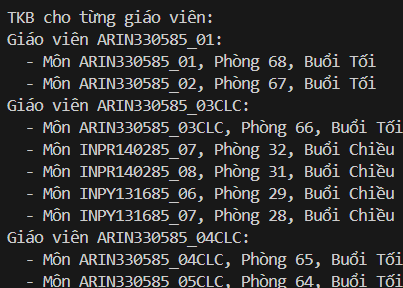
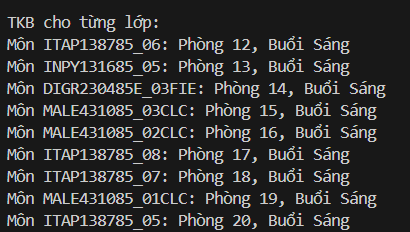
Chương trình sử dụng các ràng buộc nhưu AllDifferentConstraint() và kiem\_tra\_trung để đảm bảo không có sự trùng lặp trong giờ học và các môn học

for group in [P\_6352, P\_7084, P\_9079, P\_9831, P\_0623, P\_3995, P\_2151, P\_0562, P\_3984,

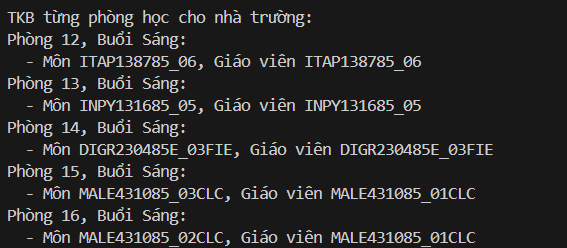
              P\_0309, P\_7094, P\_2148, P\_9153, P\_1352, P\_6452, P\_9999, P\_6920, P\_0561, P\_9983]:

    sch\_problem.addConstraint(FunctionConstraint(kiem\_tra\_trung), group)

Sau khi chạy chương trình, sẽ thu được kết quả thời khóa biểu theo giáo viên, lớp và phòng học

****

Hình 3 Lập lịch thời khóa biểu



Chương trình có thể cải tiến bằng cách cho dữ liệu đầu vào là một dataset(.csv) và tiến hành đọc file, lập lịch và đưa ra kết quả.

### Triển khai chương trình lập lịch thời khóa biểu bằng Streamlit

Chương trình sẽ sử dụng thư viện streamlit để tiến hành triển khai thuật toán bằng giao diện Streamlit  
Chương trình được chạy bằng lệnh: streamlit run ScheduleTKB\_streamlit.py

Và thu được kết quả





Hình 4 Lập lịch thời khóa biểu với streamlit

# CHƯƠNG 3: CHƯƠNG TRÌNH GIẢI QUYẾT BÀI TOÁN 8-PUZZLE

### Xây dựng chương trình 8\_Puzzle với giao diện bằng Tkinter

Chương trình là một tập hợp các thuật toán tìm kiếm trong trí tuệ nhân tạo, được áp dụng để giải quyết bìa toán **8-Puzzle Problem**. **8-Puzzle Problem**  là một trò chơi trượt các khối vuông nhỏ để sắp xếp các ô theo thứ tự đúng

Chương trình sẽ sử dụng cách thuật toán tìm kiếm như: BFS, DFS, UCS, IDFS,Hill Climbing, Greedy, A\*,… để đưa ra giải pháp để giải quyết từ trạng thái ban đầu mà người dùng nhập vào.

Chương trình được xây dựng giao diện bằng Tkinter giúp người dùng thực hiện tương tác, từ đó đưa ra kết quả.

Các hàm hỗ trợ được xây dựng để lấy kết quả từ các thuật toán như:

* getStringRepresentation(x): Chuyển đổi trạng thái từ dạng số nguyên sang chuỗi để dễ dàng xử lý.

def getStringRepresentation(x):

    if int(math.log10(x)) + 1 == 9:

        return str(x)

    else:

        return "0" + str(x)

* getChildren(state): Sinh ra các trạng thái con hợp lệ từ trạng thái hiện tại.

def getChildren(state):

    children = []

    idx = state.index('0')

    i = int(idx / 3)

    j = int(idx % 3)

    for x in range(0, 4):

        nx = i + dx[x]

        ny = j + dy[x]

        nwIdx = int(nx \* 3 + ny)

        if checkValid(nx, ny):

            listTemp = list(state)

            listTemp[idx], listTemp[nwIdx] = listTemp[nwIdx], listTemp[idx]

            children.append(''.join(listTemp))

    return children

* getPath(parentMap, inputState): Lấy đường đi từ trạng thái bắt đầu đến trạng thái mục tiêu bằng cách truy ngược từ cây cha.

def getPath(parentMap, inputState):

    path = []

    temp = 123654780

    while temp != inputState:

        path.append(temp)

        temp = parentMap[temp]

    path.append(inputState)

    path.reverse()

    return path

* printPath(path): In ra đường đi từ trạng thái đầu đến trạng thái mục tiêu.

def printPath(path):

    for i in path:

        print(getStringRepresentation(i))

* goalTest(state): Kiểm tra xem trạng thái hiện tại có phải trạng thái mục tiêu hay không.

def goalTest(state):

    if state == 123654780:

        return True

    return False

* isSolvable(digit): Kiểm tra xem trạng thái đầu vào có thể giải được hay không (dựa vào số nghịch thế).

def isSolvable(digit):

    count = 0

    for i in range(0, 9):

        for j in range(i, 9):

            if digit[i] > digit[j] and digit[i] != 9:

                count += 1

    return count % 2 == 0

Một số hàm heuristic được xây dựng giúp tính toán khoảng cách:

* getManhattanDistance(state): Tính tổng khoảng cách Manhattan giữa các ô số với vị trí mục tiêu.

def getManhattanDistance(state):

    tot = 0

    for i in range(1, 9):

        goalX = int(i / 3)

        goalY = i % 3

        idx = state.index(str(i))

        itemX = int(idx / 3)

        itemY = idx % 3

        tot += (abs(goalX - itemX) + abs(goalY - itemY))

    return tot

* getEuclideanDistance(state): Tính tổng khoảng cách Euclidean giữa các ô số với vị trí mục tiêu.

def getEuclideanDistance(state):

    tot = 0

    for i in range(1, 9):

        goalX = int(i / 3)

        goalY = i % 3

        idx = state.index(str(i))

        itemX = int(idx / 3)

        itemY = idx % 3

        tot += math.sqrt(pow((goalX - itemX), 2) + pow((goalY - itemY), 2))

    return tot

* getHeuristics(state): Một heuristic đơn giản để tính số lượng vị trí sai lệch so với trạng thái mục tiêu.

def getHeuristics(state):

    goal\_state = "123654780"

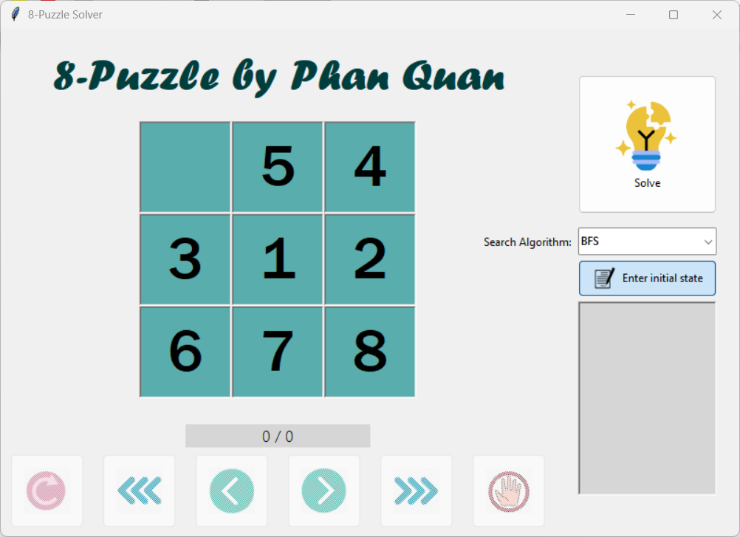
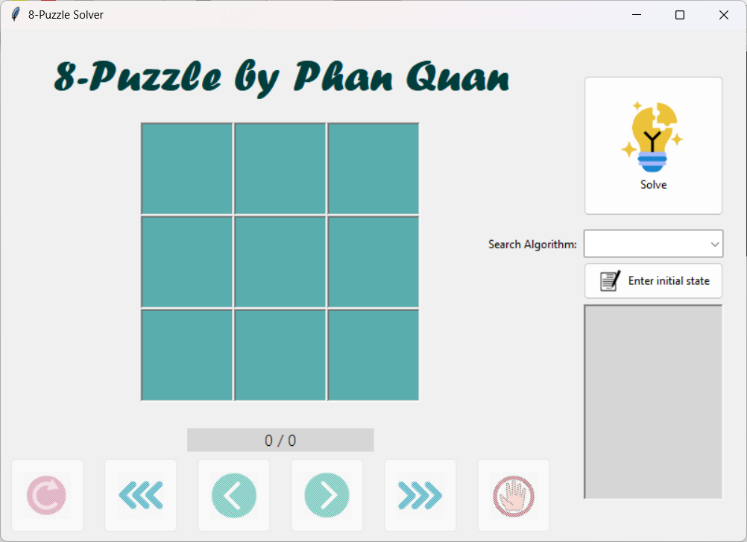
    if len(state) != len(goal\_state):

        raise ValueError("States must have the same length")

    diff\_count = sum(1 for i, j in zip(state, goal\_state) if i != j and i != '0')

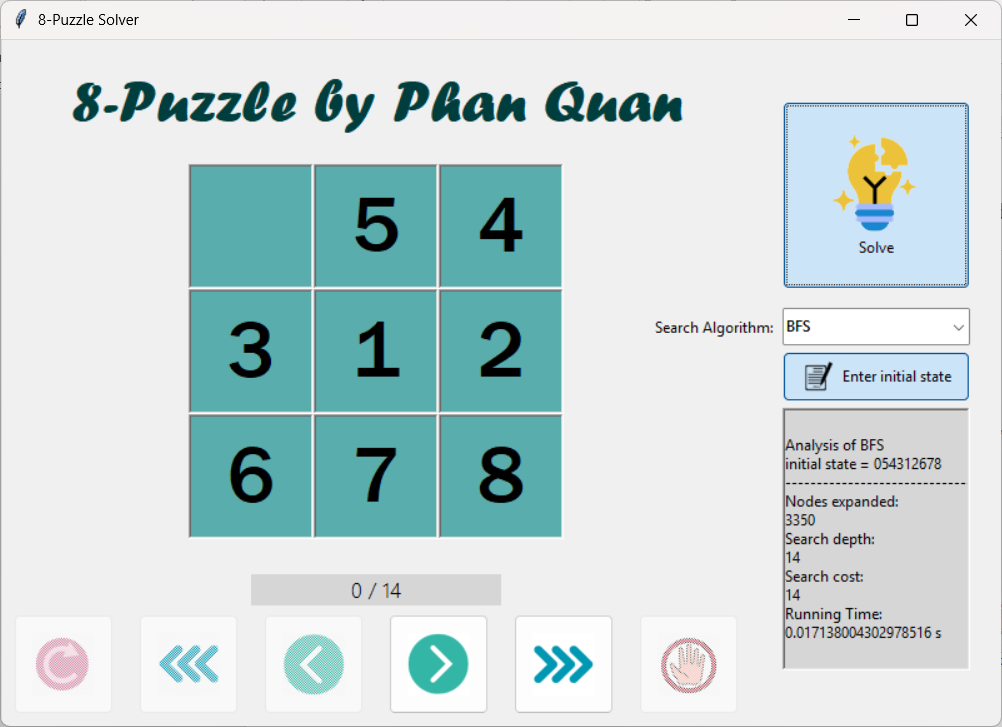
    return diff\_count

Sau khi chạy chương trình, giao diện sẽ được hiện ra, người chơi sẽ thực hiện chọn thuật toán và đưa trạng thái đầu vào



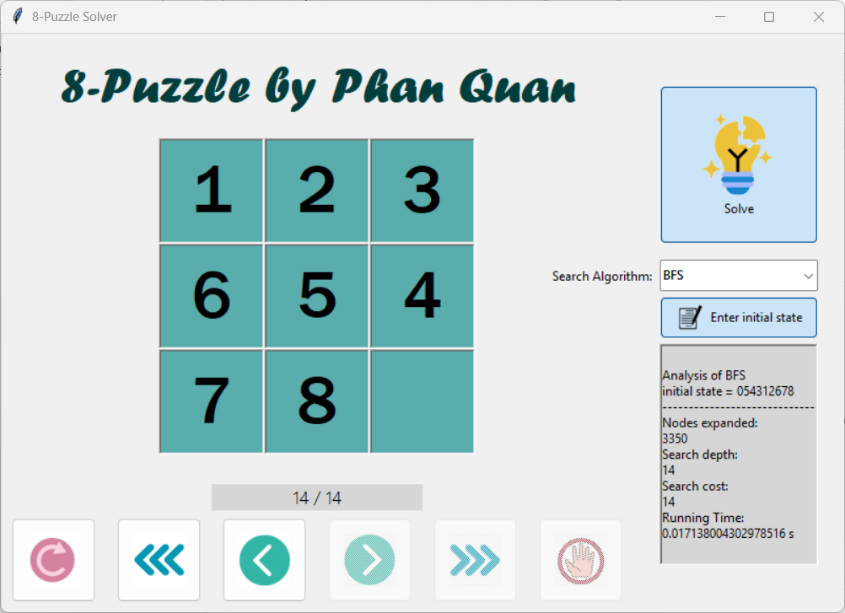
Hình 5 Giao diện 8\_Puzzle

Người chơi sau đó sẽ nhấn nút Solve và sau đó chương trình sẽ thực hiện tính toán và đưa ra giải pháp



Hình 6 Giải pháp thực hiện giải quyết 8\_Puzzle

Sau khi chương trình được thực hiện và đưa ra kết quả, người chơi có thể tương tác bằng cách click vào các nút để có thể xem được các bước thực hiện giải quyết bài toán



Hình 7 Kết quả 8\_Puzzle

Chương trình cũng sẽ phân tích và đưa ra các chỉ số của bài toán như: số nút mở rộng, độ sâu thuật toán, chi phí và thời gian giải quyết bài toán.

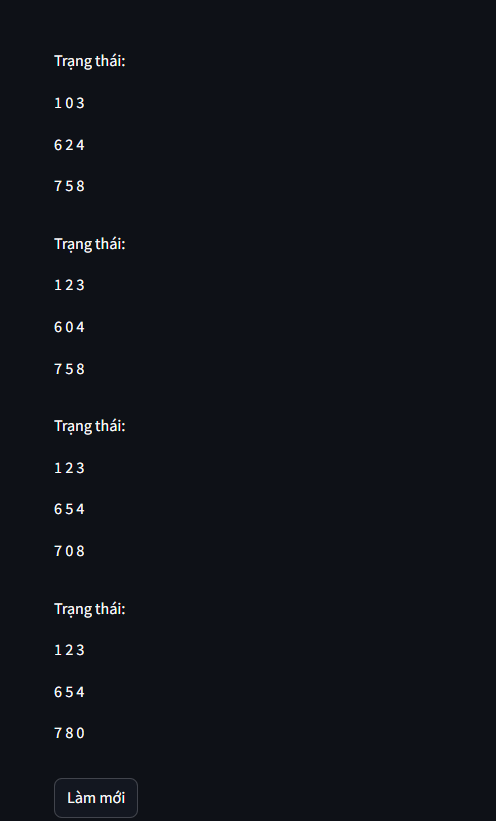
### Xây dựng chương trình 8\_Puzzle với giao diện bằng Tkinter

Chương trình sẽ sử dụng thư viện streamlit để xây dựng đồ họa cho chương trình. Và sẽ chạy chương trình bằng lênh: streamlit run interface\_streamlit.py

Người chơi sẽ chọn thuật toán và đưa ra kết qua dựa vào trạng thái ban đầu mà người dùng cung cấp:



Hình 8 8\_Puzzle bằng Streamlit



Hình 9 Kết quả 8\_Puzzle trên Streamlit

Các trạng thái sẽ được thể hiện dưới dạng một ma trận 3x3.

# KẾT LUẬN

Trong học kì qua, chúng em đã được sự hướng dẫn tận tâm và học hỏi được nhiều kiến thức vô cùng bổ ích từ thầy Trần Tiến Đức. Với lòng biết ơn sâu sắc, chúng em xin gửi đến Thầy những tình cảm trân quý nhất. Chúc Thầy sẽ luôn thành công và tâm huyết với nghề để tiếp tục truyền đạt những tri thức, kinh nghiệm quý báu cho sinh viên chúng em trong quá trình học tập ở trường để chúng em có đủ kiến thức và hành trang bước vào tương lai.

Do giới hạn về thời gian nghiên cứu cũng như lượng kiến thức, thông tin thu thập còn hạn chế nên dự án không tránh khỏi những thiếu sót trong quá trình tìm hiểu và thực hiện. Vì vậy, nhóm chúng em rất mong nhận được sự góp ý và đánh giá tích cực của Thầy để bài báo cáo hoàn thiện hơn.