Môn: **Nhập môn Trí tuệ nhân tạo**
Nhóm: **6**
Giảng viên: **Nguyễn Thành An**

Mã nhóm: **PBL**

#	Họ và tên	MSSV	Email
1	Võ Phước Thịnh	52000807	
2	Nguyễn Trung Nghĩa	52000693	nguyentrungnghiacmp528@gmail.com
3	Hồ Thịnh Phát	52000821	hothinhphatvd4@gmail.com
4	Trần Đình Phúc	52000109	phuctran12421@gmail.com

BẢNG PHÂN CÔNG CÔNG VIỆC

#	Công việc	Người thực hiện	Đánh giá
1	Cài đặt lớp SingleFoodSearchProblem	Phúc	Hoàn thành
2	Cài đặt các cấu trúc dữ liệu	Nghĩa	Hoàn thành
3	Breadth-first search	Phát	Hoàn thành
4	Depth-first search	Nghĩa	Hoàn thành
5	Uniform Cost Search	Phúc	Hoàn thành
6	Cài đặt mini animate	Phúc	Hoàn thành
7	Tổng quát lên MultiFoodSearch	Phát	Hoàn thành
8	Heurictis cho Single và Multiple	Nghĩa	Hoàn thành
9	A*, GBFS	Nghĩa	Hoàn thành

BẢNG PHÂN CÔNG CÔNG VIỆC

#	Công việc	Người thực hiện	Đánh giá
10	Tổng quát hoá A* cho cả hai vấn đề Single và Multiple	Phúc	Hoàn thành
11	Clean code	Phúc	Hoàn thành
12	Cài đặt lớp EightQueenProblem và hill_climbing_search	Nghĩa	Hoàn thành
13	Testing	Phát	Hoàn thành
14	Viết chương trình chạy yêu cầu	Phúc	Hoàn thành
15	Viết bản trình bày chung	*	

ĐÁNH GIÁ MỨC ĐỘ HOÀN THÀNH

#	Họ và tên	MSSV	% Đánh giá
1	Võ Phước Thịnh	52000807	0
2	Nguyễn Trung Nghĩa	52000693	100
3	Hồ Thịnh Phát	52000821	100
4	Trần Đình Phúc	52000109	100

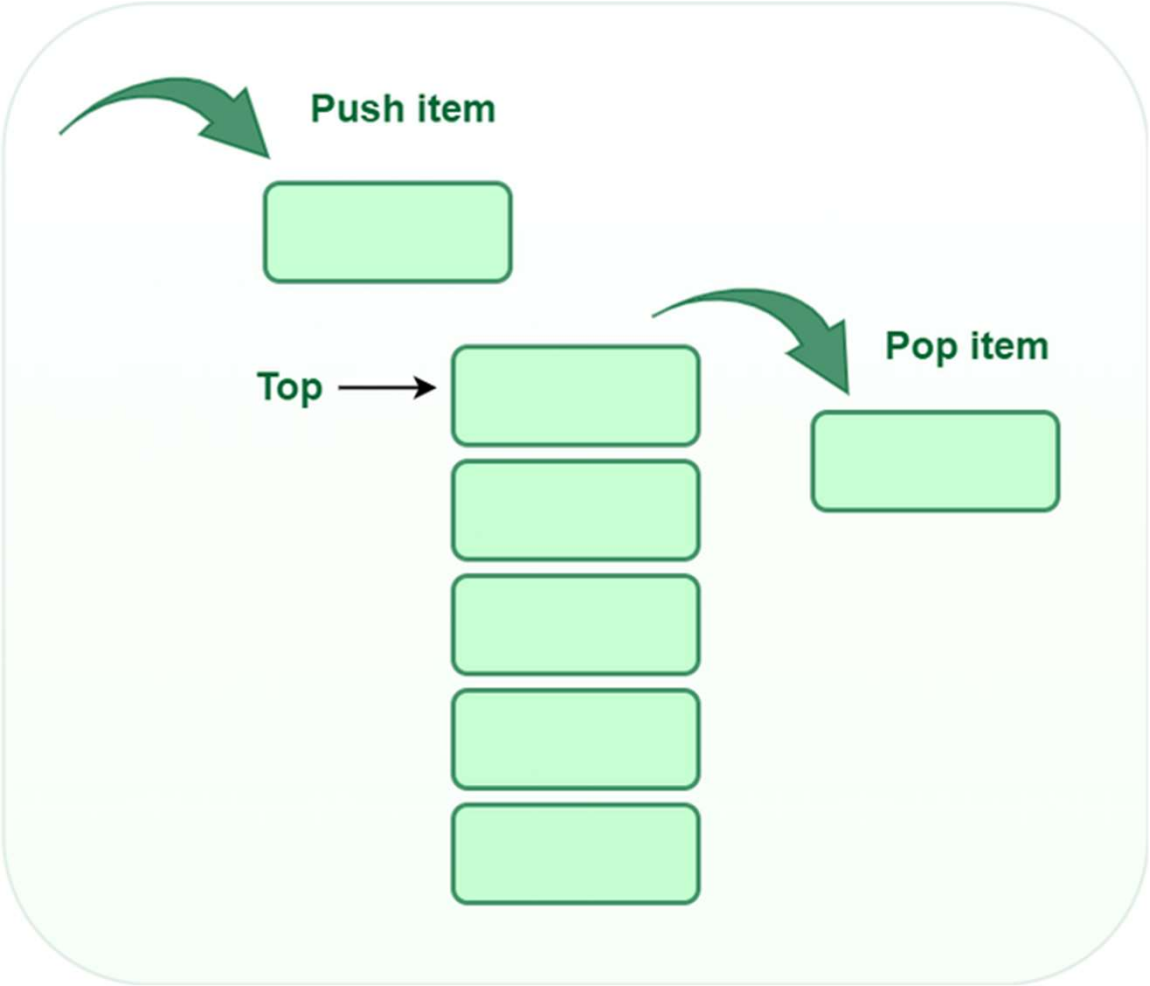
Workspace: https://github.com/trngbro/AI (Private during deadline time)

Yêu cầu 1-1

```python
class SingleFoodSearchProblem:

    def __init__(self):
        self.map = []   #map
        self.P = []     #Initial state
        self.G = []     #Goal
        self.state = []

    def isGoal(self, state):
        return boolean

    def successor(self,state):
        return child

    def load_from_file(self, filename) -> None:
        pass

    def __str__(self) -> str:
        print(something)

    def animate(self, actions) -> None:
        pass

```

Cài đặt lớp SingleFoodSearchProblem

Yêu cầu 1-2




```
You, 2 weeks ago | 1 author (You)
class Stack:
    def __init__(self):
        self.items = []

    def push(self, this):
        self.items.append(this)

    def pop(self):
        if self.items:
            return self.items.pop()
        else:
            return None

    def size(self):
        return len(self.items)

    def isEmpty(self):
        return len(self.items)==0

    def contain(self, this):
        return this in self.items

    def peek(self):
        return self.items[-1]
```

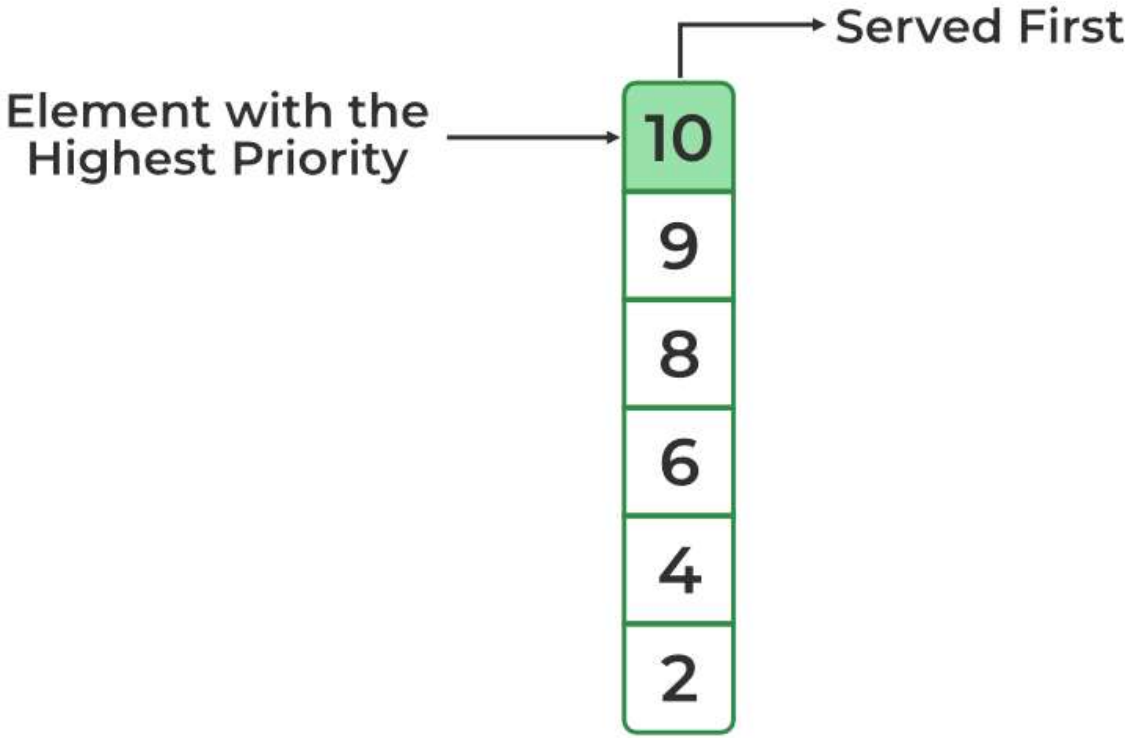
Stack workflow and implements

Yêu cầu 1-2


```
Phuc, 2 days ago | 2 authors (You and others)
class Queue:
    def __init__(self):
        self.items = []

    def isEmpty(self):
        return len(self.items) == 0

    def enqueue(self, this):
        self.items.insert(0, this)

    def dequeue(self):
        return self.items.pop()

    def contain(self, this):
        return this in self.items

    def size(self):
        return len(self.items)
```

Queue workflow and implements

Yêu cầu 1-2



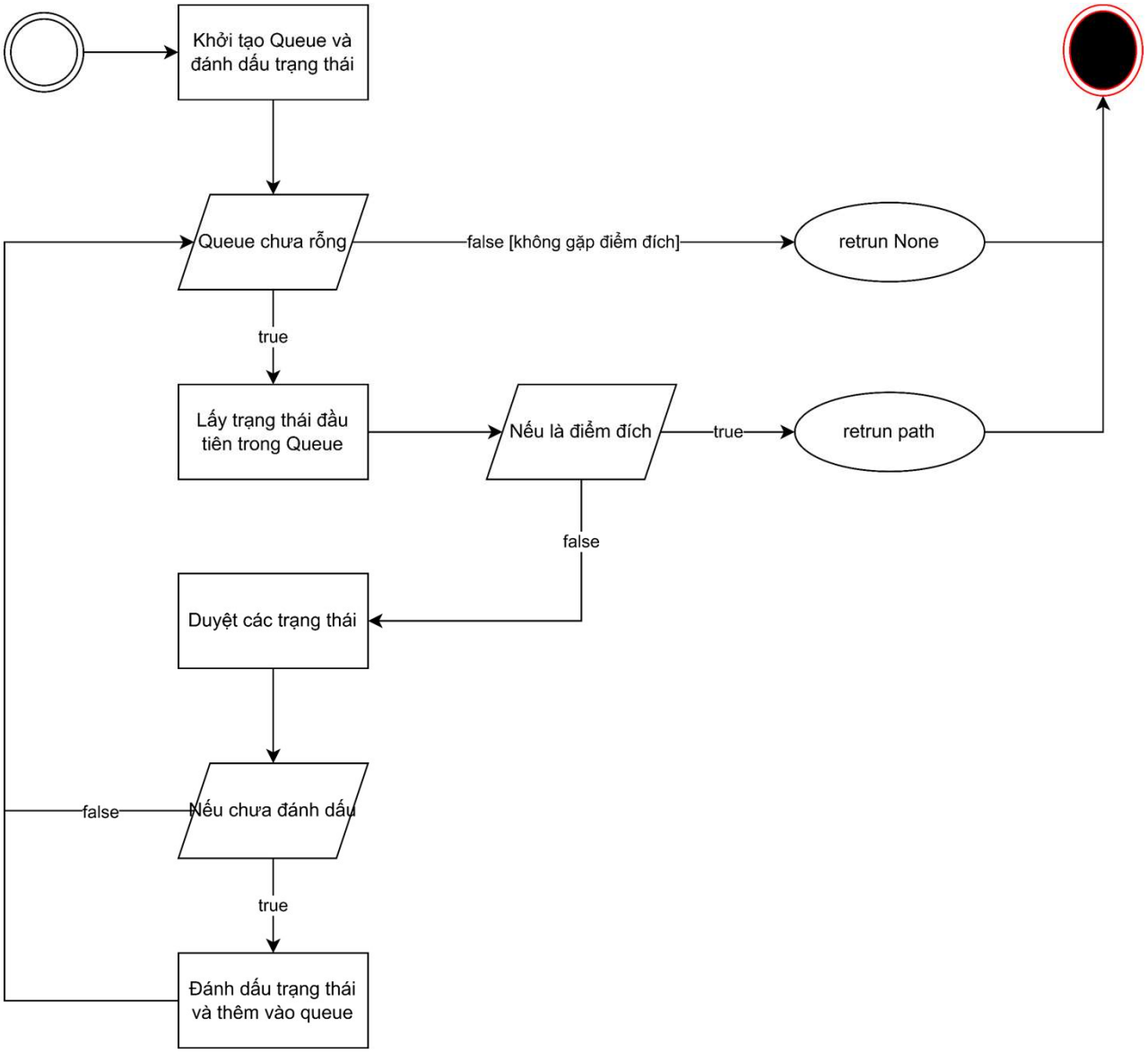

```python
You, 3 hours ago | 2 authors (You and others)
class PriorityQueue():
    def __init__(self):
        self.items = []

    def isEmpty(self):
        return len(self.items)==0

    def enqueue(self, this):
        self.items.append(this)
        self.items.sort()

    def dequeue(self):
        return self.items.pop(0)

    def contain(self, this):
        return this in [it[1] for it in self.items]

    def getPriority(self, key):
        for w, v in self.items:
            if v == key:
                return w
        raise 'KeyError'

    def updatePriority(self, v, w):
        p_w = self.getPriority(v)
        self.items.remove((p_w, v))
        self.items.append((w, v))
        self.items.sort()
```

PrioriryQueue workflow and implements

UNINFORMED SEARCH

Yêu cầu 1-3

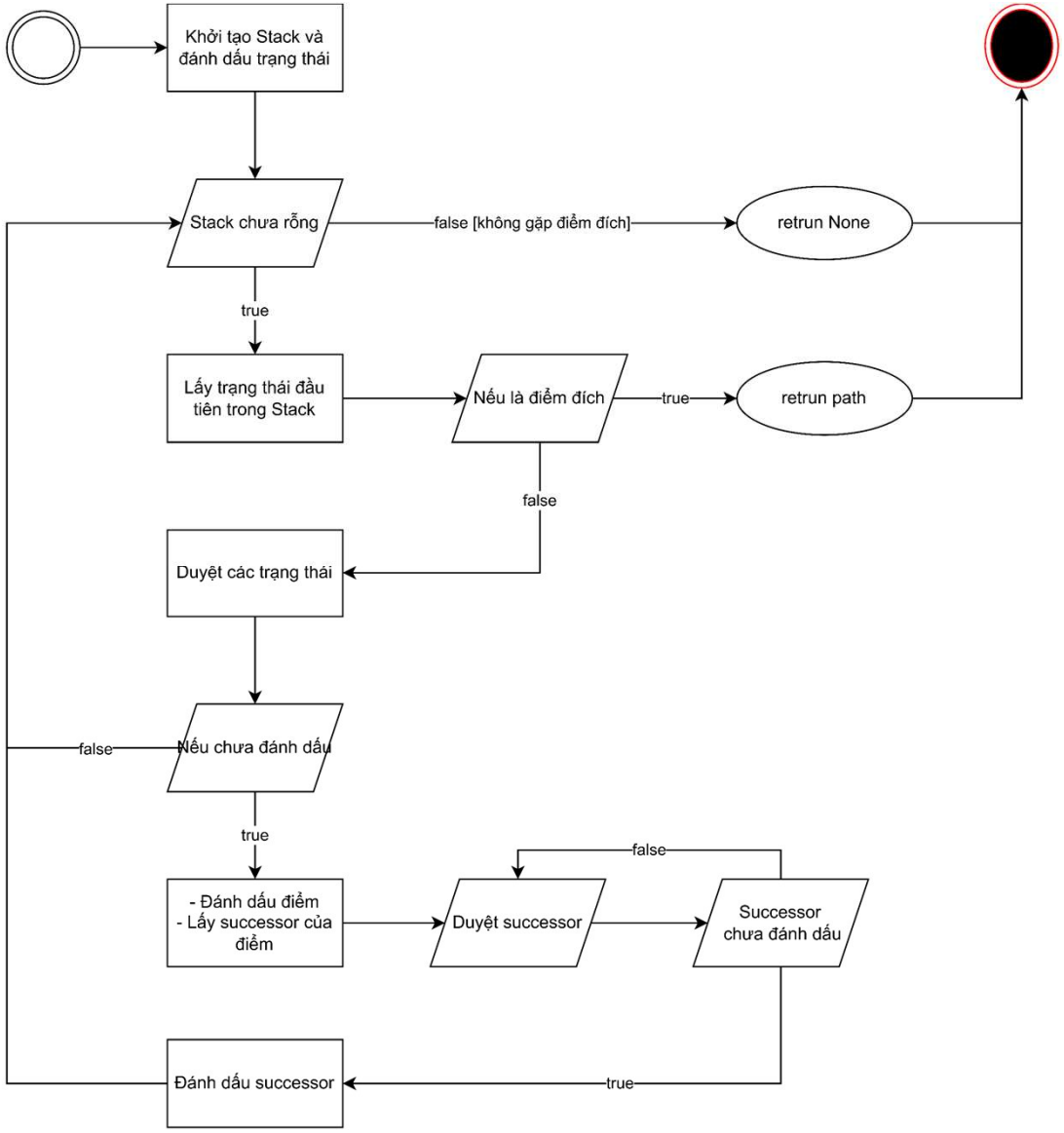
BFS



UNINFORMED SEARCH

Yêu cầu 1-3

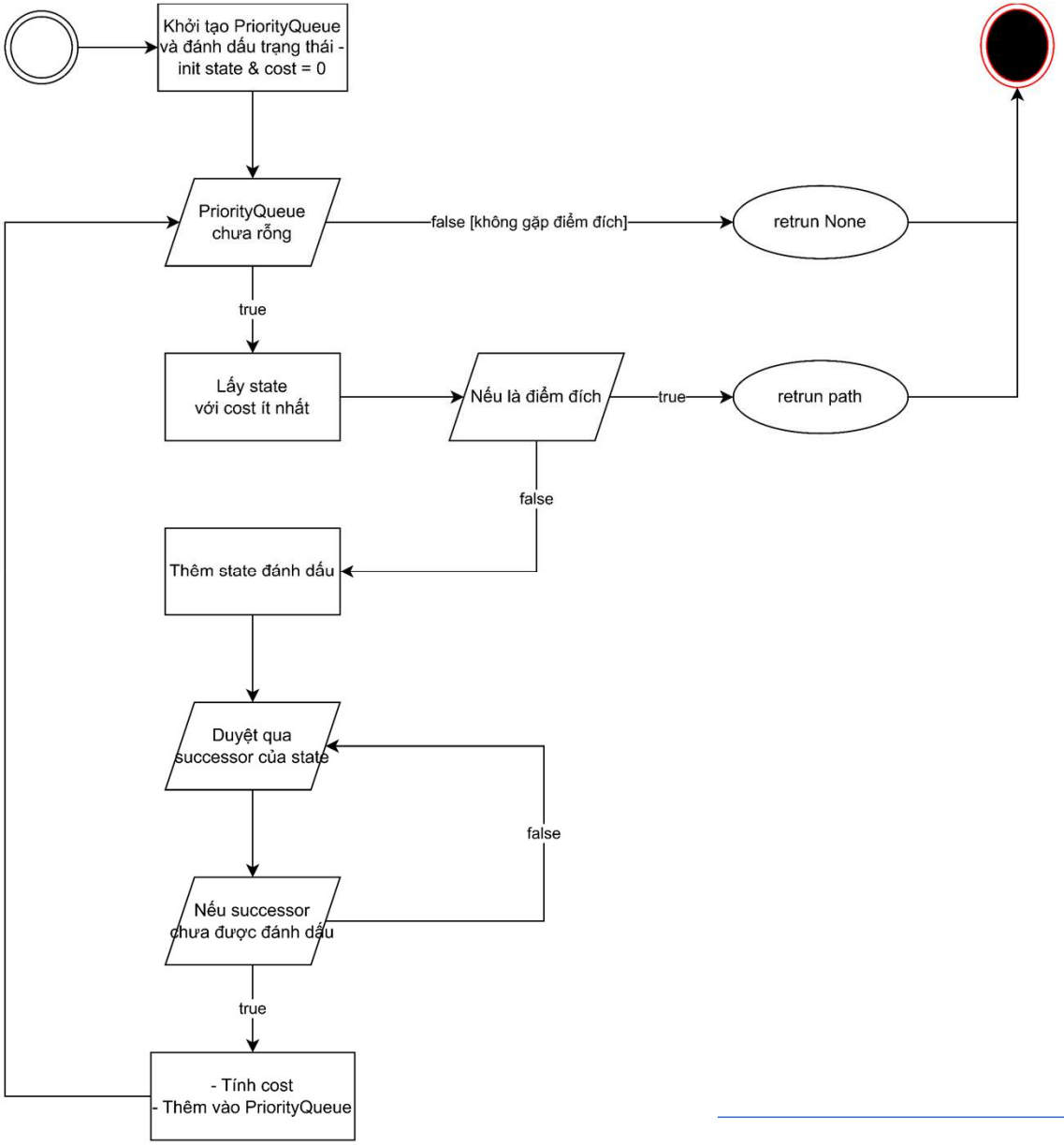
DFS



UNINFORMED SEARCH

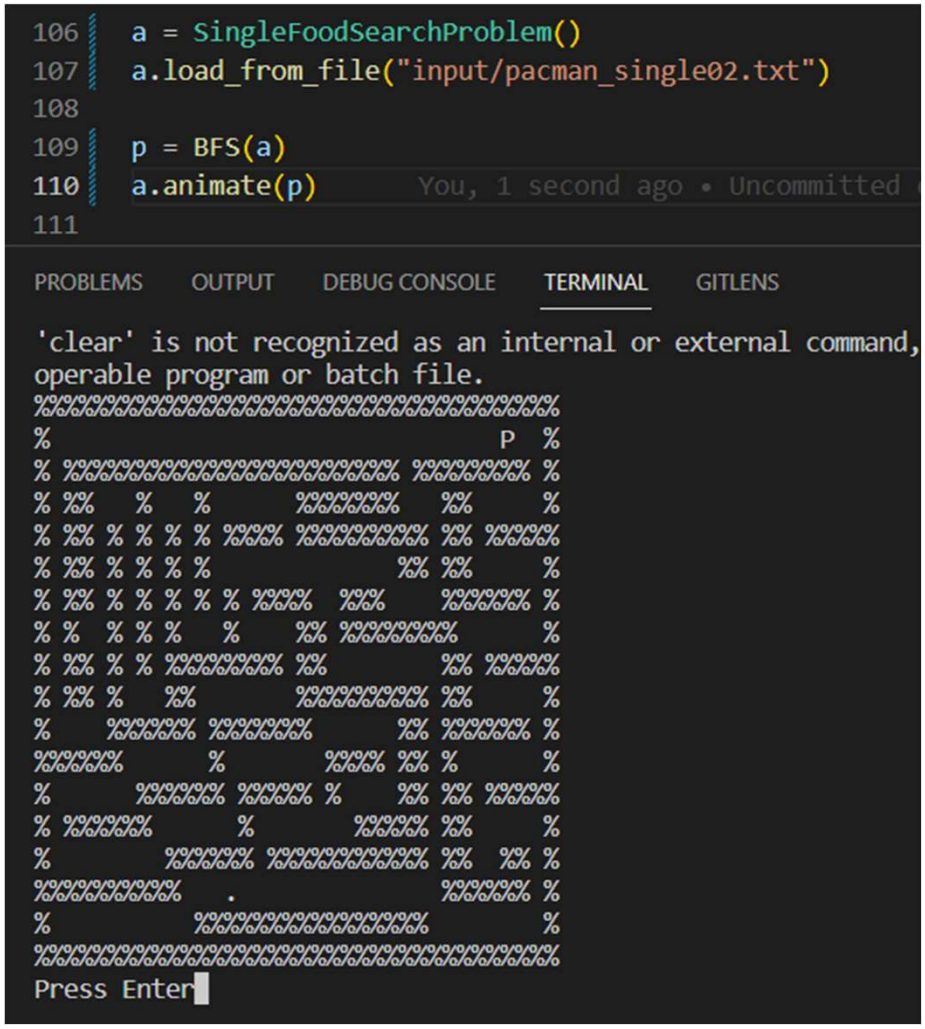
Yêu cầu 1-3

UCS



Yêu cầu 1-4

```python
def animate(self, actions) -> None:
    cur = self.P
    for i in actions:
        os.system("cls")
        os.system("clear")
        self.__str__()
        self.map[cur[0]][cur[1]] = " "
        if i == "Stop":
            break
        if i == "N":
            cur[0] -= 1
            self.map[cur[0]][cur[1]] = "P"
        if i == "S":
            cur[0] += 1
            self.map[cur[0]][cur[1]] = "P"
        if i == "W":
            cur[1] -= 1
            self.map[cur[0]][cur[1]] = "P"
        if i == "E":
            cur[1] += 1
            self.map[cur[0]][cur[1]] = "P"
        enter = input("Press Enter")
```



Cài đặt animate() theo yêu cầu

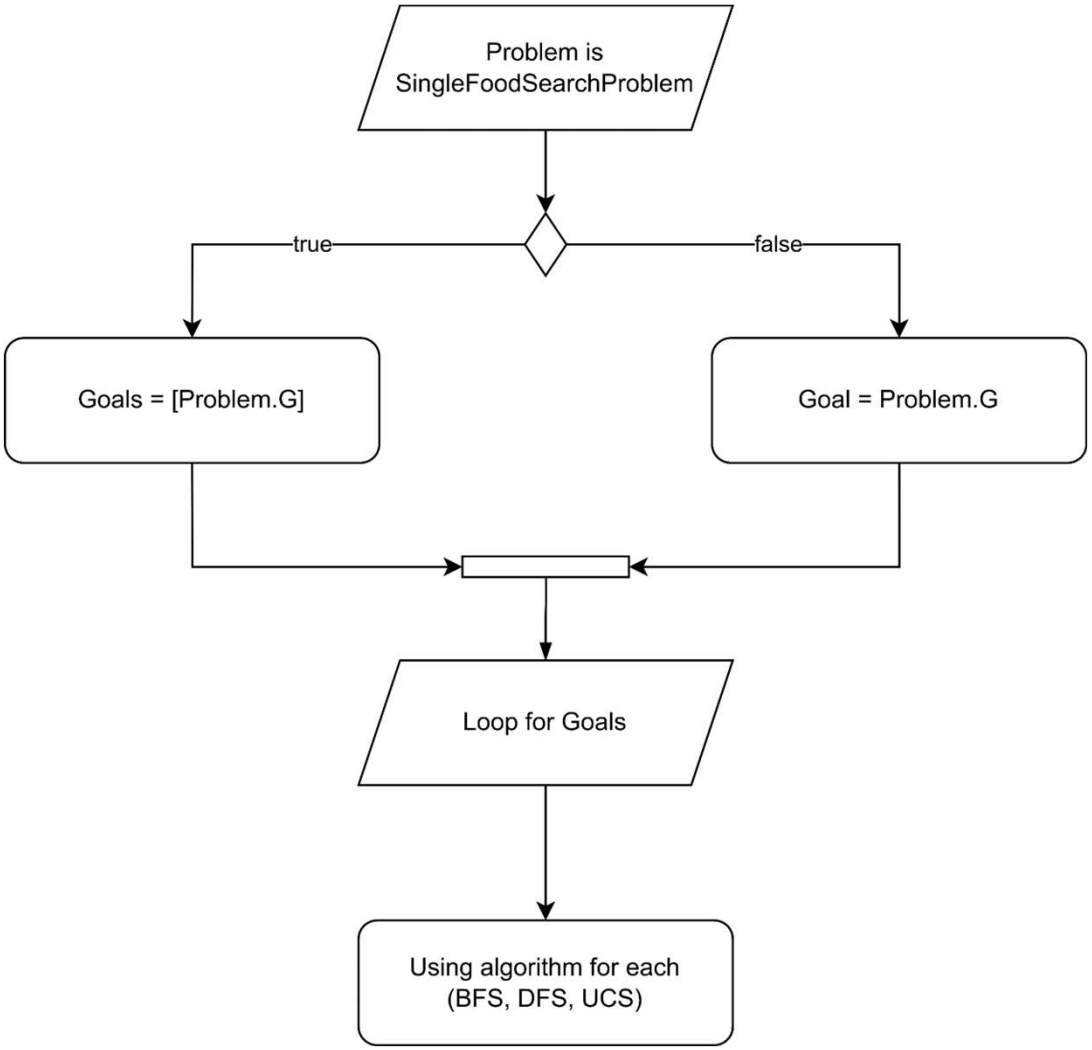
Yêu cầu 1-5

```
1
2  class MultiFoodSearchProblem:
3      def __init__(self):
4          self.map = []    #map
5          self.P = []      #init state
6          self.G = []      #list food
7          self.state = []
8
9      def successor(self,state):
10        return child
11
12     def isGoal(self,state):
13         return boolean
14
15     def load_from_file(self, filename):
16         pass
17
18     def __str__(self) -> str:
19         print(something)
20
21     def animate(self, actions) -> None:
22         pass
23
```

Cài đặt lớp MultiFoodSearchProblem

UNINFORMED SEARCH

Yêu cầu 1-6



Cải tiến YC1-3 để dung được cả cho SingleFoodSearchProblem và MultiFoodSearchProblem

Đánh giá hoàn thành

Tiêu chí	Điểm	Đánh giá
YC1-1	0.5 điểm	Hoàn thành
YC1-2	1.5 điểm	Hoàn thành
YC1-3	1.5 điểm	Hoàn thành
YC1-4	0.5 điểm	Hoàn thành
YC1-5	0.5 điểm	Hoàn thành
YC1-6	0.5 điểm	Hoàn thành

BEST FIRST SEARCH

Yêu cầu 2-1

Khoảng cách Euclidean $$d(p,q) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2}.$$

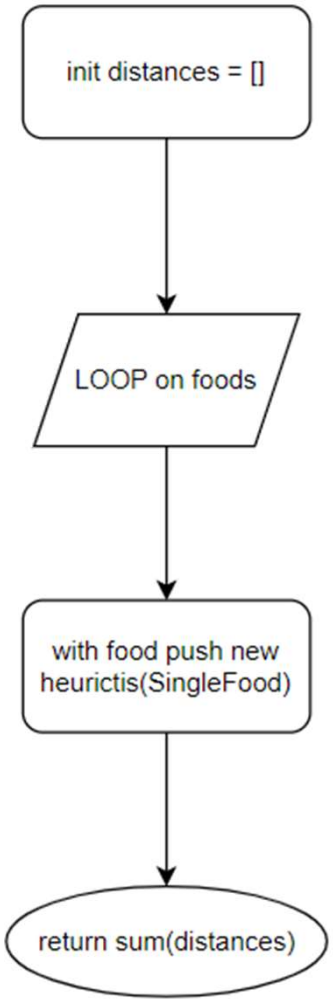
Khoảng cách Manhattan $$|x1 - x2| + |y1 - y2|$$

Khoảng cách Chebyshev $$D_{\text{Chebyshev}}(p,q) := \max_i(|p_i - q_i|).$$

Các hàm heuristic có thể dung cho SingleFoodSearchProblem

BEST FIRST SEARCH

Yêu cầu 2-2



Heuristic dung cho MultiFoodSearchProblem

BEST FIRST SEARCH

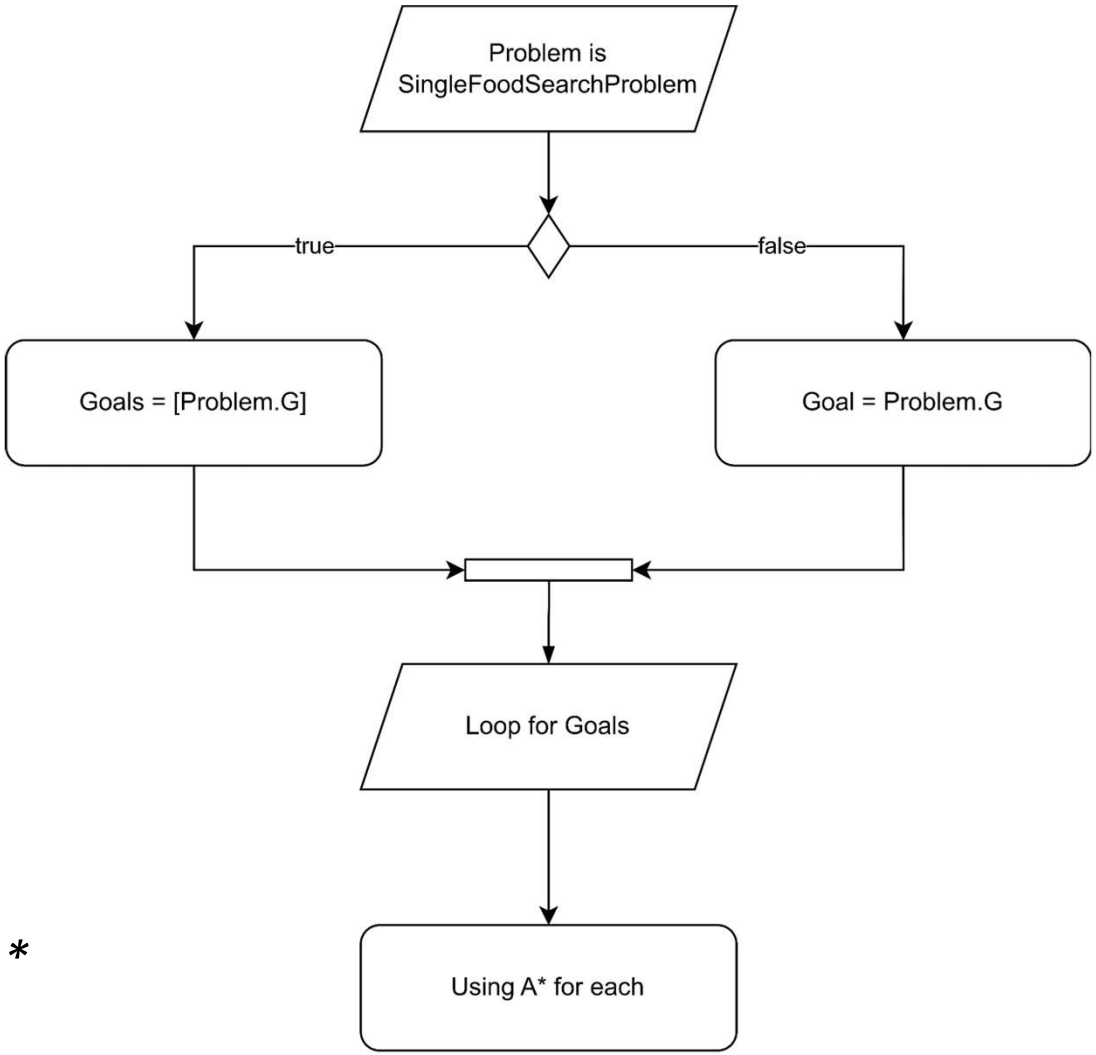
Yêu cầu 2-3

```
Function A*(problem, fn_heuristic)
    expanded // lưu các node đã mở rộng
    parents // lưu node cha của các node. (node, node's parent)
    pq // PriorityQueue
    pq.enqueue((fn_heuristic(nodeStart), cost, nodeStart))
    While pq is not empty
        h, c, cur = pq.dequece
        expanded.add(cur)
        if node.state is goal
            return getPath(goal, parents)
        for s in cur.successors
            if s is not in expanded
                s_cost = c + 1
                pq.enqueue((fn_heuristic(nodeStart) + s_cost, s_cost, cur))
                parents.add((s,cur))
```

*Cài đặt giải thuật A**

BEST FIRST SEARCH

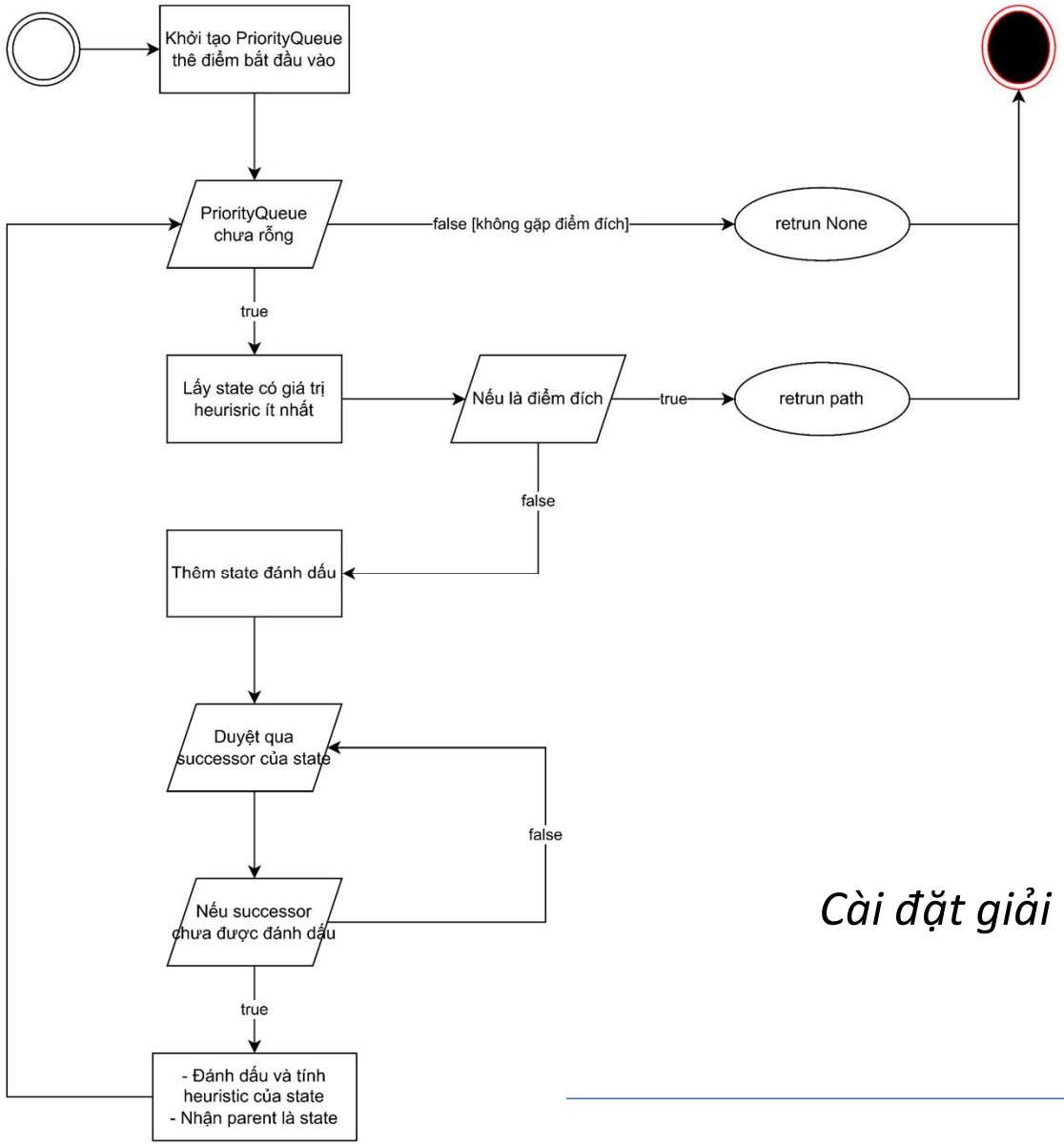
Yêu cầu 2-4



*Quy trình cải tiến A**

BEST FIRST SEARCH

Yêu cầu 2-5

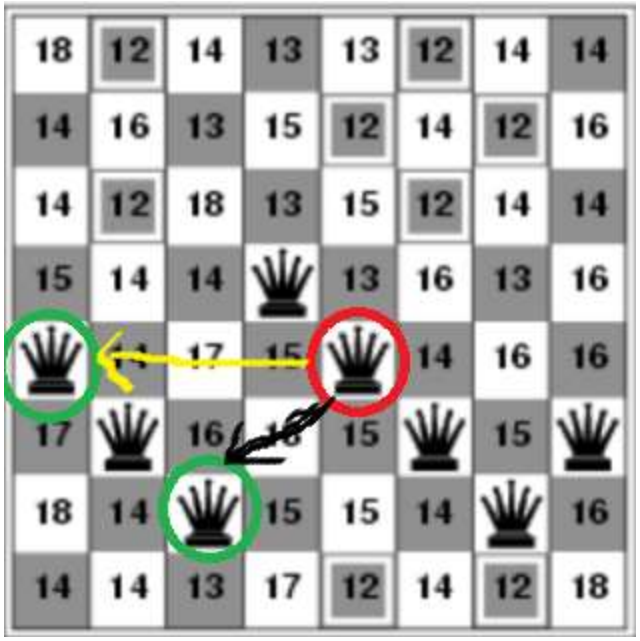


Cài đặt giải thuật GBFS

Đánh giá hoàn thành

Tiêu chí	Điểm	Đánh giá
YC2-1	1.0 điểm	Hoàn thành
YC2-2	0.5 điểm	Hoàn thành
YC2-3	0.5 điểm	Hoàn thành
YC2-4	0.5 điểm	Hoàn thành
YC2-5	0.5 điểm	Hoàn thành

Yêu cầu 3-1




```python
def h(self, state):
    n = len(state)
    count = 0
    for i in range(n):
        for j in range(i + 1, n):
            if state[i] == state[j] or abs(state[i] - state[j]) == j - i:
                count += 1
    return count
```

Hàm heuristic của bài toán 8 quân hậu

Yêu cầu 3-2

```
Hill_Climbing_Search():
    create state if not exits
    while not goal_state(board):
        successor_states = generate_successors(board)
        best_state = board
        for state in successor_states:
            if heuristic(state) < heuristic(best_state):
                best_state = state
        if best_state == board:
            return board
        board = best_state
    return board
```

Giải thuật leo đồi

Đánh giá hoàn thành

Tiêu chí	Điểm	Đánh giá
YC3-1	0.5 điểm	Hoàn thành
YC3-2	0.5 điểm	Hoàn thành

TỔNG KẾT

Đánh giá tổng quát

Tiêu chí	Điểm	Đạt được
Câu 1	5 điểm	5 điểm
Câu 2	3 điểm	3 điểm
Câu 3	1 điểm	1 điểm
Câu 4	1 điểm	…
Tổng		…

TỔNG KẾT

Thuận lợi

- Đa số các thành viên làm việc ăn ý
- Đa số các thành viên có kiến thức về Git
- Đa số thành viên nhiệt tình tham gia tìm hiểu đề tài
- Vấn đề nghiên cứu được phân chia bài bản
- Các thành viên có nền tảng lập trình tốt

TỔNG KẾT

Khó khăn

- Số lượng thành viên nghiên cứu thực tế tương đối ít
- Mất nhiều thời gian để ôn tập lại python do các thành viên không xuất phát điểm từ các ngành khoa học máy tính
- Còn nhiều thiếu sót về phân tích đề
- Lượng công việc cho từng thành viên khá lớn
- Dù khó khăn nhưng nhóm vẫn hoàn thành toàn bộ bài tập

TRÍCH DẪN

References

Book

[1] *Stuart Russell, Peter Norvig. Artificial Intelligence : A Modern Approach, 2nd edition. New Jersey, McGraw-Hill, 2005*

[2] Tom Mitchell. Machine Learning. New York, McGraw-Hill, 1997.

[3] Manning and Schuetze, Foundations of Statistical Natural Language Processing, MIT Press. Cambridge, MA, 1999.

Website

[1] sakai.it.tdt.edu.vn