**LAB 1 – SEARCH**

MSSV: 3122410266

Họ và tên: Đỗ Khôi Nguyên

Bài 1.

1.

- Tập trạng thái: A,B,C,D,E,F

- Tập các cung: Được xác định bởi danh sách kề A → {B,C,F}, B → {A, C, E, F},

C →{A,B,D, F}, D → {C, F}, E → {B,D,F}, F → {A,C,E}.

- Trạng thái bắt đầu: A

- Trạng thái kết thúc: D

2.

- Tập trạng thái: các ô trong mê cung (i,j) với i là chỉ số dòng và j là chỉ số cột.

- Tập các cung: một trạng thái kề với 4 ô xung quanh với thứ tự di chuyển là lên, xuống, trái, phải nhưng phải thỏa điều kiện không được đi ra ngoài mê cung cung như đụng vào tường.

- Trạng thái bắt đầu: ô (1,1) nơi chứa con kiến.

- Trạng thái kết thúc: ô (6,6) nơi chứa chữ A.

3.

- Tập trạng thái: [x1, x2, x3, x4, x5, x6, x7, x8, x9] với xi = {0,1,2,3,4,5,6,7,8}. Trong đó số 0 biểu diễn cho ô rỗng và các vị trí số còn lại biểu diễn cho một trạng thái trên bàn cờ.

- Tập các cung: luật di chuyển là đưa ô trống theo 4 hướng lên, xuống, trái, phải sao cho hợp lệ.

- Trạng thái bắt đầu: [4,8,1,6,3,0,2,7,5]

- Trạng thái kết thúc: [1, 2, 3, 4, 5, 6, 7, 8, 0]

Bài 2.

1.

Ma trận kề:

import numpy as np

import pprint

i = {

    'A': ['C', 'E', 'D', 'F'],

    'B': [],

    'C': ['H'],

    'D': ['H', 'E'],

    'E': ['K', 'I'],

    'F': ['I', 'G'],

    'G': [],

    'H': ['K'],

    'I': ['B', 'K'],

    'K': ['B']

}

def MaTranKe(i):

    v = ["A", "B", "C", "D", "E", "F", "G", "H", "I", "K"]

    n = len(v)

    edges = {v:i for i, v in enumerate(v)}

    matrix = np.zeros((n,n), int)

    for key in i:

        for value in i[key]:

            matrix[edges[key]][edges[value]] = 1

    pprint.pprint(matrix)

MaTranKe(i)

A black rectangle with white text

AI-generated content may be incorrect.

Danh sách kề:

import numpy as np

import pprint

i = {

    'A': ['C', 'E', 'D', 'F'],

    'B': [],

    'C': ['H'],

    'D': ['H', 'E'],

    'E': ['K', 'I'],

    'F': ['I', 'G'],

    'G': [],

    'H': ['K'],

    'I': ['B', 'K'],

    'K': ['B']

}

def DanhSachKe(i):

    graph = {}

    for key, values in i.items():

        print(key, end=": ")

        print(\*values, sep=", ")

DanhSachKe(i)

A black rectangle with white text

AI-generated content may be incorrect.

2.

- Tập các trạng thái: các đỉnh của đồ thị gồm {A,B,C,D,E,F,G,H,I,K}.

- Tập các cung: Danh sách kề, ma trận kề hay danh sách cạnh.

- Trạng thái ban đầu: A

- Trạng thái kết thúc: B

3.

Các bước chính của thuật toán DFS và BFS:

+ B1: Khởi tạo Open, Closed rỗng.

+ B2: Đưa trạng thái ban đầu vào Open.

+ B3: Trong khi tập Open còn khác rỗng thì làm

* Lấy 1 trạng thái x từ tập Open.
* Xác định các trạng thái xi kề với trạng thái x mà xi không nằm trong tập Open cũng như Closed.
* Thêm trạng thái xi vào tập Open và bỏ x vào tập Closed.

Lưu ý: Thuật toán DFS lấy trạng thái x từ Open theo nguyên tắc LIFO (Last In First Out), BFS ưu tiên lấy x theo FIFO (First In First Out).

Bảng truy hồi của DFS:

|  |  |
| --- | --- |
| Đỉnh | Đỉnh trước đó |
| A | None |
| C | A |
| H | C |
| K | H |
| B | K |

Bảng truy hồi của BFS

|  |  |
| --- | --- |
| Đỉnh | Đỉnh trước đó |
| A | None |
| C | A |
| D | A |
| E | A |
| F | A |
| H | C |
| K | E |
| I | E |
| G | F |
| B | K |

Bài 3.

1.

print("1. Lưu đồ thị xuống tập tin")

def LuuDoThi(G, file\_path, verbose = True):

    import os

    """ CÁC BẠN LÀM BÀI Ở ĐÂY """

    # Tạo thư mục chứa file\_path

    file\_dir = os.path.dirname(file\_path)

    if file\_dir != "" and os.path.exists(file\_path) == False:

        os.makedirs(file\_dir)

        if verbose: print(f"+ Tao thu muc: {file\_dir}")

    # Lưu đồ thị

    with open(file\_path, "wt") as file:

        """ CÁC BẠN LÀM BÀI Ở ĐÂY """

        # ...

        file.write(str(len(G)) + "\n")

        for key, values in G.items():

            file.write(key + " " + " ".join(values) + "\n")

            ...

        if verbose: print(f"Luu do thi thanh cong xuong tap tin: {file\_path}")

    pass

ADJ = solve("dske.txt")

LuuDoThi(ADJ, "test.txt", verbose = True)

with open("test.txt", "rt") as file:

    lines = file.readlines()

    for line in lines: print(line, end="")

A black rectangle with white text

AI-generated content may be incorrect.

2.

print("2. Đọc đồ thị từ tập tin")

import pprint

def DocDoThi(file\_path, verbose = True):

    """

    return:

    + None: doc that bai

    + <>None: tra ve do thi

    """

    import os

    result = None

    if os.path.exists(file\_path) == False:

        result = None

    else:

        G = {}

        with open(file\_path, "rt") as file:

            n = int(file.readline())

            """ CÁC BẠN LÀM BÀI Ở ĐÂY """

            # ...

            for \_ in range(n):

                line = file.readline().strip()

                if line == 1:

                    G[line] = set()

                else:

                    line = line.split()

                    G[line[0]] = set(line[1:])

            pass

        result = G

    return result

G1 = DocDoThi("dske1.txt", verbose = True)

pprint.pprint(G1)

A black rectangle with white text

AI-generated content may be incorrect.

3.

DFS:

import pprint

def DFS(G, start, goal):

    """

    return:

    + mang chua duong di tu a --> b, neu rong la ko co duong di

    + None: dinh a, b khong hop le

    """

    result = None

    if G.get(start) is None or G.get(goal) is None:

        result = None

    else:

        path = {}     # path[a]=b nghia la muon di toi dinh b phai qua a

        s\_open   = []

        s\_closed = []

        # đưa start vào open

        s\_open.append(start)

        path[start] = None

        while len(s\_open)>0:

            """ CÁC BẠN LÀM BÀI Ở ĐÂY """

            # ...

            u = s\_open.pop()

            if u == goal:

                break

            for v in G[u]:

                if v not in path:

                    path[v] = u

                    s\_open.append(v)

            s\_closed.append(u)

            pass

    return path

pprint.pprint(DFS(solve("dske.txt"), "A", "B"))

A black rectangular object with a blue border

AI-generated content may be incorrect.

BFS:

import pprint

def BFS(G, start, goal):

    """

    return:

    + mang chua duong di tu a --> b, neu rong la ko co duong di

    + None: dinh a, b khong hop le

    """

    result = None

    if G.get(start) is None or G.get(goal) is None:

        result = None

    else:

        path = {}     # path[a]=b nghia la muon di toi dinh b phai qua a

        s\_open   = []

        s\_closed = []

        # đưa start vào open

        s\_open.append(start)

        path[start] = None

        while len(s\_open)>0:

            """ CÁC BẠN LÀM BÀI Ở ĐÂY """

            # ...

            u = s\_open.pop(0)

            if u == goal:

                break

            for v in G[u]:

                if v not in path:

                    path[v] = u

                    s\_open.append(v)

            s\_closed.append(u)

            pass

    return path

pprint.pprint(BFS(solve("dske.txt"), "A", "B"))

A black rectangle with white text

AI-generated content may be incorrect.