

Báo Cáo Thực Hành Tuần 3

Họ và tên: Bùi Thị Thanh Phương

MSSV:20110280

I. Nội dung thực hành

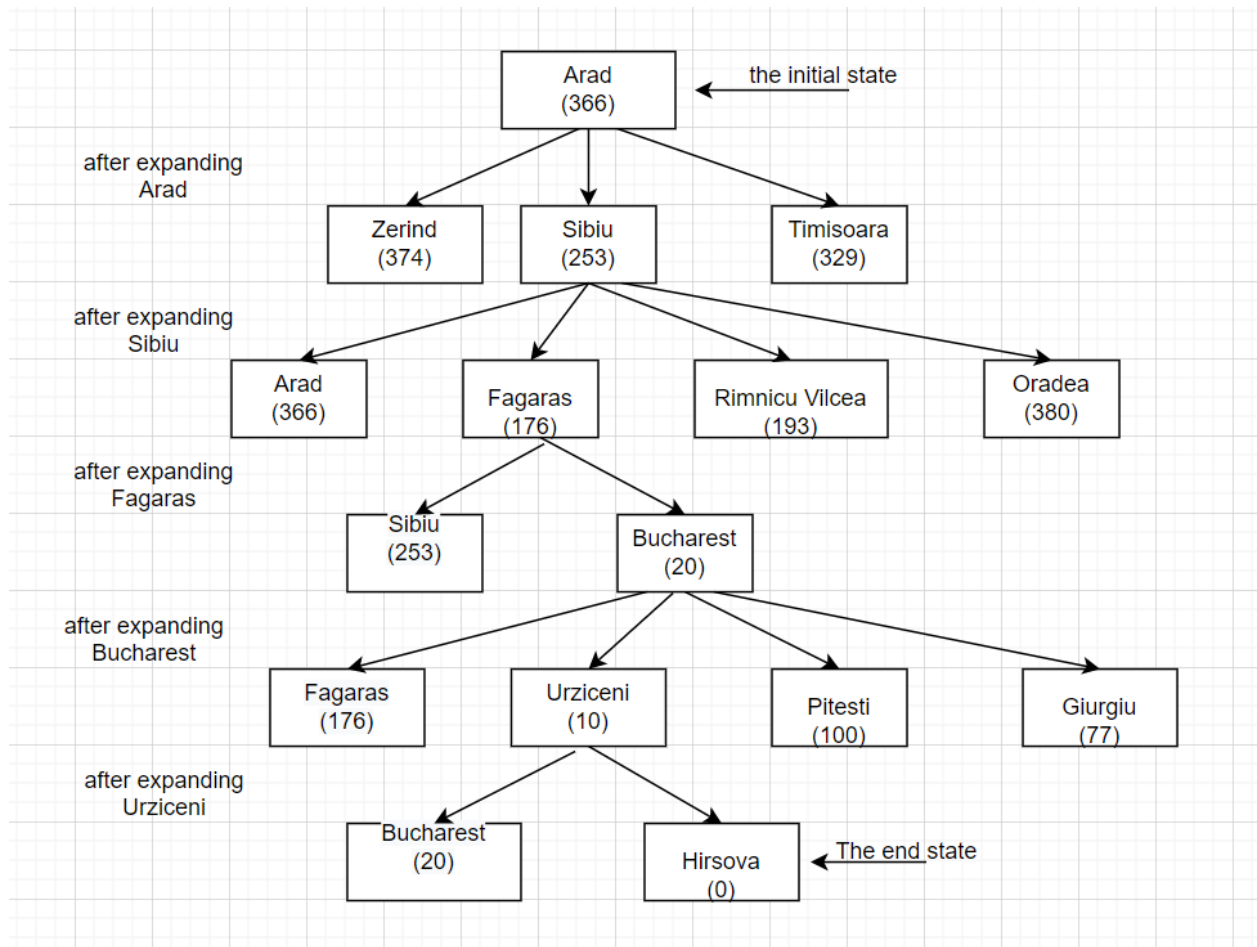
1. Cài đặt thuật toán Greedy – Best – First search để tìm đường đi từ Arad tới Hirsova như hình với $h(n)$ được xác định :

Thuật toán Greedy-Best-First Search: Thuật toán sẽ sử dụng 1 hàm đánh giá là hàm heuristic $h(n)$ (hàm heuristic $h(n)$ đánh giá chi phí để đi từ nút hiện tại n đến nút đích (mục tiêu)).

Với hàm heuristic là khoảng cách đường chim bay từ thành phố bất kì đến $h(x)=0$

Đánh giá thuật toán theo chiều ngang

- Xét bài toán đường đi từ Arad tới Hirsova như hình với $h(n)$ ta có được tính theo sơ đồ sau:



Từ sơ đồ trên ta có thể thấy đường đi từ Arad tới thành phố Hirsova :

$$\begin{aligned}
 & d(\text{Arad}) + d(\text{Sibiu}) + d(\text{Faragas}) + d(\text{Bucharest}) + d(\text{Uzicent}) + d(\text{Hirsova}) \\
 &= 366 + 253 + 176 + 20 + 10 + 0 \\
 &= 825
 \end{aligned}$$

2. Cài đặt thuật toán A* để tìm đường đi ngắn nhất từ Arad tới Hirsova như hình với hàm $h(n)$ được xác định như trong bài tập 1 và $g(n)$ là khoảng cách giữa 2 thành phố.

Nhận thấy anh chị trợ giảng đã tìm đường đi từ Arad đến Bucharest nên em sẽ làm phần còn lại như sau:

Trong tập OPEN, Zerind có giá trị f nhỏ nhất nên $T_{\max} = \text{Zerind}$. Từ zerind ta có thể được tới 2 thành phố Arad và oradea. Lấy zerind ra khỏi tập OPEN và đưa vào tập CLOSE. Tương tự, ta cũng tính giá trị h , g và f của các thành phố này.

$$H(\text{Arad})=366$$

$$G(\text{Arad}) = g(\text{Zerind}) + \text{cost}(\text{Arad}, \text{Zerind}) = 75 + 75 = 150$$

$$F(\text{Arad}) = g(\text{Arad}) + h(\text{Arad}) = 366 + 150 = 516$$

$$H(\text{Oradea}) = 380$$

$$G(\text{Oradea}) = g(\text{zerind}) + \text{cost}(\text{Oradea}, \text{Zerind}) = 75 + 71 = 146$$

$$F(\text{Oradea}) = g(\text{Oradea}) + h(\text{Oradea}) = 146 + 380 = 526$$

Nút Arad đã có trong CLOSE và $g(\text{Arad})$ mới được tạo ra có giá trị là 150 lớn hơn $g(\text{Arad})$ lưu trong CLOSE có giá trị là 0 nên ta sẽ không cập nhật giá trị g và f của Arad lưu trong CLOSE.

nút Oradea đã có trong OPEN và nút này có giá trị là 146 nhỏ hơn giá trị (Oradea) lưu trong Open nên ta sẽ thay nút (Oradea) 291 bằng (Oradea) và đặt nút cha là Zerind

Open

(Oradea, $g = 146$, $h = 380$, $f = 526$, Cha = Zerind),

(Craiova, $g = 366$, $h = 160$, $f = 526$, Cha = R. Vilcea),

(Giurgiu, $g = 508$, $h = 77$, $f = 585$, Cha = Bucharest),

(Urziceni, $g = 503$, $h = 10$, $f = 513$, Cha = Bucharest)}

CLOSE = {(Arad, $g = 0$, $h = 0$, $f = 0$),

(Sibiu, $g = 140$, $h = 253$, $f = 393$, Cha = Arad),

(R. Vilcea, $g = 220$, $h = 193$, $f = 413$, Cha = Sibiu),

(Fagaras, $g = 239$, $h = 176$, $f = 415$, Cha = Sibiu),

(Pitesti, $g = 317$, $h = 100$, $f = 417$, Cha = R. Vilcea)

(Bucharest, $g = 418$, $h = 20$, $f = 438$, Cha = Pitesti)

(Zerind, $g = 75$, $h = 374$, $f = 449$, Cha = Arad)}

Trong tập OPEN, Urziceni có giá trị f nhỏ nhất nên $T_{\max} = \text{Urziceni}$. Từ Urziceni ta có thể được tới 3 thành phố Bucharest, Hirsova và Vaslui. Lấy Urziceni ra khỏi tập OPEN và đưa vào tập CLOSE. Tương tự, ta cũng tính giá trị h , g và f của các thành phố này.

$$H(\text{Bucharest}) = 20$$

$$G(\text{Bucharest}) = g(\text{Urziceni}) + \text{cost}(\text{Bucharest}, \text{Urziceni}) = 503 + 85 = 588$$

$$F(\text{Bucharest}) = G(\text{Bucharest}) + h(\text{Bucharest}) = 588 + 20 = 608$$

$$H(\text{Hirsova}) = 0$$

$$G(\text{Hirsova}) = g(\text{Urziceni}) + \text{cost}(\text{Hirsova}, \text{Urziceni}) = 503 + 98 = 601$$

$$F(\text{Hirsova}) = g(\text{Hirsova}) + h(\text{Hirsova}) = 601 + 0 = 601$$

$$H(\text{Vaslui}) = 199$$

$$G(\text{Vaslui}) = g(\text{Urziceni}) + \text{cost}(\text{Vaslui}, \text{Urziceni}) = 503 + 142 = 645$$

$$F(\text{Vaslui}) = g(\text{Vaslui}) + h(\text{Vaslui}) = 645 + 199 = 844$$

Nút Bucharest đã có trong CLOSE và $g(\text{Bucharest})$ mới được tạo ra có giá trị là 588 lớn hơn $g(\text{Buchares})$ lưu trong CLOSE có giá trị là 418 nên ta sẽ không cập nhật giá trị g và f của Bucharest lưu trong CLOSE.

Hirsova và Vaslui không có trong tập OPEN lẫn CLOSE nên ta sẽ thêm 2 nút này vào tập OPEN.

Open

(Oradea, $g = 146$, $h = 380$, $f = 526$, Cha = Zerind),

(Craiova, $g = 366$, $h = 160$, $f = 526$, Cha = R. Vilcea),

(Giurgiu, $g = 508$, $h = 77$, $f = 585$, Cha = Bucharest),

(Hirsova, $g = 601$, $h = 0$, $f = 601$, cha = Urziceni)

(Vaslui, $g = 645$, $h = 199$, $f = 844$, cha = Urziceni) }

CLOSE = {(Arad, $g = 0$, $h = 0$, $f = 0$),

(Sibiu, $g = 140$, $h = 253$, $f = 393$, Cha = Arad),

(R. Vilcea, $g = 220$, $h = 193$, $f = 413$, Cha = Sibiu),

(Fagaras, $g = 239$, $h = 176$, $f = 415$, Cha = Sibiu),

(Pitesti, $g = 317$, $h = 100$, $f = 417$, Cha = R. Vilcea)

(Bucharest, $g = 418$, $h = 20$, $f = 438$, Cha = Pitesti)

(Zerind, $g = 75$, $h = 374$, $f = 449$, Cha = Arad)

(Urziceni, $g = 503$, $h = 10$, $f = 513$, Cha = Bucharest)}

Trong tập OPEN, Oradea có giá trị f nhỏ nhất nên $T_{\max} = \text{Oradea}$. Từ Oradea ta có thể được tới 2 thành phố Zerind, Sibiu. Lấy Oradea ra khỏi tập OPEN và đưa vào tập CLOSE. Tương tự, ta cũng tính giá trị h , g và f của các thành phố này.

$$H(\text{zerind}) = 374$$

$$G(\text{zerind}) = g(\text{Oradea}) + \text{cost}(\text{zerind}, \text{Oradea}) = 146 + 71 = 217$$

$$F(\text{zerind}) = g(\text{zerind}) + h(\text{zerind}) = 217 + 374 = 591$$

$$H(\text{Sibiu}) = 253$$

$$G(\text{Sibiu}) = g(\text{Oradea}) + \text{cost}(\text{Sibiu}, \text{Oradea}) = 146 + 151 = 297$$

$$F(\text{Sibiu}) = g(\text{Sibiu}) + h(\text{Sibiu}) = 253 + 297 = 550$$

Nút Zerind đã có trong CLOSE và $g(\text{zerind})$ mới được tạo ra có giá trị là 217 lớn hơn $g(\text{Zerind})$ lưu trong CLOSE có giá trị là 74 nên ta sẽ không cập nhật giá trị g và f của Zerind lưu trong CLOSE.

Nút Sibiu đã có trong CLOSE và $g(\text{Sibiu})$ mới được tạo ra có giá trị là 297 lớn hơn $g(\text{Sibiu})$ lưu trong CLOSE có giá trị là 140 nên ta sẽ không cập nhật giá trị g và f của Sibiu lưu trong CLOSE.

Open

(Craiova, $g = 366$, $h = 160$, $f = 526$, Cha = R. Vilcea),

(Giurgiu, $g = 508$, $h = 77$, $f = 585$, Cha = Bucharest),

(Hirsova, $g = 601$, $h = 0$, $f = 601$, cha = Urziceni)

(Vaslui, $g = 645$, $h = 199$, $f = 844$, cha = Urziceni) }

CLOSE = {(Arad, $g = 0$, $h = 0$, $f = 0$),

(Sibiu, $g = 140$, $h = 253$, $f = 393$, Cha = Arad),

(R. Vilcea, $g = 220$, $h = 193$, $f = 413$, Cha = Sibiu),

(Fagaras, $g = 239$, $h = 176$, $f = 415$, Cha = Sibiu),

(Pitesti, $g = 317$, $h = 100$, $f = 417$, Cha = R. Vilcea)

(Bucharest, $g = 418$, $h = 20$, $f = 438$, Cha = Pitesti)

(Zerind, $g = 75$, $h = 374$, $f = 449$, Cha = Arad)

(Urziceni, $g = 503$, $h = 10$, $f = 513$, Cha = Bucharest)

(Oradea, $g = 146$, $h = 380$, $f = 526$, Cha = Zerind)}

Trong tập OPEN, Craiova có giá trị f nhỏ nhất nên $T_{\max} = \text{Craiova}$. Từ Craiova, ta có thể được tới 3 thành phố Drobeta, Rimnicu Vilcea và Pitesti. Lấy Craiova, ra khỏi tập OPEN và đưa vào tập CLOSE. Tương tự, ta cũng tính giá trị h , g và f của các thành phố này.

$H(\text{Drobeta}) = 242$

$G(\text{Drobeta}) = g(\text{Craiva}) + \text{cost}(\text{Drobeta}, \text{craiva}) = 366 + 120 = 486$

$f(\text{Drobeta}) = g(\text{Drobeta}) + h(\text{Drobeta}) = 486 + 242 = 728$

$H(\text{Rimnicu Vilcea}) = 193$

$G(\text{Rimnicu Vilcea}) = g(\text{Craiva}) + \text{cost}(\text{Rimnicu Vilcea}) = 366 + 146 = 512$

$F(\text{Rimnicu Vilcea}) = g(\text{Rimnicu Vilcea}) + h(\text{Rimnicu Vilcea}) = 512 + 193 = 705$

$H(\text{pitesti}) = 100$

$G(\text{Pitesti}) = g(\text{Craiva}) + \text{cost}(\text{Pitesti}, \text{Craiva}) = 366 + 138 = 504$

$$F(\text{Pitesti}) = g(\text{Pitesti}) + h(\text{Pitesti}) = 504 + 100 = 604$$

Nút Rimnicu Vilcea đã có trong CLOSE và $g(\text{Rimnicu Vilcea})$ mới được tạo ra có giá trị là 512 lớn hơn $g(\text{Rimnicu Vilcea})$ lưu trong CLOSE có giá trị là 220 nên ta sẽ không cập nhật giá trị g và f của Sibiu lưu trong CLOSE.

Nút Pitesti đã có trong CLOSE và $g(\text{Pitesti})$ mới được tạo ra có giá trị là 504 lớn hơn $g(\text{Pitesti})$ lưu trong CLOSE có giá trị là 317 nên ta sẽ không cập nhật giá trị g và f của Sibiu lưu trong CLOSE.

Drobeta không có trong tập OPEN lẫn CLOSE nên ta sẽ thêm nút này vào tập OPEN.

Open

(Giurgiu, $g = 508$, $h = 77$, $f = 585$, Cha = Bucharest),

(Hirsova, $g = 601$, $h = 0$, $f = 601$, cha = Urziceni)

(Vaslui, $g = 645$, $h = 199$, $f = 844$, cha = Urziceni),

(Drobeta, $g = 486$, $h = 242$, $f = 728$, cha = Craiva)}

CLOSE = {(Arad, $g = 0$, $h = 0$, $f = 0$),

(Sibiu, $g = 140$, $h = 253$, $f = 393$, Cha = Arad),

(R. Vilcea, $g = 220$, $h = 193$, $f = 413$, Cha = Sibiu),

(Fagaras, $g = 239$, $h = 176$, $f = 415$, Cha = Sibiu),

(Pitesti, $g = 317$, $h = 100$, $f = 417$, Cha = R. Vilcea)

(Bucharest, $g = 418$, $h = 20$, $f = 438$, Cha = Pitesti)

(Zerind, $g = 75$, $h = 374$, $f = 449$, Cha = Arad)

(Urziceni, $g = 503$, $h = 10$, $f = 513$, Cha = Bucharest)

(Oradea, $g = 146$, $h = 380$, $f = 526$, Cha = Zerind),

(Craiova, $g = 366$, $h = 160$, $f = 526$, Cha = R. Vilcea)}

Trong tập OPEN, Giurgiu có giá trị f nhỏ nhất nên $T_{\max} = \text{Giurgiu}$. Từ Giurgiu, ta có thể được tới thành phố Bucharest. Lấy Giurgiu, ra khỏi tập OPEN và đưa vào tập CLOSE. Tương tự, ta cũng tính giá trị h , g và f của các thành phố này.

$$H(\text{Bucharest}) = 20$$

$$G(\text{Bucharest}) = g(\text{Giurgiu}) + \text{cost}(\text{Bucharest}, \text{Giurgiu}) = 508 + 90 = 598$$

$$F(\text{Bucharest}) = g(\text{Bucharest}) + h(\text{Bucharest}) = 598 + 20 = 618$$

Nút Bucharest đã có trong CLOSE và $g(\text{Bucharest})$ mới được tạo ra có giá trị là 598 lớn hơn $g(\text{Bucharest})$ lưu trong CLOSE có giá trị là 418 nên ta sẽ không cập nhật giá trị g và f của Bucharest lưu trong CLOSE.

Open

(Hirsova, $g = 601$, $h = 0$, $f = 601$, cha = Urziceni)

(Vaslui, $g = 645$, $h = 199$, $f = 844$, cha = Urziceni),

(Drobeta, $g = 486$, $h = 242$, $f = 728$, cha = Craiva)}

CLOSE = {(Arad, $g = 0$, $h = 0$, $f = 0$),

(Sibiu, $g = 140$, $h = 253$, $f = 393$, Cha = Arad),

(R. Vilcea, $g = 220$, $h = 193$, $f = 413$, Cha = Sibiu),

(Fagaras, $g = 239$, $h = 176$, $f = 415$, Cha = Sibiu),

(Pitesti, $g = 317$, $h = 100$, $f = 417$, Cha = R. Vilcea)

(Bucharest, $g = 418$, $h = 20$, $f = 438$, Cha = Pitesti)

(Zerind, $g = 75$, $h = 374$, $f = 449$, Cha = Arad)

(Urziceni, $g = 503$, $h = 10$, $f = 513$, Cha = Bucharest)

(Oradea, $g = 146$, $h = 380$, $f = 526$, Cha = Zerind),

(Craiova, $g = 366$, $h = 160$, $f = 526$, Cha = R. Vilcea),

(Giurgiu, $g = 508$, $h = 77$, $f = 585$, Cha = Bucharest)}

Trong tập OPEN, Hirsova có giá trị f nhỏ nhất nên $T_{max} = \text{Hirsova}$. Từ Hirsova ta có thể được tới 2 thành phố Urziceni và Eforie Lấy Hirsova, ra khỏi tập OPEN và đưa vào tập CLOSE.

Tương tự, ta cũng tính giá trị h , g và f của các thành phố này.

$H(\text{Urziceni}) = 10$

$G(\text{Urziceni}) = g(\text{Hirsova}) + \text{cost}(\text{Urziceni}, \text{Hirsova}) = 601 + 98 = 699$

$F(\text{Urziceni}) = g(\text{Urziceni}) + h(\text{Urziceni}) = 699 + 10 = 709$

$H(\text{Eforie}) = 161$

$G(\text{Eforie}) = g(\text{Urziceni}) + \text{cost}(\text{Urziceni}, \text{Eforie}) = 699 + 86 = 785$

$F(\text{Eforie}) = g(\text{Eforie}) + h(\text{Eforie}) = 785 + 161 = 946$

Nút Urziceni đã có trong CLOSE và $g(\text{Urziceni})$ mới được tạo ra có giá trị là 699 lớn hơn $g(\text{Urziceni})$ lưu trong CLOSE có giá trị là 503 nên ta sẽ không cập nhật giá trị g và f của Urziceni lưu trong CLOSE.

Eforie không có trong tập OPEN lẫn CLOSE nên ta sẽ thêm nút này vào tập OPEN.

Open

(Hirsova, $g = 601$, $h = 0$, $f = 601$, cha = Urziceni)

(Vaslui, g = 645, h = 199, f = 844, cha = Urziceni),
 (Drobeta, g = 486, h = 242, f = 728, cha = Craiva)
 (Eforie, g = 785, h = 161, f = 946)}
 CLOSE = {(Arad, g = 0, h = 0, f = 0),
 (Sibiu, g = 140, h = 253, f = 393, Cha = Arad),
 (R. Vilcea, g = 220, h = 193, f = 413, Cha = Sibiu),
 (Fagaras, g = 239, h = 176, f = 415, Cha = Sibiu),
 (Pitesti, g = 317, h = 100, f = 417, Cha = R. Vilcea)
 (Bucharest, g = 418, h = 20, f = 438, Cha = Pitesti)
 (Zerind, g = 75, h = 374, f = 449, Cha = Arad)
 (Urziceni, g = 503, h = 10, f = 513, Cha = Bucharest)
 (Oradea, g = 146, h = 380, f = 526, Cha = Zerind),
 (Craiova, g = 366, h = 160, f = 526, Cha = R. Vilcea),
 (Giurgiu, g = 508, h = 77, f = 585, Cha = Bucharest),
 (Hirsova, g = 601, h = 0, f = 601, cha = Urziceni)}

Ta có nút Hirsova là nút cần tìm

Vậy đường đi ngắn nhất từ Arad đến Hirsova là

$F(\text{Hirsova}) = g(\text{Hirsova}) + h(\text{Hirsova}) = 601$

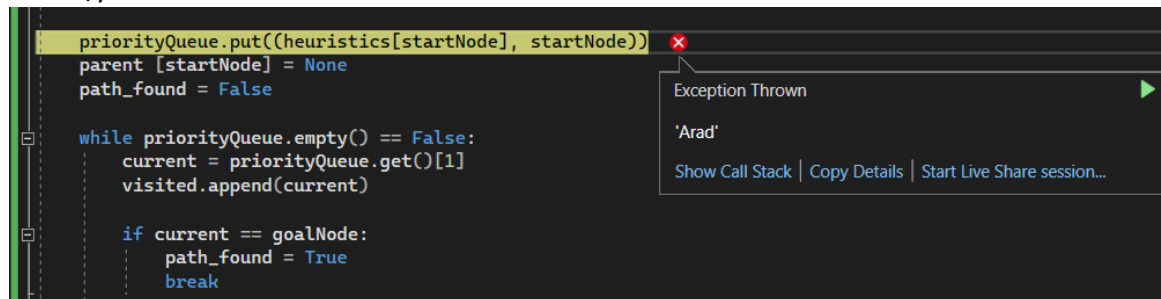
II. Cài đặt chương trình:

-Đã cài đặt chương trình nhận thấy dữ liệu đầu vào đã có một số lỗi sau:

- Lỗi cài đặt dữ liệu đầu vào:

Dữ liệu đầu vào tại file heuristics không có dữ liệu Arad

Khi chạy đã sinh ra lỗi:



```

priorityQueue.put((heuristics[startNode], startNode))
parent[startNode] = None
path_found = False

while priorityQueue.empty() == False:
    current = priorityQueue.get()[1]
    visited.append(current)

    if current == goalNode:
        path_found = True
        break
  
```

Exception Thrown
 'Arad'
 Show Call Stack | Copy Details | Start Live Share session...

Sửa lại: thêm dữ liệu: Arad 366 vào đầu file

- Dữ liệu đầu vào file citiesGraph bị thiếu

Sửa lại: thêm 3 dữ liệu sau vào cuối file:
Pitesti Rimnicu_Vilcea 97
Rimnicu_Vilcea Sibiu 80
Urziceni Vaslui 142

- Quá trình chạy code sau khi sửa file đúng tuy nhiên nhận thấy thuật toán bị sai :

Đầu tiên thuật toán GBFS, tuy kết quả cho ra đúng nhưng ở mỗi vòng lặp đều khởi tạo lại priority_queue, điều đó sẽ dẫn đến loại bỏ hết các đỉnh ở độ sâu trước đó và chỉ xét các nút ở cùng một độ sâu.

CODE: Sửa lại

```
def GBFS(startNode, heuristics, graph, goalNode):
    priorityQueue = queue.PriorityQueue()
    visited = []
    path = []
    parent = {}

    priorityQueue.put((heuristics[startNode], startNode))
    parent[startNode] = None
    path_found = False

    while priorityQueue.empty() == False:
        current = priorityQueue.get()[1]
        visited.append(current)

        if current == goalNode:
            path_found = True
            break

        for i in graph[current]:
            if i[0] not in visited:
                priorityQueue.put((heuristics[i[0]], i[0]))
                parent[i[0]] = current
                visited.append(i[0])

    if path_found:
        path.append(goalNode)
        while parent[goalNode] is not None:
            path.append(parent[goalNode])
            goalNode = parent[goalNode]
        path.reverse()

    return path
```

Thứ 2 thuật toán A* thuật toán khởi tạo lại priority_queue ở mỗi vòng lặp và hơn nữa không cập nhật giá trị đã duyệt nếu giá trị mới nhỏ hơn. Để thấy nhất trong thuật toán có hai tập open và close thì trong cài đặt không có hai tập đó.

Code sửa:

```

# Astar Algorithm
def Astar(start_node, heuristics, graph, goal_node):
    open = set([start_node])
    close = set([])

    parent = {}
    parent[start_node] = start_node

    g = {}
    g[start_node] = 0

    while len(open) > 0:
        n = None

        for v in open:
            if n == None or g[v] + heuristics[v] < g[n] + heuristics[n]:
                n = v

        if n == None:
            print('Path does not exist!')
            return None

        if n == goal_node:
            reconst_path = []

            while parent[n] != n:
                reconst_path.append(n)
                n = parent[n]

            reconst_path.append(start_node)
            reconst_path.reverse()
            return reconst_path

        # print(n)
        for (m, weight) in graph[n]:
            if m not in open and m not in close:
                open.add(m)
                parent[m] = n
                g[m] = g[n] + int(weight)

            else:
                if g[m] > g[n] + int(weight):
                    g[m] = g[n] + int(weight)
                    parent[m] = n

                if m in close:
                    close.remove(m)
                    open.add(m)

        open.remove(n)
        close.add(n)
    print('Path does not exist!')
    return None

```

- Chức năng các hàm sử dụng trong bài thực hành:

```
def getHeuristics():
```

Đây là hàm lấy khoảng cách đường đi đường chim bay của từ nút hiện tại đến nút đích ($h(\text{Hirsova}) = 0$)

Hàm:

```
def getCity():
```

Lấy vị trí của các thành phố từ file: `cities.txt`

Hàm

```
def createGraph():
```

Khởi tạo đồ thị các thành phố từ file: `citiesGraph.txt`

Hàm:

```
def GBFS(startNode, heuristics, graph, goalNode):
```

Tìm đường đi dựa vào thuật toán GBFS với

starNode: Nút bắt đầu

heuristics: Khoảng cách đường chim bay

Graph: Lưu đồ thị

goalNode: nút cuối

Hàm:

```
def Astar(start_node, heuristics, graph, goal_node):  
    open = set([start_node])  
    close = set([])
```

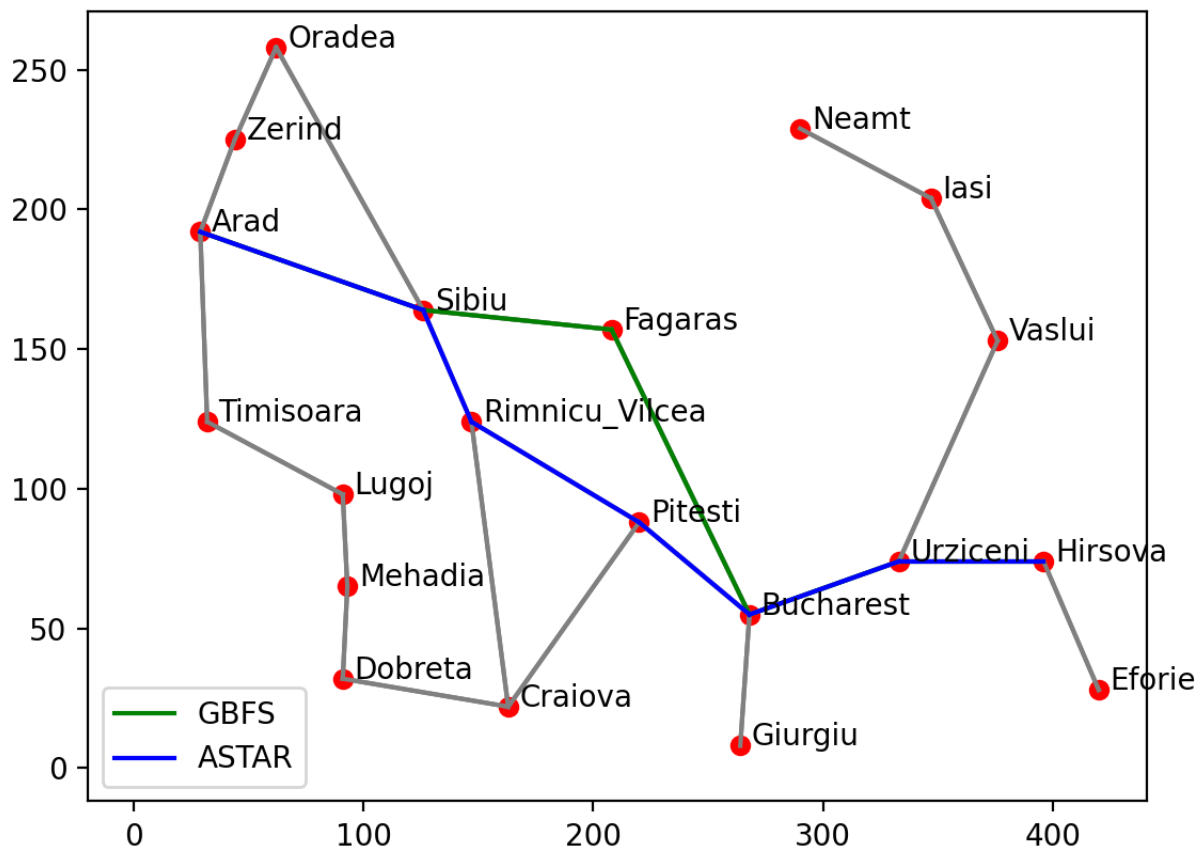
Để tìm đường đi dựa vào thuật toán Astar

- Kết quả chạy chương trình:

```

1 Arad
2 Bucharest
3 Craiova
4 Dobreta
5 Eforie
6 Fagaras
7 Giurgiu
8 Hirsova
9 Iasi
10 Lugoj
11 Mehadia
12 Neamt
13 Oradea
14 Pitesti
15 Rimnicu_Vilcea
16 Sibiu
17 Timisoara
18 Urziceni
19 Vaslui
20 Zerind
Nhap dinh bat dau: 1
Nhap dinh ket thuc: 8
GBFS => ['Arad', 'Sibiu', 'Fagaras', 'Bucharest', 'Urziceni', 'Hirsova']
ASTAR => ['Arad', 'Sibiu', 'Rimnicu_Vilcea', 'Pitesti', 'Bucharest', 'Urziceni', 'Hirsova']

```



Bài Báo cáo đến đây là hết.

Em xin cảm ơn thầy và các anh chị trợ giảng đã xem ạ.