Bài tập Thực hành 1 (Lab1)

Thông tin sinh viên

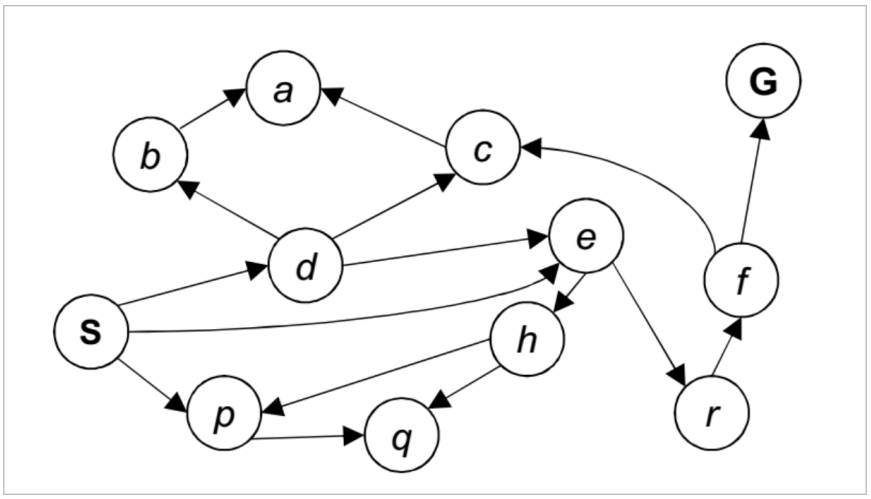
- MSSV: 22850034
- Họ và tên: Cao Hoài Việt
- Email: viet.ch2612@gmail.com

Mức độ hoàn thành công việc (100%)

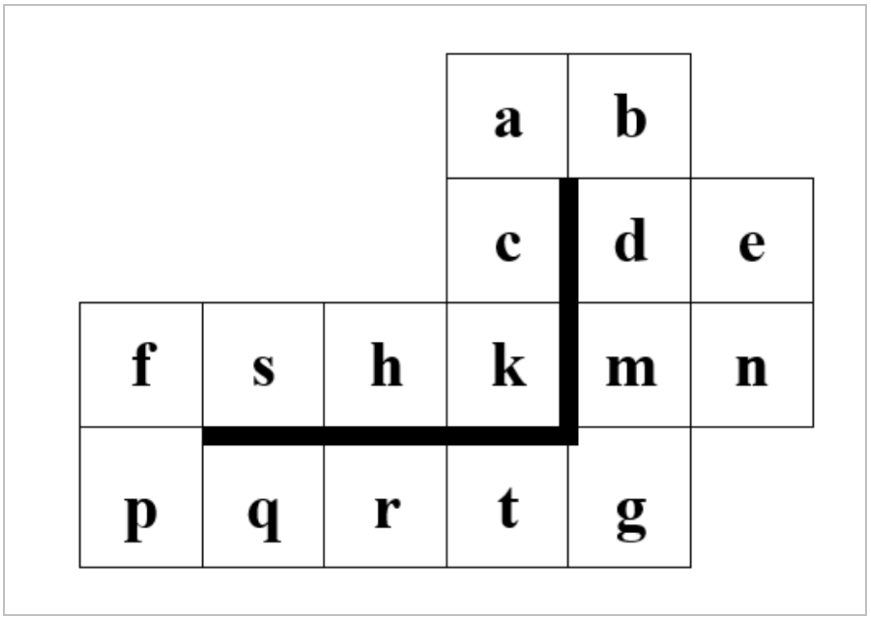
Yêu cầu	Đồ thị 1	Đồ thị 2	Đồ thị 3
2.1 BFS Phần hiển thị Kết quả	✅	✅	✅
2.2 DFS Phần Cài đặt	✅	✅	✅
2.2 DFS Phần Kết quả	✅	✅	✅

Bài làm

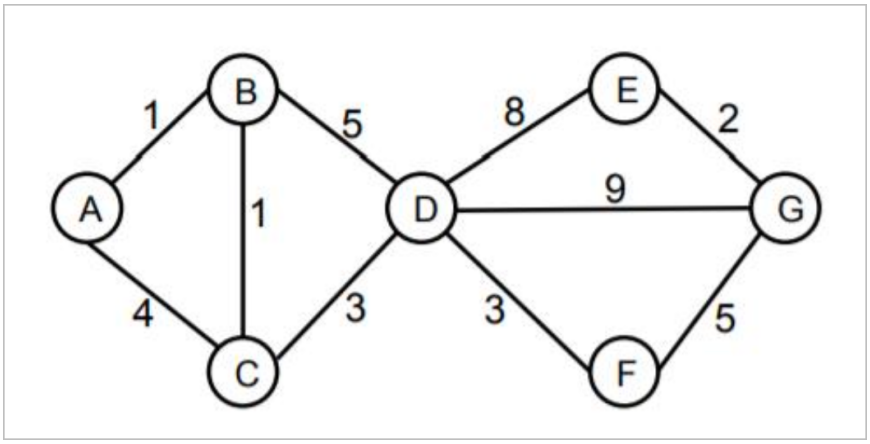
2.1. Cho biết kết quả thực thi với các đồ thị. (Theo BFS)




```
# Result
s d e p b c h r q a f g None   # Without path
['s', 'e', 'r', 'f', 'g']     # With path
```




```
# Result
s h f k p c q a r b t d g None       # Without path
['s', 'f', 'p', 'q', 'r', 't', 'g']  # With path
```




```
# Result
A B C D E G None       # Without path
['A', 'B', 'D', 'G']   # With path
```

2.2. Cài đặt thuật toán DFS và đường đi của các đồ thị

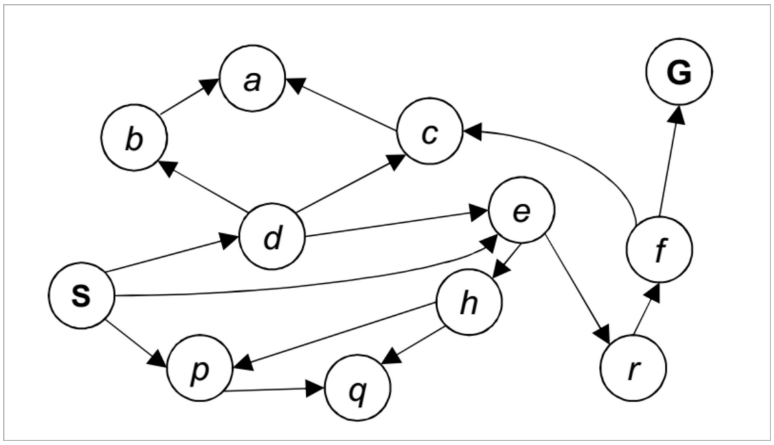
2.2.1 Thuật toán DFS không có lưu path

```
def dfs_without_path(graph, start, end):
    visited = []
    stack = []
    stack.append(start)

    while (stack):
        s = stack.pop()

        print(s, end=" ")
        if s == end:
            return

        if s not in visited:
            visited.append(s)

        for neighbor in graph[s]:
            if neighbor not in visited:
                stack.append(neighbor)
```

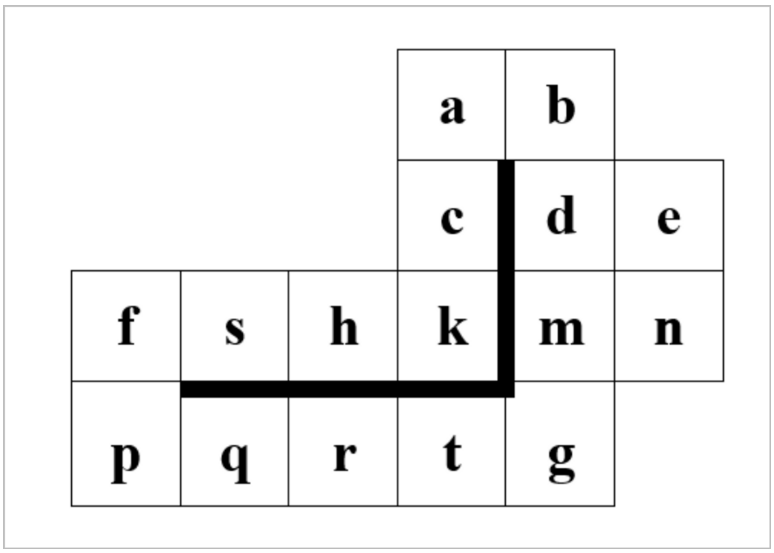
2.2.2 Thuật toán DFS có lưu path

```
def dfs(graph, start, end):
    # maintain a stack of paths
    visited = []
    stack = []
    # push the first path into the stack
    stack.append([start])
    while stack:
        # get the last path from the stack (Last in - First out)
        path = stack.pop()
        # get the last node from the path
        node = path[-1]
        # mark node as visited
        visited.append(node)
        # path found
        if node == end:
            return path
        # enumerate all adjacents, construct a new path and push it into the stack
        for neighbor in graph.get(node, []):
            if neighbor not in visited:
                new_path = list(path)
                new_path.append(neighbor)
                stack.append(new_path)
```

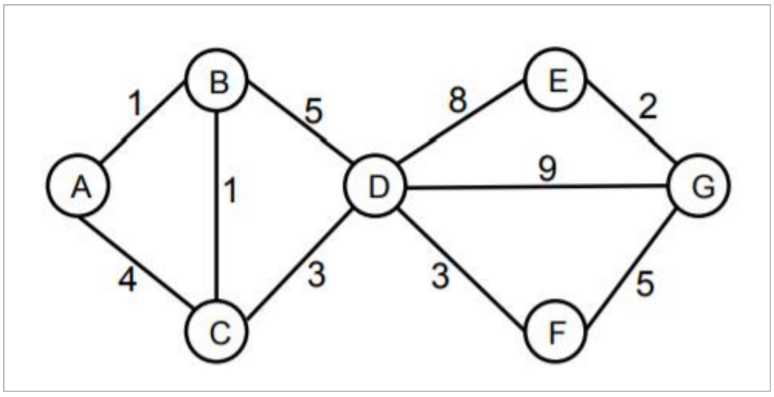
2.2.3 Kết quả khi chạy




```
s p q e r f g None          # Without path
['s', 'e', 'r', 'f', 'g']   # With path
```




```
s f p q r t g None                   # Without path
['s', 'f', 'p', 'q', 'r', 't', 'g']  # With path
```




```
A C D F G None               # Without path
['A', 'C', 'D', 'F', 'G'].   # With path
```