Assignment A1

```
from collections import defaultdict, deque

class Graph:
    directed = True

    def __init__(self):
        self.graph = defaultdict(list)

    def addEdge(self, u, v):
        self.graph[u].append(v)

        if not self.directed:
            self.graph[v].append(u)

    def DFS(self, v, d, visitSet = None) -> bool:
        visited = visitSet or set()
        visited.add(v)
        print(v,end=" ")

        if v == d:
            return True

        for neighbour in self.graph[v]:
            if neighbour not in visited:
                if self.DFS(neighbour, d, visited):
                    return True

        return False

    def BFS(self, s, d):
        visited = defaultdict(bool)
        queue = deque([s])
        visited[s] = True

        while queue:
            s = queue.popleft()
            print (s, end = " ")
            if s == d:
                return
            for i in self.graph[s]:
                if visited[i] == False:
                    queue.append(i)
                    visited[i] = True



if __name__ == '__main__':
    g = Graph()

    g.addEdge('H', 'A')
    g.addEdge('A', 'D')
    g.addEdge('A', 'B')
    g.addEdge('B', 'F')
    g.addEdge('B', 'C')
    g.addEdge('C', 'E')
    g.addEdge('C', 'G')
    g.addEdge('C', 'H')
    g.addEdge('G', 'H')
    g.addEdge('G', 'E')
    g.addEdge('E', 'F')
    g.addEdge('E', 'B')
    g.addEdge('F', 'A')
    g.addEdge('D', 'F')

    print("Following is Depth First Traversal H -> E:")
    g.DFS('H', 'E')

    print ("\n\nFollowing is Breadth First Traversal H -> E:")
    g.BFS('H', 'E')
```

Output :-

```
Following is Depth First Traversal H -> E:
H A D F B C E

Following is Breadth First Traversal H -> E:
H A D B F C E
```