Searching a graph in depth and breath (DFS and BFS)

- Depth first search:

```
def dfs(a, v):
    visited = []
    def traverse(a, v):
        visited.append(v)
        for j,k in enumerate(a[v]):
            if k == 0: continue
            if j not in visited: traverse(a, j)
    traverse(a, v)
    print(visited)
```

- Breadth first search

```
def bfs(a, v):
    visited = [v]
    temp = [j for j,k in enumerate(a[v]) if k == 1]
    while len(temp) > 0:
        visited.append(temp[0])
        for j,k in enumerate(a[temp[0]]):
            if k == 1 and j not in visited:
                temp.append(j)
        temp.pop(0)
    print(visited)
```

- The complexity of both DFS and BFS is $O(|V| + |E|)$