Depth First Search \_ adjacency list

* Time complexity:



DFS는 **DFS 하나당 각 정점에 연결되어 있는 간선의 개수만큼 탐색을 하게 되어 DFS 하나의 수행 시간이 일정하지 않다**.

전체 DFS가 다 끝난 이후를 생각해 보면, DFS가 다 끝났을 시점에는 모든 정점을 한번씩 방문하고, 모든 간선을 한번씩 검사하게 되므로 O(V+E)의 시간이 걸린다고 말할 수 있다.

따라서, 인접리스트로 구현할 경우 DFS의 시간 복잡도는 O(V+E)이다.

def dfs(v):

    check[v] = True

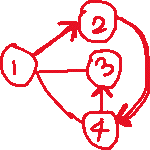
    print(v, end= ' ')

    for i in a[v]:

        if not check[i]:

            dfs(i)

EXAMPLE 1.



|  |  |
| --- | --- |
| 1 | [2,3,4] |
| 2 | [1,4] |
| 3 | [1,4] |
| 4 | [1,2,3] |

* dfs(1):

check[1] = True

print(1, end=’ ‘)

for j in [2,3,4]:

if not check[2]:

* dfs(2):

check[2] = True

print(2, end=’ ‘)

for j in [1,4]:

if not check[4]:

* dfs(4):

check[4] = True

print(4, end=’ ‘)

for j in [1,2,3]:

if not check[3]:

* dfs(3):

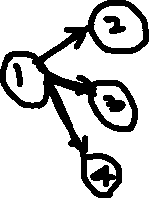
check[3] = True

print(3, end=’ ‘)

for j in [1,4]:

EXAMPLE 2.

|  |  |
| --- | --- |
| 1 | [2,3,4] |
| 2 | [1] |
| 3 | [1] |
| 4 | [1] |



* dfs(1):

check[1] = True

print(1, end=’ ‘)

for j in [2,3,4]:

if not check[2]:

* dfs(2):

check[2] = True

print(2, end=’ ‘)

for j in [1]:

if not check[3]:

* dfs(3):

check[3] = True

print(3, end=’ ‘)

for j in [1]:

if not check[4]:

* dfs(4):

check[4] = True

print(4, end=’ ‘)

for j in [1]: