DEPTH FIRST SEARCH (DFS)

Follow an unvisited path for as long as possible.

When you reach a vertex w/ only previously-visited neighbors, back up (from where you came from) & try again.
As with BFS, mark visited nodes.
As with BFS, mark visited nodes.
As with BFS, mark visited nodes.

$v_3$ has nowhere to go
As with BFS, mark visited nodes.

$V_3$ came from $\text{Adj}[v_2]$
As with BFS, mark visited nodes.

$V_2$ continues its search

...but $V_1$ has been visited

Adjacency list

```
S -> V_1 -> V_2 -> V_4 -> V_5
V_1 -> V_2 -> V_3 -> S
V_2 -> S -> V_3 -> V_1
V_3 -> V_2 -> V_1
V_4 -> S
V_5 -> S
```
As with BFS, mark visited nodes.

Now $v_2$ has nowhere to go. $v_2$ came from $\text{Adj}[v_1]$.
As with BFS, mark visited nodes.

$v_1$ doesn't know $v_3$ is marked
As with BFS, mark visited nodes.

$v_1$ discovers $v_3$ is marked
As with BFS, mark visited nodes.

$V_1$ discovers $S$ is marked and has nowhere else to go.

Adjacency list:

- $S ightarrow V_1 \rightarrow V_2 \rightarrow V_4 \rightarrow V_5$
- $V_1 \rightarrow V_2 \rightarrow V_3 \rightarrow S$
- $V_2 \rightarrow S \rightarrow V_3 \rightarrow V_1$
- $V_3 \rightarrow V_2 \rightarrow V_1$
- $V_4 \rightarrow S$
- $V_5 \rightarrow S$
As with BFS, mark visited nodes.

$v_1$ came from $\text{Adj}[S]$
As with BFS, mark visited nodes. s continues on Adj[s]...
...discovers v₂ is marked etc
Find $t$ from $s$:

Start with:
- mark $s$
- depth($s$) = 0
- DFS($s$)

**Time:** $O(|E|)$

$O(V+E)$ if not connected

DFS($u$) // DFS starting at $u$

for every neighbor $v_i$ of $u$ // i.e. scan Adj[$u$]

if $v_i = t$, DONE

if $v_i$ is unmarked

| mark $v_i$ |
| set parent($v_i$) → $u$ |
| set depth($v_i$) → $1 + \text{depth}(u)$ |

DFS($v_i$) // only if you want to keep the structure
DFS on a non-connected graph G

For every vertex $v_i$ in G
if $v_i$ is unmarked
DFS($v_i$)

It is also easy to keep a counter to keep track of the "time" at which each vertex is first encountered & fully processed.
DFS on a directed graph: similar to non-connected (process all vertices)

Iteratively, mark & explore if unmarked
DFS on a directed graph: similar to non-connected (process all vertices)
DFS on a directed graph: similar to non-connected (process all vertices)
DFS on a directed graph: similar to non-connected (process all vertices)
DFS on a directed graph: similar to non-connected (process all vertices)
DFS on a directed graph: similar to non-connected (process all vertices)
DFS on a directed graph: similar to non-connected (process all vertices)
DFS on a directed graph: similar to non-connected (process all vertices)
DFS on a directed graph : similar to non-connected (process all vertices)
DFS on a directed graph: similar to non-connected (process all vertices)

1, 2, 3, 4, 5

if \( v_3 \) came before \( v_2 \) in \( \text{Adj}[v_1] \)
DFS on a directed graph: similar to non-connected (process all vertices)

1, 2, 3, 4, 5

again
1, 2, 3, 4, 5

4 before 1, 5
... before 3?

3 before 1 in $\text{Adj}[4]$