Depth-First Search

DFS has a neat recursive implementation which eliminates the need to explicitly use a stack.

Discovery and final times are sometimes a convenience to maintain.

**DFS(G)**
for each vertex \( u \in V[G] \) do

\[
\begin{align*}
\text{color}[u] &= \text{white} \\
\text{parent}[u] &= \text{nil}
\end{align*}
\]

\( time = 0 \)
for each vertex \( u \in V[G] \) do

if \( \text{color}[u] = \text{white} \) then DFS-VISIT\([u]\)

Initialize each vertex in the main routine, then do a search from each connected component. BFS must also start from a vertex in each component to completely visit the graph.

**DFS-VISIT\([u]\)**

\[
\begin{align*}
\text{color}[u] &= \text{grey} \quad (*u \text{ had been white/undiscovered}*), \\
\text{discover}[u] &= \text{time} \\
\text{time} &= \text{time} + 1
\end{align*}
\]

for each \( v \in \text{Adj}[u] \) do

if \( \text{color}[v] = \text{white} \) then

\[
\begin{align*}
\text{parent}[v] &= u \\
\text{DFS-VISIT}(v)
\end{align*}
\]

\[
\begin{align*}
\text{color}[u] &= \text{black} \quad (*\text{now finished with } u*), \\
\text{finish}[u] &= \text{time} \\
\text{time} &= \text{time} + 1
\end{align*}
\]