CSI 503 – Data Structures and Algorithms
Breadth-First and Depth-First Search Methods

(a) Pseudocode for Breadth-First Search

BFS(G, s)  //  G is the graph and s is the source vertex.

1.  for each u \in V - \{s\}  do
    Color[u] = white;  \(d[u] = \infty\);  \(\pi[u] = \text{NULL}\).

2.  Color[s] = gray;  \(d[s] = 0\);  \(\pi[s] = \text{NULL}\).

3.  Initialize Q to contain only vertex s.

4.  while (Q is not empty)  do
    (a)  u = head[Q].
    (b)  for each v \in Adj[u]  do
        if (Color[v] = white)  then
            Color[v] = gray;  \(d[v] = d[u] + 1\);  \(\pi[v] = u\).
            Enqueue(Q, v).
        (c)  Dequeue(Q).  //  This removes u from Q.
        (d)  Color[u] = black.


PRINT-Path(g, s, v)  //  Prints the path in the BFS tree from s to v.

Idea:  Follow the parent pointers from v to s; recursion allows us to print the path in the right order.

1.  if (v = s)
    then  print s
    else
        if (\(\pi[v] = \text{NULL}\))
            then  Print “No path from s to v”.
            else
                PRINT-Path(G, s, \(\pi[v]\)).
                Print v.
(b) Pseudocode for Depth-First Search

DFS(G)  // Depth-first search of graph G(V, E).

Note: Procedure DFS uses procedure DFS-Visit described below.

1. for each vertex $u \in V$ do
   \[ \text{Color}[u] = \text{white}; \quad \pi[u] = \text{NULL}. \]
2. time = 0.  // Note: time is a global variable.
3. for each vertex $u \in V$ do
   \[ \text{if } (\text{Color}[u] = \text{white}) \text{ then} \]
   \[ \text{DFS-Visit}(u). \]

DFS-Visit($u$)

1. Color[$u$] = gray;  time = time+1;  $d[u]$ = time.  // $u$ has just been discovered.
2. for each vertex $v \in \text{Adj}[u]$ do  (Edge $\{u, v\}$ is explored.)
   \[ \text{if } (\text{Color}[v] = \text{white}) \text{ then} \]
   \[ \pi[v] = u; \quad \text{DFS-Visit}(v). \]
3. Color[$u$] = black.  // Vertex $u$ is finished.
4. time = time + 1;  $f[u]$ = time.