BFS and DFS reference – CLRS 4th edition algorithms

Input: Graph $G = (V, E)$, either directed or undirected, and ***source vertex*** $s \in V$.

Output:

- $v.d =$ distance (smallest # of edges) from s to v, for all $v \in V$.
- $v.\pi$ is v's ***predecessor*** on a shortest path (smallest # of edges) from s.
  (u, v) is last edge on shortest path $s \leadsto v$.
  Predecessor subgraph contains edges (u, v) such that $v.\pi = u$.
  The predecessor subgraph forms a tree, called the ***breadth-first tree***.

```
BFS(V, E, s)
  for each vertex u ∈ V − {s}
      u.d = ∞
      u.π = NIL
  s.d = 0
  Q = ∅
  ENQUEUE(Q, s)
  while Q ≠ ∅
      u = DEQUEUE(Q)
      for each vertex v in G.Adj[u]   // search the neighbors of u
          if v.d == ∞                 // is v being discovered now?
              v.d = u.d + 1
              v.π = u
              ENQUEUE(Q, v)           // v is now on the frontier
      // u is now behind the frontier.
```

Input: $G = (V, E)$, directed or undirected. No source vertex given.

Output:

- 2 ***timestamps*** on each vertex:
  - $v.d =$ ***discovery time***
  - $v.f =$ ***finish time***

  These will be useful for other algorithms later on.
- $v.\pi$ is v's predecessor in the ***depth-first forest*** of ≥ 1 ***depth-first trees***. If $u = v.\pi$, then (u, v) is a ***tree edge***.

```
DFS(G)
  for each vertex u ∈ G.V
      u.color = WHITE
      u.π = NIL
  time = 0
  for each vertex u ∈ G.V
      if u.color == WHITE
          DFS-VISIT(G, u)
```

```
DFS-VISIT(G, u)
  time = time + 1                  // white vertex u has just been discovered
  u.d = time
  u.color = GRAY
  for each vertex v in G.Adj[u]    // explore each edge (u, v)
      if v.color == WHITE
          v.π = u
          DFS-VISIT(G, v)
  time = time + 1
  u.f = time
  u.color = BLACK                  // blacken u; it is finished
```