作业反馈3-6

TC练习22.2-4 22.3-7.22.3-12.

22.2-4

What is the running time of BFS if we represent its input graph by an adjacency matrix and modify the algorithm to handle this form of input?

需要遍历整个矩阵 $O(|V|^2)$

22.3-7

Rewrite the procedure DFS, using a stack to eliminate recursion.

```
def DFS_STACK(G):
    for u in G.V:
        u.color = WHITE
        u.pi = NIL
    time = 0
    for u in G.V:
        if u.color is WHITE:
            time += 1
            u.d = time
            u.color = GRAY
            S = stack()
            S.push(u)
            while not S.empty():
```

```
                v = S.top()
                w = G.Adj[v].head()
                while w is not NIL and w.color is not WHITE:
                    w = G.Adj[v].next(w)
                if w is NIL:
                    time += 1
                    v.f = time
                    v.color = BLACK
                    S.pop()
                else:
                    time += 1
                    w.d = time
                    w.color = GRAY
                    w.pi = v
                    S.push(w)
```

22.3-12

Show that we can use a depth-first search of an undirected graph G to identify the connected components of G, and that the depth-first forest contains as many trees as G has connected components. More precisely, show how to modify depth-first search so that it assigns to each vertex v an integer label $v.cc$ between 1 and k, where k is the number of connected components of G, such that $u.cc = v.cc$ if and only if u and v are in the same connected component.

Add a counter cc and set it to 0. When the algorithm creates a new tree, increase cc by 1. Every time a new node u is visited, assign the current cc to $u.cc$.