DGA Ü6

1)

```
DFS-VISIT(G, u){
    t:=t+1; d(u):=t; c(u):=GRAU; K(u):=k;
    for v ∈ Adj[u] {
        if c(v)=WEISS {
            π(v):= u;
            DFS-VISIT(G, v);
        }
    }
    c(u):=SCHWARZ; t:=t+1; f(u):=t;
}
```

```
DFS(G) {
    for u∈V {
        c(u):=WEISS; π(u):=NIL; K(u)=0;
    }
    t:=0; k:=0;
    for u∈V{
        if c(u)=WEISS {
            k:=k+1;
            DFS-VISIT(G, u);
        }
    }
}
```

Wenn u und v in zwei verschiedenen Zusammenhangskomponenten liegen, existiert kein Pfad zwischen u und v. Also kann v nicht in einem der rekursiven DFS-VISIT Aufrufen von u vorkommen und umgekehrt. $\Rightarrow K(u) \neq K(v)$

Wenn u und v in der gleichen Zusammenhangskomponente liegen, existiert ein $z \in V$ auch aus der gleichen Zusammenhangskomponente, das in DFS aufgerufen wird. Da ein Pfad von z nach u und von z nach v existiert und entlang von Pfaden rekursive Aufrufe von DFS-VISIT passieren bleibt $K(z)=k=K(u)=K(v)$ gleich.

DGA Ü6

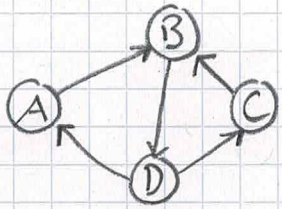
2)

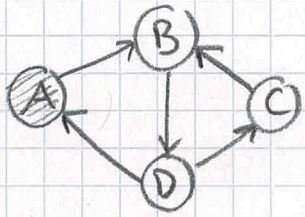
```
CNT_PATH(G,a,z) {
    c(a)=0;
    for (v ∈ Adj[a]){
      if (c(v) == -1) {
          CNT_PATH(G,v,z);
      }
      c(a)+= c(v);
    }
    return c(a);
}

ALGORITHM(G,a,z) {
    for v∈V {
      c(v)=-1;
    }
    c(z)=1;
    return CNT_PATH(G,a,z);
}
```

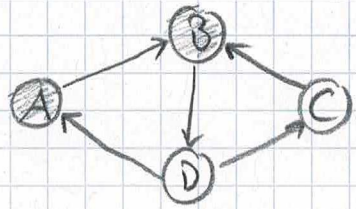
hat Aufwand $O(|V|+|E|)$ wie DFS

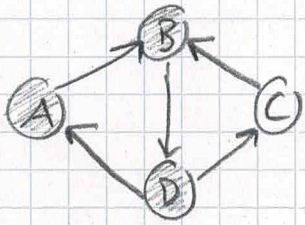
DGA Ü6

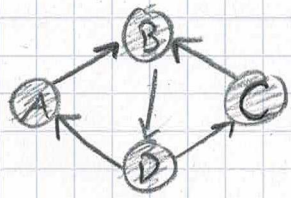
3)

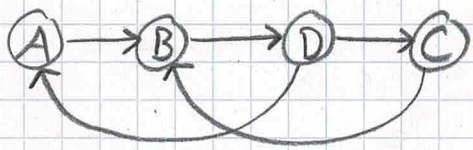










ergibt  , aber es gibt eine bessere Sortierung, nämlich