# Topologisches Sortieren

### Ziel

Knoten eines gerichteten Graphen in eine Reihenfolge bringen bei der alle Abhängigkeiten erfüllt werden.

### Wichtig

Der Graph darf nicht zyklisch sein. Also es darf keinen Weg von einem Knoten zu sich selbst geben.

## Der gezeigte Algorithmus

### Pseudo Code

```
V: Menge von Knoten
E: Menge von Kanten (start, ende)
G: Gerichteter Graph
# Achtung der algorithmus ist O(n²)
Funktion TopologischeSortierung(G = (V, E))
    L = leere Liste
    while V nicht leer do
        Zyklus = true
        for alle v in V do
            if v hat keine eingehenden Kanten then
                Zyklus = false
                Entferne v aus V
                Entferne alle Kanten die von v ausgehen aus E
                Füge v am Ende der Liste L hinzu
                print v
            endif
        endfor
        if Zyklus then
            print "Zyklus gefunden"
            break
        endif
    endwhile
end
```

### Implementation

```python
def topologischeSortierung(G):
    V, E = G
    L = []
    while len(V) > 0:
        zyklus = True
        for v in V.copy():
            if len([e for e in E if e[1] == v]) == 0:
                zyklus = False
                V.remove(v)
                E = {e for e in E if e[0] != v}
                L.append(v)
        if zyklus:
            print("Zyklus gefunden")
            break
    return L
```