zad. 2 - wykorzystanie DFS'a

S // lista zawierająca topologicznie posortowane wierzchołki

V // tablica odw. wierzchołków

```
DFS(w)
  V[w] = 1
  dla każdego sąsiada „u" wierzch. w:
      jeżeli not V[u] :
          DFS(u)
  S.push_front(w)

// algorytm
  dla każdego „w" w Grafie:
    jeżeli not V[w]:
        DFS(w)
```

, wtedy w S mamy top. posortowane wierzchołki.

D-d. nie wprost

zał. że w S nie ma top. porządku. Wtedy $\exists i,j$ t. że $i<j$ i w Grafie jest krawędź skierowana (j,i). To by znaczyło, że wierzchołek „i" musiał zostać rozwinięty po „j". Ale skoro jest krawędź (j,i) to z działania DFS'a najpierw został rozwinięty „i". Zatem ↯

algorytm ma złożoność $O(n+m)$ bo bazuje na DFS'ie