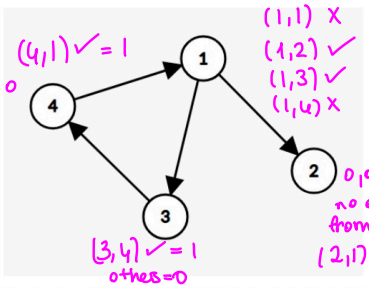
Quiz 10

1)



Which *adjacency matrix* belongs to the directed graph above?

Select one:

a.

	1	2	3	4
1	0	1	1	0
2	0	0	0	0
3	0	0	0	1
4	1	0	0	0

(1,2) (1,3) (3,4) (4,1)

b.

0	1	1	1
1	0	0	0
1	0	0	1
1	0	1	0

c.

1	1	1	0
0	1	0	0
0	0	1	1
1	0	0	1

2)

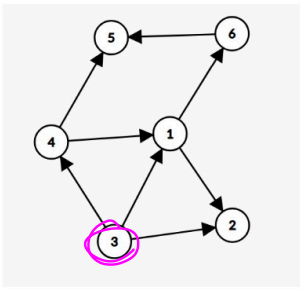
True or false: a longest (directed) path in a directed graph without directed cycles must end in a sink *(Senke)*.

Select one:

- True
- False

Rough idea: longest path doesn't end in a sink
let u be the end-vertex of the longest path P
∃ (u,v)
if v is already in path, then ∃ directed cycle contradiction
if v is not in path, P is not the longest path
add (u,v) to the end of P!

3)



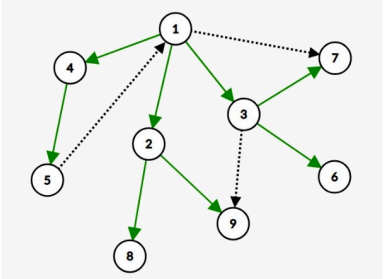
Which vertex comes first in a *topological sorting* of the graph above?

Select one:

a. 1
b. 2
c. 3
d. 4
e. 5
f. 6

v with no incoming edges, source

4)



In the graph above, the green, solid edges indicate a *depth-first search tree*. Classify the remaining edges. (*Hint: each option should be used **exactly once***).

(u,v) are ...

(1,7)	forward → on same path, goes forward doesn't form a cycle
(5,1)	back → on same path, goes back forms a cycle
(3,9)	cross → not on the same path, doesn't have a "relation"

5)

We run *depth-first search (DFS)* on a directed graph $G = (V, E)$ and determine all *pre/post numbers*.

For some edge $(u, v) \in E$, we find that

$$\text{pre}(v) < \text{pre}(u) < \text{post}(u) < \text{post}(v).$$

What can we conclude about G?

(Recall that the pre/post number of a vertex indicates the time at which it is first/last visited by the DFS algorithm)

Select one:

- a. G has a directed cycle.
- b. G does not have a directed cycle.
- c. None of the above.

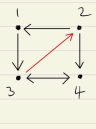
(u,v) = back edge!

What is used?

Adjazenzmatrix: $A = (A_{uv})$

$$A_{uv} = \begin{cases} 1 & \text{falls } (u,v) \in E \\ 0 & \text{sonst} \end{cases}$$

im Speicher als zweidimensionales Array



	1	2	3	4
1	0	0	1	0
2	1	0	0	1
3	0	1	0	1
4	0	0	1	0

generell: ineffizient
ausser wenn $m \geq \Omega(n^2)$

werden zeigen (per Alg.):
$\nexists$ Zyklus $\Rightarrow$ $\exists$ Senke

top. sor = task ordering

u → v

u comes before v in top. sorting = You can't complete v without completing u

post(u) < post(v) ⇒ (u,v) is a back edge

Beobachtungen

(1) ∃ back Kante ⇒ ∃ gerichteter Zyklus
(2) ∀ nicht-back $(u,v) \in E$. post(u) > post(v)
(1),(2) ⇝ $\nexists$ gerichteter Zyklus ⇒ (1) $\nexists$ back Kante
⇒ (2) umgekehrte post-order ist topologische Sortierung

kurzer Korrektheitsbeweis für DFS zum topologischen Sortieren