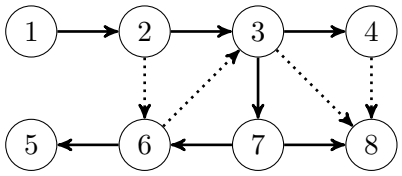
/ 4 P c) *Breadth/depth-first search:*

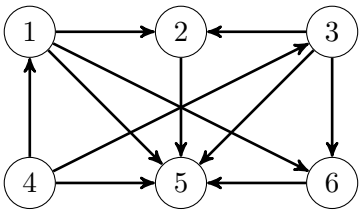
i) Consider the following directed graph:



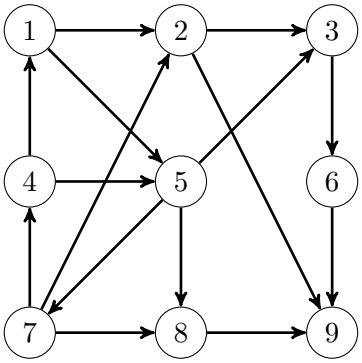
The solid edges indicate a depth-first tree. Classify the remaining edges (choose from "forward", "backward", and "cross").

- $(2, 6)$:
- $(3, 8)$:
- $(4, 8)$:
- $(6, 3)$:

ii) Does the following graph have a topological sorting? If yes, write down a topological sorting. If no, give an argument.



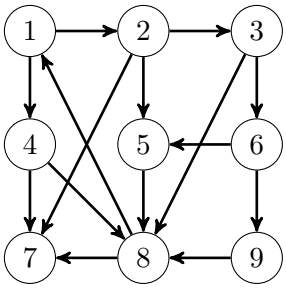
iii) Consider the following graph. Draw the breadth-first search tree resulting from a breadth-first search starting from vertex 1. Process the neighbors of a vertex in increasing order.



/ 3 P

c) *Breadth/depth-first search:*

Consider the following directed graph:



i) Draw the depth-first search tree resulting from a depth-first search starting from vertex 1. Process the neighbors of a vertex in increasing order.

ii) Compute the distances of the vertices listed below from vertex 2 (i.e. the length of the shortest path from vertex 2 to every vertex listed below). Fill in the numbers in the table below. You don't need to justify your answer.

Vertex	Distance
1	
4	
7	

Vertex	Distance
8	
9	

iii) Does the graph have a topological sorting? Check the corresponding box below.

☐ The graph has one or more topological sorting(s).

☐ The graph does not have a topological sorting.

If the graph contains one or more topological sorting(s), additionally give a topological sorting. If not, give an edge set of minimal size such that the graph has a topological sorting after removing these edges from the graph (the vertex set stays the same).

Topological sorting / Minimal edge set to remove: