Exo 8

Problem 1

1.1. Algorithm to compute rev(G) in $O(V+E)$

For each directed edge $(u \to v)$, add a reversed edg
$(v \to u)$.

```
for each u in V:
    for each v in Adj[u]:
        revAdj[v].add(u)
return revAdj
```

We scan every vertex and every edge once $O(V+E)$

1.2

If the SCC graph had a directed cycle

$C1 \to C2 \to \dots \to C1$

then all vertices in these components would be mutually reachable.

They would form one SCC, not several.

Therefore, SCC graph has no cycles: it is alway a DAG

1.3 Why scc(rev(G)) = rev(scc(G))

Reversing the graph doesn't change which vertices are mutually reachable.

SCC partition is the same.

Each edge between SCCs is reversed ($u \to v$ becomes $v \to u$).

The SCC graph is simply reversed

Thus

$$scc(rev(G)) = rev(scc(G))$$

1.4

u reaches v in G if $S(u)$ reaches $S(v)$ in $scc(G)$

Reason:

Any path $u \to \ldots \to v$ crosses SCCs in the same order gives a path in the SCC graph.

Any path between components can be expanded into real path inside G because each SCC is strongly connected.

Problem 2 (Euler Tour)

2.1

A strongly connected directed graph has an Euler tour if

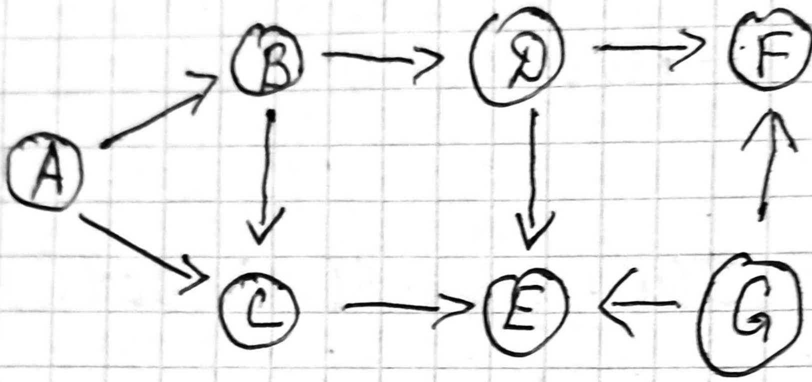
$$\text{in degree}(v) = \text{out-degree}(v) \ \forall v$$

If a tour exists, every time we enter a vertex we must leave it $\to$ in = out

If graph is strongly connected and in = out everywhere — Euler's theorem guarantees an Euler tour.

Problem 3

A -> B , A -> C , B -> C , B -> D , C -> E , D -> E ,
D - F , G -> F , G -> E



Sources: A, G

We start with A

Valid topological orders

A, B, C, D, G, E, F
A, B, C, D, G, F, E

} Both respect all arrow direction

Order starting with G

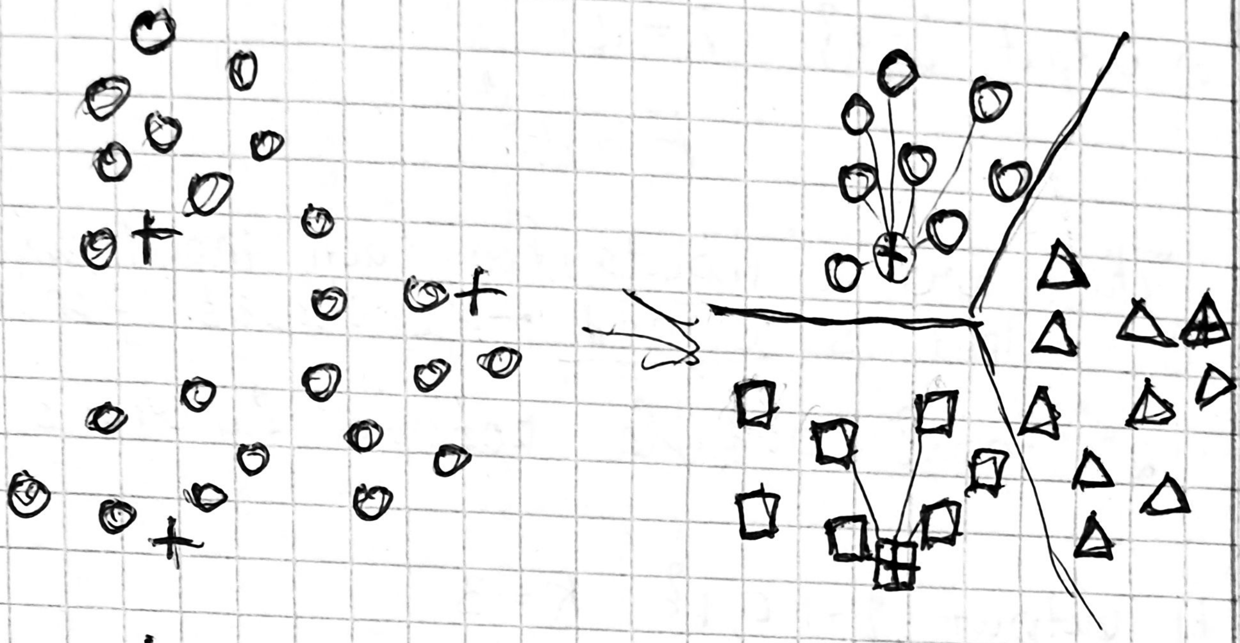
Valid orders

G, A, B, C, D, E, F

G, A, B, C, D, F, E

Project 2

K - means clustering



Init

update 1

The cat sat on the mat
The dog sat on the log
Cats and dogs are pets