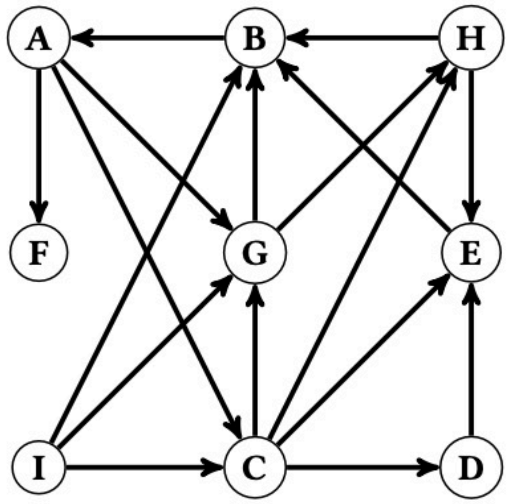
DFS - Exercise Sheet Question

Exercise 10.2 *Depth-first search* **(1 point).**

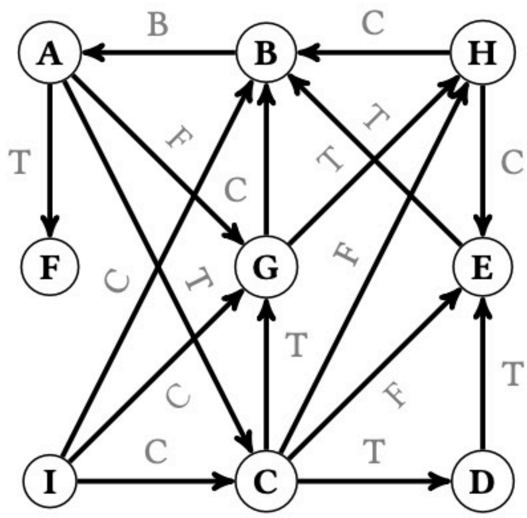
Execute a depth-first search (*Tiefensuche*) on the following graph. Use the algorithm presented in the lecture. Always do the calls to the function "visit" in alphabetical order, i.e. start the depth-first search from A and once "visit(A)" is finished, process the next unmarked vertex in alphabetical order. When processing the neighbors of a vertex, also process them in alphabetical order.



(a) Mark the edges that belong to the depth-first forest (*Tiefensuchwald*) with a "T" (for tree edge).

Solution:

In the following, both the solution to subtask (a) and the solution to subtask (d) are showed.



(b) For each vertex in the depth-first forest, give its *pre*- and *post*-number.

Solution:

A(1,16) B(5,6) C(2,13) D(3,8) E(4,7) F(14,15) G(9,12) H(10,11) I(17,18).

(c) Give the vertex ordering that results from sorting the vertices by pre-number. Give the vertex ordering that results from sorting the vertices by post-number.

Solution:

Pre-ordering: A, C, D, E, B, G, H, F, I.

Post-ordering: B, E, D, H, G, C, F, A, I.

(d) Mark every forward edge (*Vorwärtskante*) with an “F”, every backward edge (*Rückwärtskante*) with a “B”, and every cross edge (*Querkante*) with a “C”.

Solution:

See above in the solution to part (a).

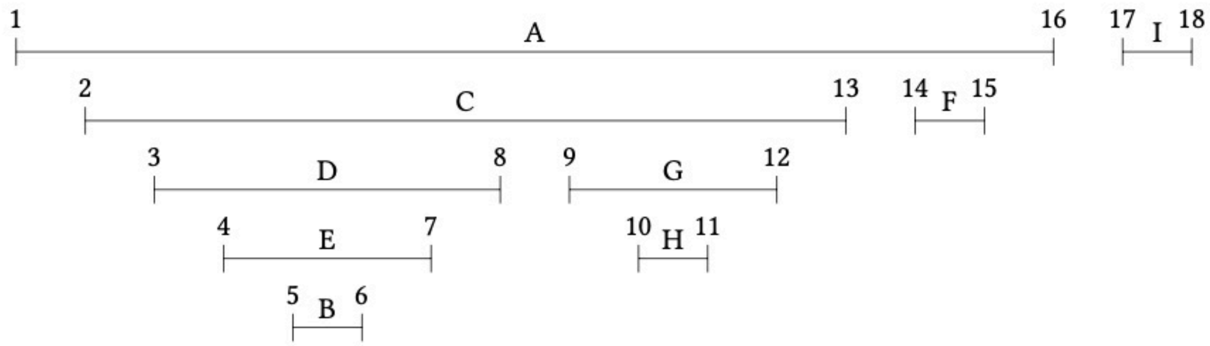
(e) Does the above graph have a topological ordering? If yes, write down the topological ordering we get from the above execution of depth-first search; if no, argue how we can use the above execution of depth-first search to find a directed cycle.

Solution:

The decreasing order of the post-numbers gives a topological ordering whenever the graph is acyclic. This is the case if and only if there are no back edges. If there is a back edge, then together with the tree edges between its end points it forms a directed cycle. In our graph, the only back edge is B $\to$ A, and the tree edges from A to B are A $\to$ C, C $\to$ D, D $\to$ E and E $\to$ B. Together they form the directed cycle (A $\to$ C $\to$ D $\to$ E $\to$ B $\to$ A).

(f) Draw a scale from 1 to 18, and mark for every vertex v the interval I_v from pre-number to post-number of v. What does it mean if $I_u \subset I_v$ for two different vertices u and v?

Solution:



If $I_u \subset I_v$ for two different vertices u and v, then u is visited during the call of visit(v).

(g) Consider the graph above where the edge from B to A is removed and an edge from F to I is added. How does the execution of depth-first search change? Does the graph have a topological ordering? If yes, write down the topological ordering we get from the execution of depth-first search; if no, argue how we can use the execution of depth-first search to find a directed cycle. If you sort the vertices by *pre-number*, does this give a topological sorting?

Solution:

The execution of the depth-first search only changes in the last step, where I is visited from F instead of starting the call of “visit(I)” after completing “visit(A)”.

This gives the following post-ordering: B, E, D, H, G, C, I, F, A. Since the graph has no back edges anymore, it has a topological ordering. The topological ordering we get from the execution of the depth-first search (reversed post-ordering) is: A, F, I, C, G, H, D, E, B.

The pre-ordering is A, C, D, E, B, G, H, F, I; it does not give a topological ordering, since there is for example the edge (G, B) in the graph.

Guidelines for correction:

The following 5 elements are important in this exercise. If all of them are solved correctly, award 1 point. If at least 3 are solved correctly, award 1/2 point.

- Labeling the graph in parts (a) and (d).
- Determining the pre- and post numbers as well as the pre- and post orderings in parts (b) and (c).
- Finding a directed cycle in part (e) using the execution of the DFS.
- Mentioning what it means if $I_u \subset I_v$ in part (f).
- Answering the three question in part (g).

Graph Definitions

Definition 1. Let $G = (V, E)$ be a graph.

- For $v \in V$, the **degree** $\deg(v)$ of v (german "Knotengrad") is the number of edges that are incident to v.
- A sequence of vertices $(v_0, v_1, \ldots, v_k)$ (with $v_i \in V$ for all i) is a **walk** (german "Weg") if $\{v_i, v_{i+1}\}$ is an edge for each $0 \leq i \leq k-1$. We say that v_0 and v_k are the **endpoints** (german "Startknoten" and "Endknoten") of the walk. The **length** of the walk $(v_0, v_1, \ldots, v_k)$ is k.
- A sequence of vertices $(v_0, v_1, \ldots, v_k)$ is a **closed walk** (german "Zyklus") if it is a walk, $k \geq 2$ and $v_0 = v_k$.
- A sequence of vertices $(v_0, v_1, \ldots, v_k)$ is a **path** (german "Pfad") if it is a walk and all vertices are distinct (i.e., $v_i \neq v_j$ for $0 \leq i < j \leq k$).
- A sequence of vertices $(v_0, v_1, \ldots, v_k)$ is a **cycle** (german "Kreis") if it is a closed walk, $k \geq 3$ and all vertices (except v_0 and v_k) are distinct.
- A **Eulerian walk** (german "Eulerweg") is a walk that contains every edge exactly once.
- A **closed Eulerian walk** (german "Eulerzyklus") is a closed walk that contains every edge exactly once.
- A **Hamiltonian path** (german "Hamiltonpfad") is a path that contains every vertex.
- A **Hamiltonian cycle** (german "Hamiltonkreis") is a cycle that contains every vertex.
- For $u, v \in V$, we say u **reaches** v (or v is **reachable** from u; german "u erreicht v") if there exists a walk with endpoints u and v.
- A **connected component** of G is an equivalence class of the (equivalence) relation defined as follows: Two vertices $u, v \in V$ are equivalent if u reaches v.
- A graph G is **connected** (german "zusammenhängend") if for every two vertices $u, v \in V$ u reaches v or equivalently if there is only one connected component.
- A graph G is a **tree** (german "Baum") if it is connected and has no cycles.

Graph Definitions – Graph Quiz (exam question)

/ 5 P

c) *Graph quiz:* For each of the following claims, state whether it is true or false. You get 1P for a correct answer, -1P for a wrong answer, 0P for a missing answer. You get at least 0 points in total.

As a reminder, here are a few definitions for a (directed) graph $G = (V, E)$:

For $k \geq 2$, a (directed) *walk* is a sequence of vertices $v_1, \ldots, v_k$ such that for every two consecutive vertices v_i, v_{i+1}, we have $\{v_i, v_{i+1}\} \in E$ (resp. $(v_i, v_{i+1}) \in E$ for a directed walk).

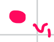
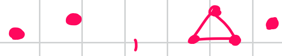
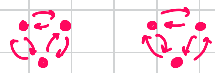
A (directed) *closed walk* is a (directed) walk with $v_1 = v_k$.

A (directed) *cycle* is a (directed) closed walk where $k \geq 3$ and all vertices (except v_1 and v_k) are distinct.

A (directed) *closed Eulerian walk* is a (directed) closed walk which traverses every edge in E exactly once.

For a vertex v in a directed graph $G = (V, E)$, the *in-degree* of v is the number of edges in E that end in v (i.e., of the form (w, v)), and the *out-degree* of v is the number of edges in E that start in v (i.e., of the form (v, w)).

Claim	**true**	**false**
A connected graph must contain a cycle.	☐	☐
A graph $G = (V, E)$ with $\lvert E\rvert \leq \lvert V\rvert - 1$ is a tree.	☐	☐
Let $G = (V, E)$ be a graph with $\lvert E\rvert \geq 4$, which contains a closed Eulerian walk. If we remove one edge from E, the resulting graph does not contain a closed Eulerian walk, no matter which edge we remove (the vertex set does not change).	☐	☐
Let $G = (V, E)$ be a *directed* graph. If the in-degree and out-degree of every vertex $v \in V$ is even, then G contains a *directed* closed Eulerian walk.	☐	☐

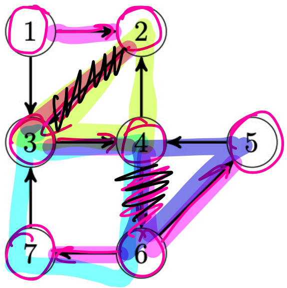
	T/F	Justification
A connected graph must contain a cycle.	False	Counterex. : Tree , 
A graph $G = (V, E)$ with $\lvert E\rvert \leq \lvert V\rvert - 1$ is a tree.	False	Counterex. : 
Let $G = (V, E)$ be a graph with $\lvert E\rvert \geq 4$, which contains a closed Eulerian walk. If we remove one edge from E, the resulting graph does not contain a closed Eulerian walk, no matter which edge we remove (the vertex set does not change).	True	Remove any edge $\{u, v\}$ deg(u) and deg(v) will decrease by one Initially G contained a closed eulerian w. ⇒ Every vertex in G had even degree deg(u) and deg(v) are odd All vertices don't have even deg ⇒ G can't contain a closed eulerian walk
Let $G = (V, E)$ be a *directed* graph. If the in-degree and out-degree of every vertex $v \in V$ is even, then G contains a *directed* closed Eulerian walk.	False	Counterex. : 

DFS - Exam Questions

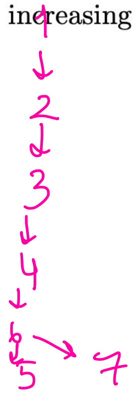
top. ordering:

1, 6, 7, 5, 3, 4, 2

/ 2 P d) *Depth-first search:* Consider the following directed graph:



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



ii) Write out two edges e_1, e_2 such that the directed graph above has a topological ordering after removing e_1 and e_2 (the vertex set does not change).

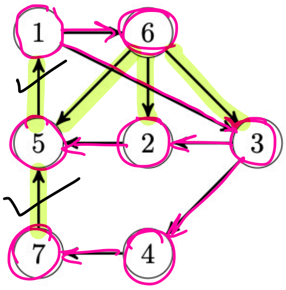
Remark: There could be multiple valid solutions. In this case, you only need to write down one of them.

(4,6)

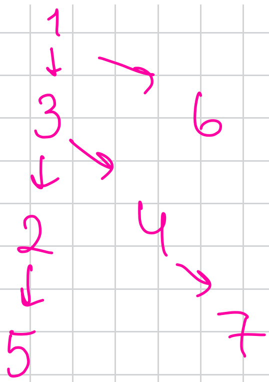
(2,3)

/ 2 P

d) *Depth-first search:* Consider the following directed graph:



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



ii) Write out all the cross edges and all the back edges (specify which ones are cross edges, and which ones are back edges).

(5,1) back

(7,5) cross

$\left.\begin{array}{l}(6,5)\\(6,2)\\(6,3)\end{array}\right\}$ cross

Topological Sorting - Exam Question

/ 3 P d) *Directed Acyclic Tournament*

A *tournament* is a directed graph $G = (V, E)$ such that:

- G has no self loops, i.e., $(v, v) \notin E$, for all $v \in V$. (Note that the graphs that we usually consider have no self loops.)
- For every two distinct vertices $u, v \in V$, either $(u, v) \in E$ or $(v, u) \in E$ but not both.

Let G be a directed acyclic graph that is also a tournament. Show that G has a unique topological sorting.

Since G is a DAG, it has at least one top. sorting $(v_1, \dots, v_n)$

G is also a tournament $\Rightarrow$ For all i $1 \le i < n$ either $(v_i, v_{i+1}) \in E$ or (v_{i+1}, v_i) (but not both)

Since $(v_1 \dots v_n)$ is a top. sorting, the edge between v_i and v_{i+1} must be (v_i, v_{i+1}). (It cannot be (v_{i+1}, v_i))

$(v_1, \dots, v_n)$ is a path in G

v_i is before v_{i+1} in any top. sorting

$(v_1, \dots, v_n)$ is the unique top. sorting !

DP Mini Exam - Proof at the end

In this problem, you are given an array $A = [a_1, \dots, a_n]$ of n pairwise distinct positive integers, i.e. $a_i \in \mathbb{Z}_{\geq 1}$ for $i \in \{1, \dots, n\}$ and $a_i \neq a_j$ for $i \neq j \in \{1, \dots, n\}$.

i $\Leftrightarrow$ ii

/ 2 P b) Show that for any array A as above, the following two conditions are equivalent.

i) There are non-empty sets $I_1, I_2 \subseteq \{1, \dots, n\}$ such that $\sum_{i \in I_1} a_i = \sum_{i \in I_2} a_i$ and $I_1 \neq I_2$.

ii) There are non-empty sets $I_1, I_2 \subseteq \{1, \dots, n\}$ such that $\sum_{i \in I_1} a_i = \sum_{i \in I_2} a_i$ and $I_1 \cap I_2 = \emptyset$.

ii $\Rightarrow$ i)

Assume there are non-empty sets $I_1, I_2 \subseteq \{1 \dots n\}$ s.t. $\sum_{i \in I_1} a_i = \sum_{i \in I_2} a_i$ and $I_1 \cap I_2 = \emptyset$

$\Rightarrow I_1 \neq I_2$

| I_1 and I_2 are non-empty + $I_1 \cap I_2 = \emptyset$
I_1 has at least one element that's not in I_2.

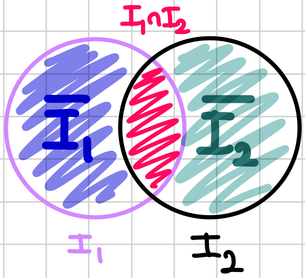
I_1 and I_2 also satisfies this

i $\Rightarrow$ ii)

Assume there are non-empty sets $I_1, I_2 \subseteq \{1 \dots n\}$ s.t. $\sum_{i \in I_1} a_i = \sum_{i \in I_2} a_i$ and $I_1 \neq I_2$

We construct $\overline{I_1}$ and $\overline{I_2}$ s.t. $\overline{I_1}, \overline{I_2} \subseteq \{1 \dots n\}$ are non-empty, $\sum_{i \in \overline{I_1}} a_i = \sum_{i \in \overline{I_2}} a_i$, $\overline{I_1} \cap \overline{I_2} = \emptyset$

$\overline{I_1} := I_1 \setminus (I_1 \cap I_2)$



$\overline{I_2} := I_2 \setminus (I_1 \cap I_2)$

$I_1 \neq I_2 \Rightarrow \overline{I_1} \neq \overline{I_2}$

| we just removed the common elements, they have to have some uncommon elements s.t. $I_1 \neq I_2$ can hold

$\overline{I_1} \cap \overline{I_2} = \emptyset$: By def we've removed all common elements from both. The intersection is definitely $\emptyset$

$\sum_{i \in \overline{I_1}} a_i = \sum_{i \in \overline{I_2}} a_i$:

$$\sum_{i \in \overline{I_1}} a_i = \sum_{i \in I_1} a_i - \sum_{i \in I_1 \cap I_2} a_i$$

$$= \sum_{i \in I_2} a_i - \sum_{i \in I_1 \cap I_2} a_i \qquad | \sum_{i \in I_1} a_i = \sum_{i \in I_2} a_i$$

$$= \sum_{i \in \overline{I_2}} a_i$$

$\overline{I_1}$ and $\overline{I_2}$ are non-empty : All $a_i > 0$ and $\overline{I_1} \neq \overline{I_2}$ $\Rightarrow$ $\overline{I_1}$ and $\overline{I_2}$ are non-empty