Eidgenössische
Technische Hochschule
Zürich

Ecole polytechnique fédérale de Zurich
Politecnico federale di Zurigo
Federal Institute of Technology at Zurich

Departement of Computer Science
Johannes Lengler, Markus Püschel, David Steurer
Kasper Lindberg, Kostas Lakis, Lucas Pesenti, Manuel Wiedmer

10 November 2025

Algorithms & Data Structures   Exercise sheet 8   HS 25

The solutions for this sheet are submitted on Moodle until 16 November 2025, 23:59.

Exercises that are marked by * are challenge exercises. They do not count towards bonus points.

You can use results from previous parts without solving those parts.

We first list some 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 $0 \le i \le k-1$. 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, $k \ge 2$ 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., $v_i \ne v_j$ for $0 \le i < j \le k$).
- A sequence of vertices $(v_0, v_1, \ldots, v_k)$ is a **cycle** (german "Kreis") if it is a closed walk, $k \ge 3$ and all vertices (except v_0 and v_k) are distinct.
- An **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, or equivalently, there exists a path with endpoints u and v.
- A **connected component** of G (german "Zusammenhangskomponente") 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.

Exercise 8.1 *Party & Beer & Party & Beer* **(1 point).**

For your birthday, you organize a party and invite some friends over at your place. Some of your friends bring their partners, and it turns out that in the end everybody (including yourself) knows exactly 7 other people at the party (note that the relation of knowing someone is commutative, i.e. if you know someone then this person also knows you and vice versa). Show that there must be an even number of people at your party.

Exercise 8.2 *UNO.*

Consider a deck where we have $n \geq 2$ different numbers and $m \geq 2$ different colors, leading to a total of nm cards. Your friend proposes the following game. They will shuffle the cards and then you will draw $X \geq 2$ cards from the top of the deck. Afterwards, you can start playing the cards. The first card you play can be any card. From the second card onwards, it must hold that the card you play has either the same number or the same color as the last card you played. That is, you can play a red 8 on a red 6 and a blue 5 on a green 5, but not a yellow 4 on a pink 3. If you manage to play all the cards from your hand, you win. Then, your friend has to pay you 1 CHF. If you cannot do this, you have to pay them instead.

While your friend is in general a nice person, you do not trust their shuffling. That is, they may as well put the cards in any order and you would not know. What is a value of X for which you definitely should not play the game? For example, you should not play if $X = 2$, as your friend might choose a red 1 and a blue 2. If $X = nm$, then you should play (you can lay the cards in a "snake" order).

Remark. It is possible to come up with values $V_1(n, m)$ for the first question and $V_2(n, m)$ for the second such that they differ by 1.

a) Come up with the largest value of X (depending on n and m) for which you can prove that you should not play, and give an argument for why this is the case.

b)* Come up with the smallest value of X for which you can prove that you should indeed play (i.e. no matter which cards you receive, you will be able to play them). Prove it by induction on n.[1] The induction hypothesis should be "Any subset of at least $V(n, m)$ cards from a deck with n numbers and m colors is playable", where $V(n, m)$ is the value you came up with for the number of cards required to guarantee playability.

Exercise 8.3 *Graph connectivity* **(2 points)**.

In this exercise, you will need to prove or find counterexamples to some statements about the connectivity of graphs. We first need to introduce/recall a few definitions.

Definition 2. A *cycle* is a sequence of vertices $v_1, \ldots, v_k, v_{k+1}$ with $k \geq 3$ such that all $v_1, \ldots, v_k$ are distinct, $v_1 = v_{k+1}$ and such that any two consecutive vertices are adjacent. We say that such a cycle has length k.

Definition 3. A graph is *connected* if there is a walk between every pair of vertices. It is called *disconnected* otherwise.

Definition 4. A vertex v in a connected graph is called a *cut vertex* (or *articulation point*) if the subgraph obtained by removing v (and all its incident edges) is disconnected.

[1]Assume $n \geq m$, which is justified by symmetry of the problem.

Definition 5. An edge e in a connected graph is called a *cut edge* (or *bridge*) if the subgraph obtained by removing e (but keeping all the vertices) is disconnected.

In the following, we always assume that the original graph is connected. Prove or find a counterexample to the following statements:

a) If a vertex v is part of a cycle, then it is not a cut vertex.

b) If a vertex v is not a cut vertex, then v must be part of a cycle.

c) If an edge e is part of a cycle (i.e. e connects two consecutive vertices in a cycle), then it is not a cut edge.

d) If an edge e is not a cut edge, then e must be part of a cycle.

e) If u and v are two adjacent cut vertices, then the edge $e = \{u, v\}$ is a cut edge.

f) If $e = \{u, v\}$ is a cut edge, then u and v are cut vertices. What if we add the condition that u and v have degree at least 2 ?

Exercise 8.4 *Introduction to Trees.*

In this exercise the goal is to prove a few basic properties of trees (for the definition of a tree, see Definition 1).

(a) A **leaf** is a vertex with degree 1. Prove that in every tree G with at least two vertices there exists a leaf.

__Hint:__ Consider the longest path in G. Prove that its endpoint is a leaf.

(b) Prove that every tree with n vertices has exactly $n - 1$ edges.

__Hint:__ Prove the statement by using induction on n. In the induction step, use part (a) to find a leaf. Disconnect the leaf from the tree and argue that the remaining subgraph is also a tree. Apply the induction hypothesis and conclude.

(c) Prove that a graph with n vertices is a tree if and only if it has $n - 1$ edges and is connected.

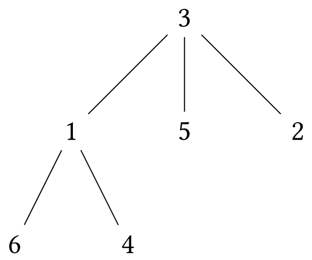
__Hint:__ One direction is immediate by part (b). For the other direction (every connected graph with $n-1$ edges is a tree), use induction on n. First, prove there always exists a leaf by considering the average degree. Then, disconnect the leaf from the graph and argue that the remaining graph is still connected and has exactly one edge less. Apply the induction hypothesis and conclude.

(d) Write the pseudocode of an algorithm that is given a graph G as input and checks whether G is a tree.

As input, you can assume that the algorithm has access to the number of vertices n, the number of edges m, and to the edges $\{a_1, b_1\}, \{a_2, b_2\}, \ldots, \{a_m, b_m\}$ (i.e., the algorithm has access to $2m$ integers $a_1, \ldots, a_m, b_1, \ldots, b_m$, where each edge of G is given by its endpoints a_i and b_i). You can assume that the graph is valid (specifically, $1 \leq a_i, b_i \leq n$ and $a_i \neq b_i$). The algorithm outputs "YES" or "NO", corresponding to whether G is a tree or not. Your algorithm must always complete in time polynomial in n (e.g., even $O(n^{10}m^{10})$ suffices). You do not have to show the correctness of your algorithm or what the running time of your algorithm is.

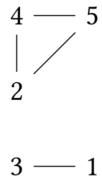
__Hint:__ Use part (c). There exists a (relatively) simple $O(n + m)$ solution. However, the official solution is $O(n \cdot m)$ for brevity and uses recursion to check if G is connected.

Example 1: $n = 6$
$m = 5$
$a_1, b_1 = 1, 3$
$a_2, b_2 = 6, 1$
$a_3, b_3 = 3, 5$
$a_4, b_4 = 2, 3$
$a_5, b_5 = 4, 1$



Output: YES

Example 2: $n = 5$
$m = 4$
$a_1, b_1 = 1, 3$
$a_2, b_2 = 4, 5$
$a_3, b_3 = 5, 2$
$a_4, b_4 = 2, 4$

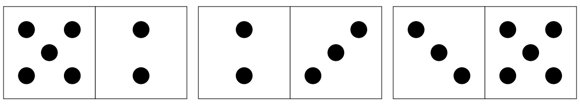


Output: NO

Exercise 8.5 *Domino.*

(a) A domino set consists of all possible $\binom{6}{2} + 6 = 21$ different tiles of the form $[x|y]$, where x and y are numbers from $\{1, 2, 3, 4, 5, 6\}$. The tiles are symmetric, so $[x|y]$ and $[y|x]$ is the same tile and appears only once.

Show that it is impossible to form a line of all 21 tiles such that the adjacent numbers of any consecutive tiles coincide like in the example below.



(b) What happens if we replace 6 by an arbitrary $n \geq 2$? For which n is it possible to line up all $\binom{n}{2} + n$ different tiles along a line?