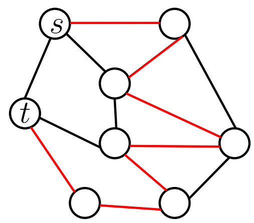
COMP 3804 — Tutorial March 4, 2026

Problem 1: A *Hamilton path* in an undirected graph is a path that contains every vertex exactly once. In the figure below, you see a Hamilton path in red. A *Hamilton cycle* is a cycle that contains every vertex exactly once. In the figure below, if you add the black edge $\{s, t\}$ to the red Hamilton path, then you obtain a Hamilton cycle.



If $G = (V, E)$ is an undirected graph, then the graph G^3 is defined as follows:

1. The vertex set of G^3 is equal to V.
2. For any two distinct vertices u and v in V, $\{u, v\}$ is an edge in G^3 if and only if there is a path in G between u and v consisting of at most three edges.

Problem 1.1: Describe a *recursive* algorithm HamiltonPath that has the following specification:

Algorithm HamiltonPath(T, u, v)**:**
Input: A tree T with at least two vertices; two distinct vertices u and v in T such that $\{u, v\}$ is an edge in T.
Output: A Hamilton path in T^3 that starts at vertex u and ends at vertex v.

Hint: You do not have to analyze the running time. The base case is easy. Now assume that T has at least three vertices. If you remove the edge $\{u, v\}$ from T, then you obtain two trees T_u (containing u) and T_v (containing v).

1. One of these two trees, say, T_u, may consist of the single vertex u. How does your recursive algorithm proceed?
2. If each of T_u and T_v has at least two vertices, how does your recursive algorithm proceed?

Problem 1.2: Prove the following lemma:

Lemma: For every tree T that has at least three vertices, the graph T^3 contains a Hamilton cycle.

Problem 1.3: Prove the following theorem:

Theorem: For every connected undirected graph G that has at least three vertices, the graph G^3 contains a Hamilton cycle.

Algorithm DIJKSTRA(G, s):
for each $v \in V$
do $d(v) = \infty$
endfor;
$d(s) = 0$;
$S = \emptyset$;
$Q = V$;
while $Q \neq \emptyset$
do u = vertex in Q for which $d(u)$ is minimum;
delete u from Q;
insert u into S;
for each edge (u, v)
do if $d(u) + wt(u, v) < d(v)$
then $d(v) = d(u) + wt(u, v)$
endif
endfor
endwhile

Problem 2: Let $G = (V, E)$ be a directed graph, in which each edge (u, v) has a positive weight $wt(u, v)$, and let s be a source vertex in V.

We have seen in class that Dijkstra's algorithm can be implemented using a min-heap. During this algorithm, a sequence of EXTRACT_MIN-operations and DECREASE_KEY-operations is executed.

Prove that during any such sequence of operations, the smallest element in the min-heap never decreases. Thus, if at some moment, the smallest element is 45, then later on, the smallest value will always be at least 45.

Problem 3: Let $G = (V, E)$ be an undirected connected graph, in which each edge $\{u, v\}$ has a weight $wt(u, v)$. Assume that all edge weights in G are distinct. Let $e_1, e_2, \ldots, e_m$ be the sequence of edges of G, sorted in increasing order of their weights wt.

Let $G' = (V, E)$ be a copy of G, in which each edge $\{u, v\}$ has a weight $wt'(u, v)$. Assume that all edge weights in G' are distinct. Assume that $e_1, e_2, \ldots, e_m$ is also the sequence of edges of G', sorted in increasing order of their weights wt'.

Prove that G and G' have the same minimum spanning tree.