34.5-1 First we prove that Subgraph-isomorphism is in NP. Let \((G_1, G_2)\) be an instance. Assume the nodes of each graph are numbered 1, 2, \ldots, \(|G|\). Let \(x\) be a witness, namely a list of nodes \(i_1, i_2, \ldots, i_n\) where \(n = |G_1|\).

Verifying that \(G_1\) is isomorphic to a subgraph of \(G_2\) now means verifying that the subgraph induced by the node list \(x\) is equal to \(G_1\). This can be done by for each \(i_k, i_l\) verify that \((i_k, i_l) \in E_1\) if and only if \((k, l) \in E_2\). Since there are \(O(n^2)\) such pairs, the check takes polynomial time. This proves that Subgraph-isomorphism is in NP.

Note: It is important the witness actually is given as a node list, not as a set, since determining whether two graphs are isomorphic is believed to be hard in general.

We now show that Subgraph-isomorphism is NP-hard. We show this by using Subgraph-isomorphism to solve the known NP-complete problem Clique. Let \((G, k)\) be an instance of clique where we want to determine whether the graph \(G\) contains a clique of size \(k\). We create the instance of Subgraph-isomorphism \((G, G_k)\), where \(G_k\) is the complete graph with \(k\) nodes.

We show that these two instances are equal.

If \(G\) has a \(k\)-clique, then \(G\) has the complete \(k\)-graph as subgraph, and the instance \((G, G_k)\) is also a yes-instance.

If \(G\) has \(G_k\) as subgraph, then it has a \(k\)-clique as subgraph, and hence \((G, k)\) is a yes-instance of Clique.

We conclude that Subgraph-isomorphism is NP-hard and hence NP-complete.

34.5-7 We begin by verifying that Longest-simple-cycle is in NP. Let \((G, k)\) be an instance where we want to determine whether or not there exists a cycle of length \(k\). Let \(x\) be a list of nodes. It can verified in polynomial time that \(x\) is indeed a cycle, that no vertex is in \(x\) more than once and that the length of \(x\) is \(k\) or greater. Hence Longest-simple-cycle is in NP.

We now show that Longest-simple-cycle is NP-hard by showing how to reduce Hamiltonian-cycle to Longest-simple-cycle. Let \(G\) be an instance of Hamiltonian-cycle. We let \((G, |G|)\) be an instance of Longest-simple-cycle. It is clear that \(G\) has a Hamiltonian cycle exactly when \(G\) has a cycle of length \(|G|\). Hence Longest-simple-cycle is NP-hard.
These two results together show that \textsc{Longest-Simple-Cycle} is NP-complete.

- \textbf{Show that \textsc{Hamiltonian-path} is NP-complete.} A graph has a Hamiltonian path if there is a path that passes each node exactly once.

In \textsc{NP}: We can in polynomial time verify that a given path is indeed a Hamiltonian path. We leave out the details.

\textsc{NP-hard}: We reduce from \textsc{Hamiltonian-cycle}, which is known to be \textsc{NP-complete}. We want to construct the reduction by for a given graph $G = (V, E)$ construct a graph $G' = (V', E')$ such that $G'$ has a Hamiltonian path exactly when $G$ has a Hamiltonian cycle. If we can do that, we can construct an algorithm that solves \textsc{Hamiltonian-cycle} by using \textsc{Hamiltonian-path} a sub-routine. This would prove that \textsc{Hamiltonian-path} is at least as hard \textsc{Hamiltonian-cycle} (up to a polynomial factor).

The construction is as follows. Let $u \in V$ be an arbitrary vertex. We add a copy of $u$ (together with all edges) and call it $u'$. Then we add two vertices $v$ and $v'$ and the edges $(u, u')$ and $(v, v')$. We call the result $G'$. We now verify that $G'$ has the properties we need. Assume $G$ has a Hamiltonian cycle. Then $G'$ has a path starting in $u$ and ending in $u'$. We can extend this path to a Hamiltonian path by adding the edges $(u, u')$ and $(v, v')$.

Now assume that $G'$ has a Hamiltonian path. Obviously this path must have $v$ and $v'$ as end-points, which implies that there must be a path from $u$ to $u'$. We can then close the path by letting removing the edge to $u'$ and let it go to $u$ instead. (From the construction of $u'$ this is always possible.)

We have proved that the reduction is correct and that \textsc{Hamiltonian-path} is \textsc{NP-complete}.

- \textbf{We now consider the problem \textsc{Minimum-Degree-Spanning-Tree}:} Given a graph $G$ and an integer $k$, decide whether $G$ contains a spanning tree $T$ such that the degree of each node in $T$ is at most $k$.

We show that this problem is in \textsc{NP}. Let $(G, k)$ be an instance and let the witness $x$ be a spanning tree. We need to check that $x$ is indeed a tree and that the degree of the nodes in $x$ don’t exceed $k$. This can be done in polynomial time.

We now give a reduction from \textsc{Hamiltonian-path}, which is known \textsc{NP-complete} from the previous task. Let $G$ be an instance of \textsc{Hamiltonian-}
PATH. We let \((G, 2)\) be an instance \textsc{Minimum-degree-spanning-tree}. We claim that \(G\) has a Hamiltonian path exactly when \(G\) has a spanning tree of degree at most 2. Assume such a spanning tree exists. With degree 2 it cannot branch, and since the tree spans the graph, only two vertices can have degree 1. Therefore this tree is indeed a Hamiltonian path. On the other hand, if the graph has a Hamiltonian path, this path is obviously a spanning tree of degree 2. Therefore the reduction is correct.

We can conclude that \textsc{Minimum-degree-spanning-tree} is NP-complete.