Exercise 1: One-minute-long exercise
Prove that any language $L \subset \{0, 1\}^*$ that is neither empty nor $\{0, 1\}^*$ is hard for $\text{NL}$ for polynomial-time reductions.

Exercise 2: Graph representation and why it does not matter. Let $\Sigma = \{0, 1, /, \cdot, \#\}$ with $\#$ the end-of-word symbol. For a directed graph $G = (V, E)$ with $V = [0, n - 1]$ for some $n \in \mathbb{N}$ and $E \subseteq V \times V$, we consider the following two representations of $G$ by a word in $\Sigma^*$:

- By its adjacency matrix $m_G \in \Sigma^*$:
  $$m_G \overset{\text{def}}{=} m_{0,0} m_{0,1} \cdots m_{0,n-1} \cdot \cdots \cdot m_{n-1,0} \cdots m_{n-1,n-1} \#$
  where for all $0 \leq i, j < n$, $m_{i,j}$ is 1 if $(i, j) \in E$, 0 otherwise.

- By its adjacency list $l_G \in \Sigma^*$:
  $$l_G \overset{\text{def}}{=} k_0^0 / \cdots / k_{m_1}^0 \cdot \cdots \cdot k_{m_i}^{n-1} / \cdots / k_{m_{n-1}}^{n-1} \#$
  where for all $0 \leq i < n$, $k_0^i, \ldots, k_{m_i}^i$ are binary words listing the (codes of) right neighbors of vertex $i$.

1. Show that it is possible to check in logarithmic space that a word $w \in \Sigma^*$ is a well-formed description of a graph (for any of the two representations).

2. Describe a logarithmic space bounded deterministic Turing machine taking as input a graph $G$, represented by its adjacency matrix, and computing the adjacency list representation of $G$.

Exercise 3: A few $\text{NL}$-complete problems
Show that the following problems are $\text{NL}$-complete for logspace reductions (you may use the fact that REACH is $\text{NL}$-hard for logspace reductions):

1. Deciding if a non-deterministic automaton $A$ accepts a word $w$.

2. Deciding if a directed graph has a cycle.

Exercise 4: Inclusions of complexity classes

Definition 1 A function $f : \mathbb{N} \rightarrow \mathbb{N}$ is said to be space-constructible if $\forall n \in \mathbb{N}, f(n) > \log(n)$ and there exists a deterministic Turing machine that computes $f(|x|)$ in space $O(f(|x|))$ given $x$ as input.

Show that for a space-constructible function $f$,
$$\text{NSPACE}(f(n)) \subseteq \text{DTIME}(2^{O(f(n))})$$
Exercise 5: Restrictions in the definition of \( \text{SPACE}(f(n)) \)

In the course, we restricted our attention to Turing machines that always halt, and whose computations are space-bounded on every input. In particular, remember that \( \text{SPACE}(f(n)) \) is defined as the class of languages \( L \) for which there exists some deterministic Turing machine \( M \) that always halts (i.e. on every input), whose computations are \( f(n) \) space-bounded (on every input), such that \( M \) decides \( L \).

Now, consider the following two classes of languages:

- \( \text{SPACE}'(f(n)) \) is the class of languages \( L \) such that there exists a deterministic Turing machine \( M \), running in space bounded by \( f(n) \), such that \( M \) accepts \( x \) iff \( x \in L \). Note that if \( x \notin L \), \( M \) may not terminate.

- \( \text{SPACE}''(f(n)) \) is the class of languages \( L \) such that there exists a deterministic Turing machine \( M \) such that \( M \) accepts \( x \) using space bounded by \( f(n) \) iff \( x \in L \) (\( M \) may use more space and not even halt when \( x \notin L \)).

1. Show that for a space-constructible function \( f = \Omega(\log n) \), \( \text{SPACE}'(f(n)) = \text{SPACE}(f(n)) \)
2. Show that for a space-constructible function \( f = \Omega(\log n) \), \( \text{SPACE}''(f(n)) = \text{SPACE}(f(n)) \)