Exercise 1:
Let $\Sigma$ be a fixed finite alphabet. Show that the language of deterministic finite automata (DFAs) on $\Sigma$ that accept every word is decidable. Formally, show that
$$L = \{\langle A \rangle \mid A \text{ is a deterministic finite automaton with } L(A) = \Sigma^*\}$$
is a decidable language.

Remark: You can use that it is not difficult to construct a Turing machine which tests whether an input is the well formed encoding of a deterministic finite automaton.

Exercise 2: The class $P$
Show that the following languages are in $P$.

(a) 5-Cycle $= \{\langle G \rangle \mid G \text{ is a graph and contains a cycle of length 5}\}$.

Remark: A cycle of length 5 in $G$ are five distinct nodes $v_0, \ldots, v_4$ such that the edges $\{v_i, v_{i+1 \mod 5}\}$, $i = 0, \ldots, 4$ exist in $G$.

(b) $L = \{a^n b^m \mid n \geq 0\}$

(c) 17-INDEPENDENT SET $= \{\langle G \rangle \mid G \text{ is a graph and contains an independent set of size 17}\}$.

Remark: An independent set of a graph with size $s$ is a set $S \subseteq V$, $|S| = s$ such that $\{v, w\} \notin E$ for all $u, w \in S$.

(d) Find a proper citation (e.g., via google) which states whether $\text{Primes} = \{\langle n \rangle \mid n \in \mathbb{N} \text{ is prime}\}$ is in $P$ or not.

Exercise 3: Decision vs. construction?
In this exercise we want to show the relation of the decision and the optimization variant of the knapsack problem.

The knapsack problem (KP) is defined as follows. Input: $b \in \mathbb{N}$, weights $w_1, \ldots, w_N \in \{1, \ldots, b\}$ and profits $p_1, \ldots, p_N \in \mathbb{N}$. A feasible solution is a $K \subseteq \{1, \ldots, N\}$ such that
$$\sum_{i \in K} w_i \leq b.$$
In the *optimization variant* the goal is to find a feasible solution which maximizes the profit
\[ \sum_{i \in K} p_i . \]

We can define a *decision variant* of the problem as follows:

\[ KP = \{ \langle b, w_1, \ldots, w_N, p_1, \ldots, p_N, k \rangle | b, w_1, \ldots, w_N, p_1, \ldots, p_N \text{ forms a valid knapsack instance} \]

and it has a feasible solution with profit at least \( k \} \).

It is well known that the decision variant of KP is an NP-complete problem.

In the following we will (implicitly) show that the optimization variant is also NP-complete.

(a) Show how to use a polynomial number of repetitions of a KP algorithm to determine the optimal value (profit) of a feasible solution for a given knapsack instance.

(b) Use the previous result to show that if the decision variant of KP is in \( \mathcal{P} \) then the optimization variant (=computing an optimal solution) of KP is in \( \mathcal{P} \) as well.

*Hint: Try to use the algorithm from a) to iteratively decide whether an item \( i \) is needed in an optimal solution.*