Test 1

Problem 1. Why do we need to study automata? Provide two main reasons.

Problem 2–4. Let us consider the automaton that has two states: $s$ (student is struggling) and $c$ (student is doing well); $s$ is the starting state, $c$ is the final state. The only two symbols are $h$ (help provided) and $n$ (no help provided).

- From $s$, $n$ lead back to $s$, and $h$ leads to $c$.
- From $c$, any symbol leads back to $c$.

Problem 2. Trace, step-by-step, how this finite automaton will check that the word $hnh$ belongs to this language. Use the tracing to find the parts $x$, $y$, and $z$ of the word $nhn$ corresponding to the Pumping Lemma. Check that the “pumped” word $xyyz$ will also be accepted by this automaton.

Problem 3. Write down the tuple $(Q, \Sigma, \delta, q_0, F)$ corresponding to this automaton:

- $Q$ is the set of all the states,
- $\Sigma$ is the alphabet, i.e., the set of all the symbols that this automaton can encounter;
- $\delta : Q \times \Sigma \rightarrow Q$ is the function that describes, for each state $q$ and for each symbol $s$, the state $\delta(q, s)$ to which the automaton that was originally in the state $q$ moves when it sees the symbol $s$ (you do not need to describe all possible transitions this way, just describe two of them);
- $q_0$ is the staring state, and
- $F$ is the set of all final states.

Problem 4. Use a general algorithm that we had in class to generate a context-free grammar corresponding to this automaton. Show how this grammar will generate the word $hnh$.

Problem 5. Let $A_1$ be the automaton described in Problem 2. Let $A_2$ be an automaton that accepts all the strings that contains even number (0, 2, 4, etc.) of helps $h$. This automaton has two states: the starting state $e$ which is also final, and the odd state $d$. The transitions are as follows:
• from the start state, \( n \) lead back to the start state, while \( h \) leads to the odd state \( d \);
• from the odd state, \( h \) leads to the start state, while \( n \) leads back to the odd state.

Use the algorithm that we had in class to describe the following two new automata:

• the automaton that recognizes the union \( A_1 \cup A_2 \) of the two corresponding languages, and
• the automaton that recognizes the intersection of the languages \( A_1 \) and \( A_2 \).

**Problem 6.** Use the general algorithm that we learned in class to design a non-deterministic finite automaton that recognizes the language \((h \cup n)(h \cup n)^*\):

• first, describe the automata for recognizing \( h \) and \( n \);
• then, combine them into the automata for recognizing the union \( h \cup n \), and the Kleene star \((h \cup n)^*\);
• finally, combine the automata for \( h \cup n \) and \((h \cup n)^*\) into an automaton for recognizing the desired composition of the two languages.

**Problem 7.** Use the general algorithm to transform the resulting non-deterministic finite automaton into a deterministic one.

**Problem 8–9.** Use a general algorithm to transform the finite automaton from Problem 2 into the corresponding regular expression.

**Problem 10.** Prove that the language \( L \) of all the words that have at least twice more \( h \)'s than \( n \)'s is not regular.