Solution to Homework 1

Task 1: general description. In class, we designed automata for recognizing integers and real numbers.

Task 1.1. Use the same ideas to describe an automaton for recognizing people’s names. A general name should start with a capital (= uppercase) letter, all other letters should be small (= lowercase).

A natural idea is to have 3 states: start (s), correct name (n), and error (e). Start is the starting state, n is the only final state. The transitions are as follows:

- from s, any capital letter $A, \ldots, Z$ lead to n, every other symbol leads to e;
- from n, any small letter leads back to n, every other symbol leads to e;
- from e, every symbol leads back to e.

Solution. The desired automaton takes the following form:
Task 1.2. Trace, step-by-step, how the finite automaton from Part 1.1 will check whether the following two words (sequences of symbols) are correct names for Java constants:

- the word Luc (which this automaton should accept) and
- the word LUC (which this automaton should reject).

Solution. Let us trace how this automaton will accept the word Luc We are originally in the state $s$:

Then, we read the first letter $L$ of the word Luc, so we move to state $n$:

Then, we read the second letter $u$ of the word Luc, and we stay in the state $n$: 
Then, we read the third symbol \( c \) of the word \( Lu c \), and we stay in the state \( n \):

The word is read, we are in the final state, so the word \( Lu c \) is accepted.

Let us now trace how the automaton will react to the word \( LUC \). We also start in the start state \( s \):

Then, we read the first letter \( L \) of the word \( LUC \), so we move to the state \( n \);
After that, we read the second symbol $U$ of the word $LUC$ and move to state $e$:

Then, we read the last symbol $C$ of the word $LUCC$ and stay in the state $e$:

We have read all the symbols, we are in the state $e$ which is not final, so the word $LUC$ is not accepted.
Task 1.3. Write down the tuple $(Q, \Sigma, \delta, q_0, F)$ corresponding to the automaton from Part 1.1:

- $Q$ is the set of all the states,
- $\Sigma$ is the alphabet, i.e., the set of all the symbols that this automaton can encounter; for simplicity, consider only four symbols: the plus sign, letters $a$ and $A$, and an underscore;
- $\delta : Q \times \Sigma \to Q$ 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.

Solution. $Q = \{s, n, e\}$, $\Sigma = \{a, A, 1\}$, $q_0 = s$, $F = \{n\}$, and the transition function $\delta$ is described by the following table:

<table>
<thead>
<tr>
<th></th>
<th>$a$</th>
<th>$A$</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>$e$</td>
<td>$n$</td>
<td>$e$</td>
</tr>
<tr>
<td>$n$</td>
<td>$n$</td>
<td>$e$</td>
<td>$e$</td>
</tr>
<tr>
<td>$e$</td>
<td>$e$</td>
<td>$e$</td>
<td>$e$</td>
</tr>
</tbody>
</table>
**Task 1.4.** Apply the general algorithm for union and intersection to:

- the automaton from Part 1.1 as Automaton $A$ and
- an automaton for recognizing Java names for classes as Automaton $B$.

In Java, a name for a class should start with a capital letter, all other symbols can be letters (small or capital), digits, or an underscore symbol. A natural idea is to also have 3 states: start ($s$), correct class name ($c$), and error ($e$). Start is the starting state, $c$ is the only final state. The transitions are as follows:

- from $s$, any capital letter $A, \ldots, Z$ lead to $c$, every other symbol leads to $e$;
- from $c$, any small letter $a, \ldots, z$, digit, or underscore leads back to $c$, every other symbol leads to $e$;
- from $e$, every symbol leads back to $e$.

For simplicity, in your automaton for recognizing the union and intersection of the two languages, feel free to assume that you only have symbols $a$, $A$, and 1.

**Solution.** If we limit ourselves to these 3 symbols, then the Automaton $A$ takes the following form:

The Automaton $B$ has the following form:
In the beginning, before we see any symbols, both automata are in the state $s$, so the combined automaton is in the state $(s, s)$. Then:

- if we read $A$, Automaton $A$ goes into state $n$ and automaton $B$ goes into state $c$, so we go into the state $(n, c)$;
- if we read 1 or $a$, both automata go into the $e$ states, so the combined automaton goes into the state $(e, e)$.

We can similarly describe transitions from these three new states. As a result, we get the following automaton: