Task: Apply the general algorithm for transforming the finite automaton into a regular language (i.e., a language described by a regular expression) to the following automaton.

This automaton has two states: \( s (= \text{straight-A student}) \) and \( e (= \text{everyone else}) \); \( s \) is the starting state, it is also the final state. The only two symbols are \( A \) and \( B \).

- From \( s \), \( A \) leads to \( s \), and \( B \) to \( e \).
- From \( e \), any symbol leads back to \( e \).

Solution. We start with the described automaton:

According to the general algorithm, first we add a new start state \( ns \) and a few final state \( f \), and we add jumps:

- from the old start state to the new start state, and
- from each old final state to the new final state.

As a result, we get the following automaton.

Then, we need to eliminate the two intermediate states \( s \) and \( e \) one by one. We can start with eliminating \( s \) or with eliminating \( e \). Let us show what happens in both cases.

First version, when we first eliminate the state \( s \). First, we draw all possible arrows:
Now, to find expressions to place at all these arrows, we will use the general formula
\[ R'_{i,j} = R_{i,j} \cup (R_{i,k}R^*_{k,k}R_{k,j}) , \]
where \( k \) is the state that we are eliminating, i.e., in this case, the state \( k = s \).

By applying this formula, and by using simplification formulas described in the lecture, we get the following results:

- \( R'_{ns,e} = R_{ns,e} \cup (R_{ns,s}R^*_{s,s}R_{s,e}) = \emptyset \cup (\Lambda A^* B) = \emptyset \cup A^* B = A^* B \);
- \( R'_{ns,nf} = R_{ns,nf} \cup (R_{ns,s}R^*_{s,s}R_{s,nf}) = \emptyset \cup (\Lambda A^* \Lambda) = \emptyset \cup A^* = A^* \);
- \( R'_{e,e} = R_{e,e} \cup (R_{e,s}R^*_{s,s}R_{s,e}) = (A \cup B) \cup (\emptyset \ldots) = (A \cup B) \cup \emptyset = A \cup B \);
- \( R'_{e,nf} = R_{e,nf} \cup (R_{e,s}R^*_{s,s}R_{s,nf}) = \emptyset \cup (\emptyset \ldots) = \emptyset \cup \emptyset = \emptyset \).

Thus, the 3-state \( a \)-automaton takes the following form:

Now, all that remains to do is to go from here to the 2-state \( a \)-automaton by eliminating the remaining state \( e \):
The final expression is the corresponding expression for $R_{ns,nf}'$:

$$R_{ns,nf}' = R_{ns,nf} \cup (R_{ns,e}R_{e,e}^*R_{e,nf}) = A^* \cup (A^*B(A \cup B)^*\emptyset) = A^* \cup \emptyset = A^*.$$  

The formula on the previous line is a regular expression corresponding to the original automaton.

**First version – answer: $A^*$.**

**Second version, when we first eliminate the state $e$.** First, we draw all possible arrows:

![Diagram](image)

Now, to find expressions to place at all these arrows, we will use the general formula

$$R_{i,j}' = R_{i,j} \cup (R_{i,k}R_{k,k}^*R_{k,j}),$$

where $k$ is the state that we are eliminating, i.e., in this case, the state $k = e$.

By applying this formula, and by using simplification formulas described in the lecture, we get the following results:

- $R_{ns,s}' = R_{ns,s} \cup (R_{ns,e}R_{e,e}^*R_{e,s}) = \Lambda \cup (\emptyset \ldots) = \Lambda \cup \emptyset = \Lambda$;
- $R_{ns,nf}' = R_{ns,nf} \cup (R_{ns,e}R_{e,e}^*R_{e,nf}) = \emptyset \cup (\emptyset \ldots) = \emptyset \cup \emptyset = \emptyset$;
- $R_{s,s}' = R_{s,s} \cup (R_{s,e}R_{e,e}^*R_{e,s}) = A \cup (B(A \cup B)^*\emptyset) = A \cup \emptyset = A$;
- $R_{s,nf}' = R_{s,nf} \cup (R_{s,e}R_{e,e}^*R_{e,nf}) = \Lambda \cup (B(A \cup B)^*\emptyset) = \Lambda \cup \emptyset = \Lambda$.

Thus, the 3-state $a$-automaton takes the following form:

![Diagram](image)
Now, all that remains to do is to go from here to the 2-state a-automaton by eliminating the remaining state $x$:

![Diagram](image)

The final expression is the corresponding expression for $R_{ns,f}'$:

$$R_{ns,nf}' = R_{ns,nf} \cup (R_{ns,s}R_{s,s}R_{s,nf}) = \emptyset \cup (\Lambda A^* \Lambda) = A^*.$$  

The formula $A^*$ is also a regular expression corresponding to the original automaton.

**Second version – answer:** $A^*$. 