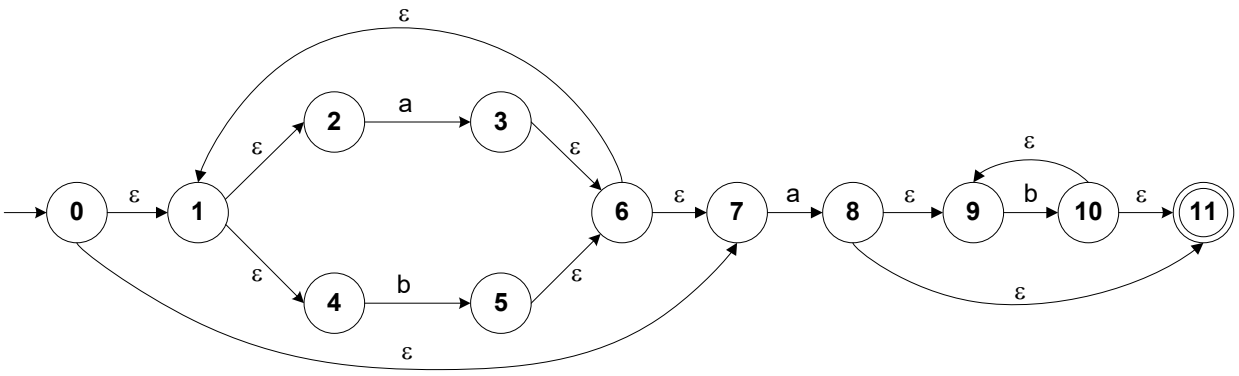
Exercise 1.

Consider the following regular expression : (a|b)*ab*

a) Construct a nondeterministic finite automaton (NFA) for the above regular expression using Thompson's construction.
b) Convert this nondeterministic finite automaton into a deterministic finite automaton (DFA).
c) Minimize the number of states in the resulting automaton.

Solution :

a) Construction of an NFA using Thompson's construction for the regular expression **(a|b)*ab***:

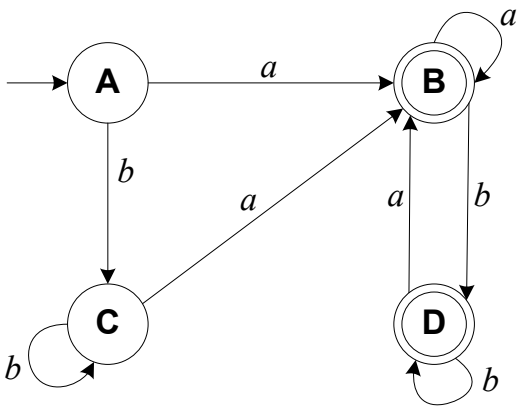


b) Transformation NFA → DFA:

- Initial state = ε-closure(0) = {0,1,2,4,7}

State	a	b
{0,1,2,4,7} = **A**	**B**	**C**
{1,2,3,4,6,7,8,9,11}= **B**	**B**	**D**
{1,2,4,5,6,7}= **C**	**B**	**C**
{1,2,4,5,6,7,9,10,11}= **D**	**B**	**D**

- The accepting states (The final states): { B, D }

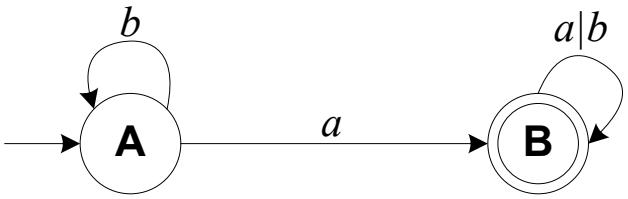


c) Minimization:

1) Π = {A,C} , {B,D}

2) Π = {A,C} , {B,D}

State	a	b
A	B	A
B	B	B



Minimal DFA for the regular expression (a|b)*ab*

Exercise 2.

- The lexical entities of a mini programming language are as follows:

Keywords	begin , end , if , then , else
Identifiers	Strings consisting of a single letter, or a letter followed by several letters or digits
Constants	Strings consisting of a single digit, or several digits
Operators	< , <= , = , < > , > , >= , + , -

a) Provide the regular expressions that describe these lexical entities.

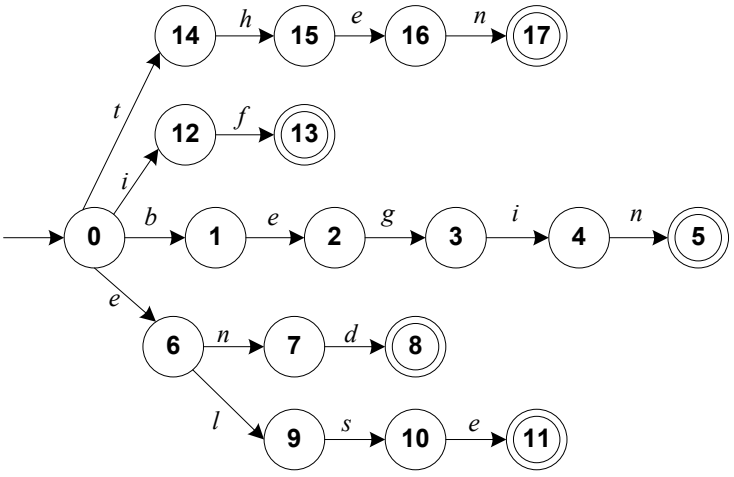
b) Construct a finite-state automaton for these expressions.

Solution :

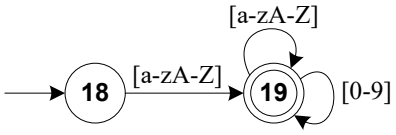
a) Regular expressions describing the lexical entities :

- Keywords : **begin | end | if | then | else**
- Identifiers : **[A-Za-z]([A-Za-z0-9])***
- Constants: **[0-9]+**
- Operators: **< | <= | = | < > | > | >= | + | -**

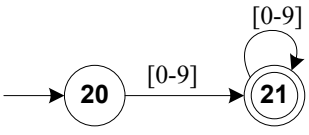
b) Construct a finite-state automaton for these regular expressions.



FA for the keywords

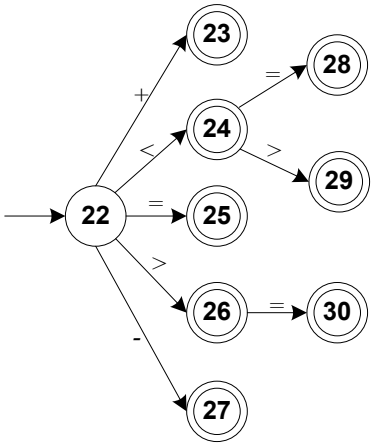


FA for the identifiers



FA for the constants

FA for the operators



Exercise 3.

The Thompson construction rules transform a regular expression $\mathbf{R_1}$ into a nondeterministic finite automaton $\mathbf{N_1}$. Propose analogous construction rules for nondeterministic finite automata for the following operators:

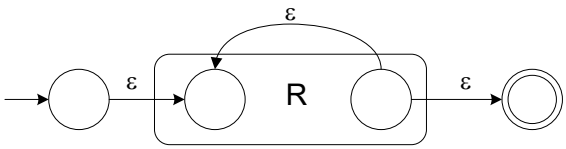
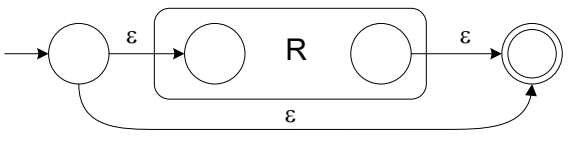
- $\mathbf{R_1^+}$
- $\mathbf{R_1?}$ (whose meaning is $\mathbf{R_1 \mid \varepsilon}$)

We modify Thompson's construction rule for the expression **R*** by **not** adding a new initial state and a new final state. Instead, we add two ε-labeled transitions: one from the final state of R's automaton back to its initial state, and the other from the initial state to the final state.

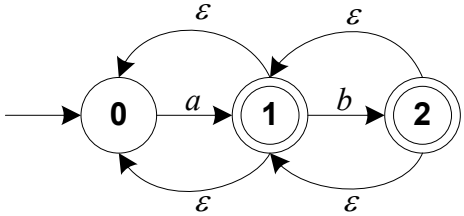
Is this modified rule always valid for the expression **R***? In the general case (composition of rules), does the proposed modification affect the validity of the constructions? Give a concise example to justify your answer.

Solution :

- Thompson's rules for $R1^+$ and R1? :

▪ $R1^+$:	▪ R1? :
	

- Counterexample: The regular expression a*b*: with the new rule we obtain the following NFA :



Contradiction: for example, we can see that the string **ababab** can be recognized by this **NFA**, whereas it does not belong to the language generated by the original regular expression **a*b*.**