Exercise 1

Just to recall:

Name and define all grammars and automata in the Chomsky hierarchy:

- FSA
- RG
- PDA
- CFG
- LBA
- CSG
- TM
- G0

Explain the reasons for vertical (automata – grammars) and horizontal (FSA – TM) split.

Define formally the finite state automaton.

Practice:

Design a finite state automaton that counts a tennis score.

Design finite state automata working with the alphabet \{a,b\}, that accept the following words:

- the number of letters „a“ is divisible by 3,
- the number of letters „a“ is divisible by 2,
- the number of letters „a“ is divisible by 3 or by 2,
- the number of letters „a“ is divisible by 3 and by 2,
- the number of letters „a“ is divisible by 3 but not by 2,
- finishes by „baba“,
- contains „baba“,
- starts with „baba“,
- finishes by „b“ and its length is \((3k+1)\),
- finishes by „b“ or its length is \((3k+1)\),
- finishes by two or more letters „b“ that are immediately preceded by at least one letter „a“,
- its length is at least 2 and the first and the last letter in the word are identical,
- the first and the last letter in the word are identical,
- its length is at least 4 and the first two letters are identical to the last two letters,
- the first two letters are identical to the last two letters.

What are the following automata doing (describe the language)?

\[
\begin{array}{c|c|c}
 & 0 & 1 \\
\hline
a) & q_0 & q_1 \\
    & q_1 & q_2 \\
    & q_2 & q_0 \\
\hline
b) & q_0 & q_1 \\
    & q_1 & q_2 \\
    & q_2 & q_1 \\
\hline
c) & q_0 & q_1 \\
    & q_1 & q_2 \\
    & q_2 & q_0 \\
\hline
d) & q_0 & q_1 \\
    & q_1 & q_2 \\
    & q_2 & q_0 \\
\end{array}
\]
Exercise 2

Just to recall:

*Formulate the Myhill-Nerode Theorem and the Pumping Lemma.*

*Define the state equivalence.*

Practice:

*Decide and prove if the following languages are regular*

- $0^k1^l$, $k \leq 1$
- $0^k1^k$
- $1^k0^k1^l$
- $1^k0^k1^k$
- $1^k1^k0^l$
- $1^k1^k0^k$
- $0^m1^n$ for $n \leq m \leq 2n$
- $0^m1^n0^n$ for $m \leq n$
- $ww$, $w \in \{0,1\}^*$
- $w$, $w \in \{0,1\}^*$ and $w$ contains identical counts of letters 0 and 1
- $ww^R$, $w \in \{a\}^*$
- $a^{2n}$
- $a^*$
- $a^+$
- $a^p$, $p$ is a prime number
- $a^*bc^i$
- $a^*bc^i$ pro $i < j$

*Find all equivalent states in the following automata:*

**a)**

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>7</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
</tr>
</tbody>
</table>

**b)**

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>F</td>
</tr>
<tr>
<td>B</td>
<td>A</td>
</tr>
<tr>
<td>C</td>
<td>D</td>
</tr>
<tr>
<td>D</td>
<td>B</td>
</tr>
<tr>
<td>E</td>
<td>C</td>
</tr>
<tr>
<td>F</td>
<td>E</td>
</tr>
</tbody>
</table>

**c)**

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
</tr>
</tbody>
</table>

**d)**

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>G</td>
</tr>
<tr>
<td>B</td>
<td>A</td>
</tr>
<tr>
<td>C</td>
<td>D</td>
</tr>
<tr>
<td>D</td>
<td>B</td>
</tr>
<tr>
<td>E</td>
<td>C</td>
</tr>
<tr>
<td>F</td>
<td>E</td>
</tr>
</tbody>
</table>

**e)**

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
</tr>
</tbody>
</table>

**f)**

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
</tr>
</tbody>
</table>

**g)**

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
</tr>
</tbody>
</table>

```
Exercise 3

Just to recall and to think about:
1) *The Pumping lemma as formulated in the lecture assumes that the part to iterate is located at the beginning of the world (|uv| ≤ n). Think if it is possible to formulate the Pumping lemma, where:*
   - the part to iterate is located anywhere in the world
   - the part to iterate is located at the end of the world
   - the part to iterate is located close to some given place in the world.
2) *Does the Pumping lemma hold also for complements of regular languages? In particular “we can divide the word outside the regular language in such a way that we can iterate the “middle” part and the resulting word is still outside the language”*
3) *Define formally the equivalence of automata. Is there any relation to state equivalence?*
4) *What is a reduced finite state automaton? Define it.*

Practice:
*Decide and prove if the following automata are equivalent:*

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>a)</td>
<td>←0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>←4</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>←6</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>b</th>
<th>a</th>
</tr>
</thead>
<tbody>
<tr>
<td>b)</td>
<td>A</td>
<td>F</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>A</td>
</tr>
<tr>
<td></td>
<td>C</td>
<td>D</td>
</tr>
<tr>
<td></td>
<td>D</td>
<td>B</td>
</tr>
<tr>
<td></td>
<td>E</td>
<td>C</td>
</tr>
<tr>
<td></td>
<td>←F</td>
<td>E</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>←6</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>b</th>
<th>a</th>
</tr>
</thead>
<tbody>
<tr>
<td>c)</td>
<td>A</td>
<td>G</td>
</tr>
<tr>
<td></td>
<td>B</td>
<td>A</td>
</tr>
<tr>
<td></td>
<td>C</td>
<td>D</td>
</tr>
<tr>
<td></td>
<td>D</td>
<td>B</td>
</tr>
<tr>
<td></td>
<td>E</td>
<td>D</td>
</tr>
<tr>
<td></td>
<td>←F</td>
<td>E</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>←G</td>
<td>F</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>0</td>
</tr>
</tbody>
</table>

What is the shortest word in automaton (a) that differentiates states 1 and 5? Are there more such words?

In the following automata find all the shortest words that differentiate a given pairs of states:

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>e)</td>
<td>←0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>f)</td>
<td>←0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>←4</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>g)</td>
<td>←0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>←0</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>3</td>
</tr>
</tbody>
</table>
Exercise 4

Just to recall and to think about:
1) Which set and string operations have the closure property for regular languages?
2) Do we obtain a complement of language if we swap the role of accepting and non-accepting states in a nondeterministic finite-state automaton?

Practice:
Convert the following automaton to a reduced finite state automaton:

Propose algorithms that decide if the following propositions hold:
- $L(A) = \emptyset$
- $L(A) = L(B)$
- $L(A) = X^*$
- $L(A) \subseteq L(B)$
- $L(A)$ je nekonečný

Let $L = \{ab, c\}$. Describe the following languages: $L^+, L^*, (L^*)^*$

Propose a FSA accepting all words in the alphabet $\{a, b\}$ that do not contain the word “baba”.

Let $X = \{0, 1\}$ be an alphabet. Design a FSA accepting words where the number of letters 0 is divisible by:
- 2
- 3
- 2 or 3
- 2 and 3
- 2 but not by 3.

Let $L_1 = \{u | u \in \{0, 1\}^* \text{ and } |u|_0 = 2k\}$ and $L_2 = \{u | u \in \{0, 1\}^* \text{ and } |u|_0 = 3k\}$ be languages. Propose the smallest FSA accepting the language $L_2 \setminus L_1$.

What is the minimal number of states in a FSA that accepts the following language?

$L_n = \{ w | w \in (0, 1)^*, w = u1v, |v| = n - 1 \}$.

What is the minimal number of states in a FSA that accepts the language $(L_n)^R$?
Let the following automaton accepts some language \( L \): 

Propose (nondeterministic) FSAs accepting the following languages:

- \( L_1 = \{uv \mid uav \in L \lor ubv \in L\} \)
- \( L_2 = \{uv \mid uav \in L\} \)
- \( L_3 = \{uav \mid uv \in L\} \)
- \( L \cup L_1, L \cup L_2, L \cup L_2 \)
Exercise 5

Just to recall and to think about:

1) What are the core algebraic operations to obtain all regular languages?
2) How is the language \( \{ \lambda \} \) composed from the elementary languages?
3) Is there any relation between the proof of the Kleene’s theorem and all-pairs-shortest-path algorithms?

Practice:

1) Write a regular expression whose value is a language in the alphabet \( \{a,b\} \) consisting exactly from the words that start with “ba” and finish with “ab”. Convert the regular expression to a corresponding FSA.
2) Write a regular expression whose value is a language containing words \( a, a^*a, a*a*a, \ldots \).
3) Convert the following regular expressions to FSAs accepting the languages that are values of these expressions:
   - \( ab+ba \)
   - \( a^2+b^2+ab \)
   - \( a+b^* \)
   - \( (ab+c)^* \)
   - \( ((ab+c)^*a(bc)^*+b)^* \)
   - \( ((ab+c)^*a(bc)^*+b)^* \)
   - \( (01^*+101)^*0^*1 \)
   - \( (01)^*1111(01)^*+(0+1)^*000 \)

4) Convert the following automata to regular expressions such that the value of the expression is the language accepted by a given FSA:
Exercise 6

Just to recall and to think about:

1) Regular expression is a word. Is the language consisting of all regular expressions acceptable by some finite state automaton?
2) How can a finite state automaton inform about its computation? What is the difference between Moore and Mealy machines?
3) What is the advantage of non-determinism and the possibility to move the reading head in both directions?

Practice:

1) Let $L$ be a regular language. Is the language \{\text{u| #u\#} \in L\} also regular? Prove it!
   Note: The symbol “#” is a part of the alphabet.
2) Let $L$ be a language accepted by a finite state automaton $A$. Construct a two-way (non-deterministic) finite state automaton accepting the following language:
   • \{#u# | uu^R \in L\}
   • \{#u# | uu \in L\}
   • \{#u# | uv \in L \& |u| = |v|\}
   • \{#u# | u = wv \& w^Rwv \in L\}
   Convert the obtained automata to a finite state automata.
3) Design a Mealy machine working with the alphabet \{0,1\} that inverts the input word (0\rightarrow1, 1\rightarrow0). Convert the machine to an equivalent Moore machine.
4) Design a Mealy Machine working with the alphabet \{0,1\} that implements the following output function:

   output 1, if the input symbol is a part of sequence of 1s, which is directly preceded by symbols 00,
   output 0 in all other cases.

   Convert the Mealy machine to an equivalent Moore machine.
5) Design a formal machine that sums two binary numbers. Think about defining the input for such a machine.
Exercise 7

Just to recall and to think about:
1) Explain the notion of a formal grammar.
2) What is the difference between generative and analytical formal grammar?
3) How are the types of grammars in the Chomsky hierarchy distinguished?
4) What is the reason for the names of context-sensitive and context-free grammars?
5) If we combine left-linear and right-linear rewriting rules do we obtain a grammar accepting a regular language?

Practice:

1) Design a formal grammar that generates the following language:

- \{a^ib^j \mid i \geq 0\}
- well-formed expressions with left and right brackets
- well-formed arithmetic expressions with a single constant “c”, operations + and *, and left and right brackets
- \{w \mid w \in \{a, b\}^*, |w|_b = 3k\}
- binary numbers that are multiples of 3
- \{a^ibjc^i \mid i \geq 0\}
- \{a^ib^jc^{ij} \mid i, j \geq 0\}
- \{a^ib^jc^i \mid 0 \leq i \leq j, k\}
- \{a^ib^jc^i \mid i = j \lor j = k\}
- \{ww \mid w \in \{a, b\}^*\}
- \{ww \mid w \in \{a, b\}^*\}
- \{0^n1^n0^n \mid 0 \leq m \leq n\}
- \{0^n1^n \mid 0 \leq n \leq m \leq 2n\}
- \{0^n1^n \mid 0 \leq n \leq m \leq 2n\}

2) Is the following grammar context-sensitive (the capital letters denote the non-terminal symbols)? Can the grammar be converted to a context-sensitive form?

\[
\begin{align*}
S & \to aSbA \mid \lambda \\
A & \to aBbA \mid bCB \mid CD \\
B & \to bbBa \mid aS \\
C & \to aAaA \mid \lambda \\
D & \to SC \mid aABb
\end{align*}
\]

3) Convert the following FSA to a grammar that generates the language accepted by the FSA. What type of grammar do we obtain?
4) Convert the following grammar to a finite state automaton accepting the same language. Can any grammar be converted to an “equivalent” finite state automaton?

\[\begin{align*}
S & \rightarrow abS \mid bbaA \mid \lambda \\
A & \rightarrow abA \mid bB \mid C \\
B & \rightarrow acS \mid bC \mid \lambda \\
C & \rightarrow abb \mid bA \mid A
\end{align*}\]
Exercise 8

To recall and to think about:

1) What is the main difference between the reduced context-free grammar and reduced FSA?

2) Assume a word generated by a given context-free grammar. Is the derivation for this word unique?

3) Does the order of application of production rules influence the final generated word for a CFG? Explain.

4) Is there another way to describe how the word is derived (different from the derivation)?

Practice:

1) The figure shows a syntax (derivation) tree for some context-free grammar G.

* What is the generated word given by this tree?
* Write a left derivation for this word.
* Write all the rewriting rules used in this syntax tree.
* Can we say something about ambiguity of the grammar G?

2) Reduce the following CFGs:

\[
\begin{align*}
S &\rightarrow aA | bB | aSa | bSb | \lambda \\
A &\rightarrow aAB | bB \\
B &\rightarrow aAb | BB \\
C &\rightarrow CC | cS \\
D &\rightarrow DE \\
E &\rightarrow \lambda
\end{align*}
\]

3) Decide (and prove) if the following grammar G satisfies \(L(G) = \emptyset\).

\[
\begin{align*}
S &\rightarrow aS | AB | CD \\
A &\rightarrow aDb | AD | BC \\
B &\rightarrow bSb | BB \\
C &\rightarrow BA | ASb \\
D &\rightarrow ABCD | \lambda
\end{align*}
\]
Exercise 9

To recall and to think about:

1) What is main difference between a pushdown automaton and a finite state automaton?
2) Describe the mechanism how the pushdown automata accept words.
3) Is there any difference if the pushdown automaton accepts the works using an acceptance state or using an empty stack?
4) Do deterministic pushdown automata accept the same class of languages as nondeterministic pushdown automata?

Practice:

1) Design pushdown automata accepting the following languages. For each automaton explore both types of accepting the words. If possible, try to design the automaton as a deterministic automaton. For the automata that use an empty stack to accept the words, try to design an automaton with a single state.

- \{0^n1^m \mid 0 \leq n \leq m \}
- \{wcw^R \mid w \in \{a,b\}^*\}
- \{ww^R \mid w \in \{a,b\}^*\}
- \{w \mid w \in \{a,b,c\}^* w \downarrow_{a,b} = uu^R\}, where w \downarrow_{a,b} is a word w, where all symbols different from “a” and “b” were removed
- \{w \mid w \in \{a,b\}^* |w_b|=|w_a|\}
- \{ucv \mid u,v \in \{a,b\}^* |u| \neq |v| \}
- \{L_i = \{ucv \mid u,v \in \{a,b\}^* u and v are different in the i-th symbol from the left \}
- \{ucv \mid u,v \in \{a,b\}^* u \neq v \}
- well-formed bracketed expression (such as “(())”)
- \{a^ib^j \mid i \geq 0\}
- \{a^ib^ic^{i+j} \mid i,j \geq 0\}
- \{a^ib^j \mid i=j \vee j=k\}

2) Convert the following grammar G to a pushdown automata accepting language L(G) using the acceptance states. Show, how the automaton accepts the word (a+a)*a.

\[
E \rightarrow E + T \mid T \\
T \rightarrow T * F \mid F \\
F \rightarrow (E) \mid a
\]

3) Take the context-free grammars from Exercise 7 and convert them to pushdown automata.
Exercise 10

To recall and to think about:

1) What is typical for derivations done with the grammar in the Greibach normal form is used?
2) What is typical for syntax trees for grammars in the Chomsky normal form is used?
3) What is the relation between the depth of the syntax tree for grammars in the Chomsky normal form and the length of the generated word?
4) Formulate and prove the pumping lemma for context-free languages.
5) Formulate and prove the pumping lemma for linear languages (use the ideas from CFL).

Practice:

1) Convert the following grammars to the Chomsky normal form:

\[
\begin{align*}
S & \rightarrow A \mid 0SA \mid \lambda \\
A & \rightarrow 1A \mid 1 \mid B1 \\
B & \rightarrow 0B \mid 0 \mid \lambda
\end{align*}
\]
\[
S \rightarrow 0A10B11 \\
A \rightarrow 0A1 \mid \lambda \\
B \rightarrow 0B11 \mid \lambda
\]

2) Convert the following grammar to the Greibach normal form:

\[
\begin{align*}
S \rightarrow (E) \\
E & \rightarrow F+F \mid F*F \\
F & \rightarrow a \mid S
\end{align*}
\]

3) Decide and prove if the these languages are context-free:

- \{a^i b^j c^k \mid i \geq 0\}
- \{a^i b^j c^k \mid 0 \leq i \leq j \leq k\}
- \{a^i b^j c^k \mid i, j \geq 0\}
- \{ww^R \mid w \in \{a, b\}^*\}
- \{ww \mid w \in \{a, b\}^*\}
- \{0^n 1^n 0^n \mid 0 \leq m \leq n\}
- \{0^n 1^n \mid 0 \leq n \leq m \leq 2n\}
- \{0^n 1^n 0^n \mid 0 \leq n\}
- \{0^n 1^n 1^n \mid 0 \leq n\}
- \{a^i a^j \mid 0 \leq i \leq j\}
- \{a^*\}
- \{a^2\}