1 WORDS AND LANGUAGES

1.1 Words and Alphabets

An **alphabet** is a finite, nonempty set of symbols. By convention it used the symbol $\sum$ for an alphabet.

Example 1:
$\sum = \{0,1\}$ the binary alphabet;
$\sum = \{a,b, \ldots z\}$ the set of all lowercase letters.

A string (or sometimes a **word**) is a finite sequence of symbols chosen from an alphabet.

Example 2: 010101010 is a string chosen from the binary alphabet, as is the string 0000 or 1111.

The **empty string ε *(or "epsilon")*** is the string with zero occurrences of symbols. This string is denoted **ε** and may be chosen from any alphabet.

The **length** of a string indicates how many symbols are in that string.

If we have $|w|=0$, then the word is empty and is denoted by ε (empty word).

If $a \in \Sigma$ and word $w \in \Sigma$, then by $|w|_a$ is denoted the number of appearance of the symbol a in w.

Example: the string w=0111 using the binary alphabet has a length of |w|=4.

If $\sum$ is an alphabet, it can express the set of all strings of a certain length from that alphabet by using an exponential notation.

$\sum^k$ is defined to be the set of strings of length k, each of whose symbols is in $\sum$.

Example: for the given alphabet $\sum = \{0,1,2\}$ we have:

$\sum^0 = \{\varepsilon\}$
$\sum^1 = \{0,1,2\}$
$\sum^2 = \{00,01,02,10,11,12,20,21,22\}$
$\sum^3 = \{000,001,002,\ldots 222\}$

The set of words over an alphabet Σ is denoted by Σ*

$$\sum^{*}=\{a_1,....a_n \,|\, a_1,a_2...,a_n \in \sum, n\geq 0\}$$

$$\sum^{*}=\Sigma^{0}\cup\Sigma^{1}\cup\Sigma^{2}\cup...$$

The set of nonempty words over Σ is denoted by

$$\sum^{+}=\sum{}^{*}/\{\varepsilon\}$$

To **concatenate** strings, it is simply need to put them right next to one another.

Example: If x and y are strings, where x=111 and y=000 then concatinaton of two strings is xy = 111000.

The concatenation of the word w_1 and w_2 is denoted by w_1w_2 .

The concatenation is **associative**:

$$w_1(w_2w_3)=(w_1w_2)w_3.$$

The concatenation is **not commutative**:

$$w_1w_2\neq w_2w_1$$

The power of the word w is denoted by w^n.

$$w^n = ww...w$$

$$w^n = w^{n-1}w, n\geq 1$$

Example: if it is given the word $w=a$, then $w^2=aa$, $w^3=aaa$, etc.

If $w=a_1a_2....a_n$, then $w^{-1}=a_na_{n-1}....a_1$ is mirror or image of the word w**.**

Word v is a **prefix** of the word u, if it exists a word z, such that:

$$z=vu.$$

Word v is called s**ubword** of the word z, if there are words p and q, such that:

$$z=pvq.$$

Word v is called **suffix** of word z, if there is a word x, such that:

$$z=xv.$$

Practical Tasks

1. Determine the cardinality of the following languages over the alphabet $\Sigma = \{0, 1\}$:
   - Σ^0;
   - Σ^2;
   - Σ^3.
2. Describe what the Kleene star operation * over the following alphabets produces:
   - $\Sigma=\{0,1\}$;
   - $\Sigma=\{b\}$;
   - $\Sigma=\{\varnothing\}$.
3. Describe what the what the + operation over the following alphabets produces:
   - $\Sigma=\{0,1\}$;
   - $\Sigma=\{b\}$;
   - $\Sigma=\{\varnothing\}$.
4. For the given words $w_1=abc$ and $w_2=xyz$, calculate:
   - $w=w_1w_2$;
   - $|w|$;
   - $(w)^{-1}$;
   - for the w, present prefix, suffix and subword.
5. For the $\Sigma=\{a, b, c, d\}$ determine:
   - in Σ^4 how many elements start with aa;
   - calculate the number of elements for $|\Sigma^0+\Sigma^3|$.

1.2 Languages

A **language** is:

1) a set of strings from some alphabet (finite or infinite);

2) any subset L of Σ^*.

Some special languages:

- $L=\{\varnothing\}$ The empty set/language, containing no strings.
- $L=\{\varepsilon\}$ A language containing one string, the empty string.

A notation we will commonly use to define languages is by a "set-former":

$L=\{\, w \mid \text{something about } w \,\}$.

Example:

- $L=\{w \mid w$ consists of an equal number of 0's and 1's $\}=\{01, 0011, 000111,...\}$;
- $L=\{\, 0^n1^n \mid n\geq 1 \,\}=\{01,0011,000111...\}$;
- $L=\{\, 0^n1^n \mid n>1 \,\}=\{\, \varepsilon , 01,0011,000111...\}$.

Operations over the Languages

If L_1 and L_2 are languages over Σ

1. Union $L_1 \cup L_2 = \{U \in \sum^* \left| U \in L_1 \;\; sau \;\; U \in L_2 \right.\}$.
2. Intersection $L_1 \bigcap L_2 = \{U \in \sum^* \left| U \in L_1 si U \in L_2 \right.\}$.
3. Difference $L_1 / L_2 = \{U \in \sum^* \left| U \in L_1 \; si \; U \notin L_2 \right.\}$.
4. Concatenation $L_1L_2 = \{UV \left| U \in L_1, V_1 \in L_2 \right.\}$.
5. The power of the language
$$L^0 = \{\varepsilon\}\, ; n \geq 1 ; L^n = L^{n-1}L \, .$$
6. Mirror $L^{-1} = \{U^{-1} \left| U \in L \right.\}$
7. The Kleene star of the language L
$$L^* = \bigcup_{i=0}^{\infty} L^i = L^0 \cup L \cup L^2 \cup ... \cup L^i \cup ...$$
$$L^+ = \bigcup_{i=0}^{\infty} L^i = L \cup L^2 \cup ... \cup L^i \cup ...$$

Practical Tasks

1. For the given alphabet $\Sigma = \{0, 1\}$ and language $L=\{00,011\}$ determine: L^2, L^3, L^{-1}.

2. For the given alphabet $\Sigma = \{0, 1\}$ and languages $L_1=\{\varepsilon, 1\}$ and $L_2=\{0,11\}$ determine: L_1L_2, L_2L_1, $L_1^2L_2$, L_2^*, L_2^+.

3. Find a possible alphabet Σ for the following languages:
   - $L = \{th, ch, gh, sh\}$;
   - $L = \{apple, orange, 500\}$;
   - the language of all binary strings.

4. Present the language over some alphabet Σ, where each word from language contains substring *ab*.

5. Present finite language over some alphabet Σ, where each word has length 3.

6. Present the language over some alphabet Σ, where each word from language has a symbol *b* in the 4th position from the right.

7. Present the language over some alphabet Σ, where each word from language has no consecutive *a*.

8. Present the infinite language over some alphabet Σ, where each word from language contains even number of symbol *a*.

9. Present the language over the alphabet $\Sigma =\{0, 1\}$ in which every occurrence of 1 is not before an occurrence of 0.

10. Present the infinite language over some alphabet Σ, where each word from language contains equal number of *a* and *b*.

11. Present finite and infinite language over alphabet $V=\{x,y,z\}$.

1.3 Grammars

A **grammar** G is an ordered quadruple $G=(V_N, V_T, P, S)$ where:

- V_N - is a finite set of non-terminal symbols;
- V_T - is a finite set of terminal symbols;
  $V_N \cap V_T = \varnothing$
- $S \in N$ is a start symbol;
- P – is a finite set of productions of rules.
  $P \subseteq (V_N \cup V_T)^* \; V_N (V_N \cup V_T)^* \times (V_N \cup V_T)^*.$

Let $G=(V_N, V_T, P, S)$ is a grammar. The language generated by the grammar G is denoted by $L(G)$ and represents the set of all strings of terminals that are derivable from the starting state S.

Example:
Let $G = (V_N, V_T, S, P)$,
where $V_N = \{A, B, S\}$
$V_T = \{a, b\}$,
S is a start symbol
$P = \{S \to bABb, A \to aA, A \to a, B \to Bb, B \to b\}$.
The word generated by this grammar are:

- $\mathrm{S} \to bABb \to baBb \to babb$
  This derivation can be presented in the following way:
  $\mathrm{S} \xrightarrow{3} babb$ or $\mathrm{S} \xrightarrow{*} babb$
- $\mathrm{S} \to bABb \to baABb \to baaBb \to baabb$
  or $\mathrm{S} \xrightarrow{4} baabb$ or $\mathrm{S} \xrightarrow{*} baabb$
- $\mathrm{S} \to bABb \to baABb \to baaABb \to baaaABb \to baaaaBb \to baaabb$
  or $\mathrm{S} \xrightarrow{6} baaabb$ or $\mathrm{S} \xrightarrow{*} baaabb$
- $\mathrm{S} \to bABb \to baABb \to baaBbb \to baaBbbb \to baabbbb$
  or $\mathrm{S} \xrightarrow{5} baabbbb$ or $\mathrm{S} \xrightarrow{*} baabbbb$

The language $L(G)$ generated by this grammars is:

$L(G) = \{ ba^n b^m b | n \geq 1, m \geq 1\} = \{babb, baabb, babb, baabb....\}$

Practical Tasks

1. Present the language $L(G)$ generated by the given grammar $G = (V_N, V_T, S, P)$:
   - $V_N = \{ S\}; V_T = \{a\}, \ P = \{S \to aS, S \to \varepsilon\}.$
   - $V_N = \{ S\}; V_T = \{a\}, \ P = \{S \to aS, S \to a \}.$
   - $V_N = \{ A\}; V_T = \{0, 1\}, \ P = \{A \to 0A1, A \to 01 \}.$
2. Identify the grammar for the following languages:
   - $L=\{ a^n b^n | n \geq 0\}$
   - $L=\{ a^n b^n | n > 0\}$
   - $L=\{ a^n b^{n+1} | n \geq 1\}$
   - $L=\{ a^n b^n c^m d^m \mid n > 1, m > 1\}$
   - $L=\{ a^n b^n c^m d^m \mid n \geq 0, m \geq 0\}$
   - $L=\{ a^n b^m c^m d^n \mid n > 1, m > 1\}$
   - $L=\{ a^m b^n c^{m+n} \mid n \geq 1, m \geq 1\}$
   - $L=\{ a^m b^{m+n} c^n \mid n \geq 1, m \geq 1\}$
3. For the given grammar identify the generated word: $G = (V_N, V_T, S, P)$:

V_N = { <programm>,<set of afirmation>, <afirmation>, <assigment>,<test>,<variable>,<number>, <alpha>};

V_T = {begin, end, succ, pred, while, do, :=, ≠, ;, (,), 0,1….9, A, B,….Z},

P = {<program>→ begin end

<program>→ begin <set of afirmation>end

<set of afirmation>→<afrmation>

<set of afirmation>→<set of afirmation><afirmation>

<afirmation>→<assigment>

<afirmation>→ while<test>do<afirmation>| <program>

<afirmation>→ <program>

<test>→<variable>≠<variable>

<assigment>→<variable>:=0
<assigment>→<variable>:=succ(<variable>)
<assigment>→<variable>:=pred(<variable>)
<variable>→<alpha>
<variable>→<variable><alpha>
<variable>→<variable><number>
<number>→0|1|2|3|4|5|6|7|8|9
<alpha>→A|B|…|Z
}.

4. Define grammar tha generates the variable identifiers from Java.
5. Define the grammar that generates all real literals in Java.
6. Define the grammar that generates the strings that correspond to valid currency amounts. A valid string is either a dollar sign followed by a number which has no leading 0's, and may have a decimal point in which case it must be followed by exactly two decimal digits, OR a one or two-digit amount followed by the cent sign c. The single exception to this rule is that strings which begin with "$0." and are followed by exactly two digits are also acceptable. Thus, $432.63, $1, $0.29, 47c, 2c are all accepted, but $021, $4.3, $8.63c, $0.0 are not accepted.

1.4 Derivation Trees

A derivation in the language generated by a grammar can be represented graphically using an ordered rooted tree, called a derivation (or parse) tree:

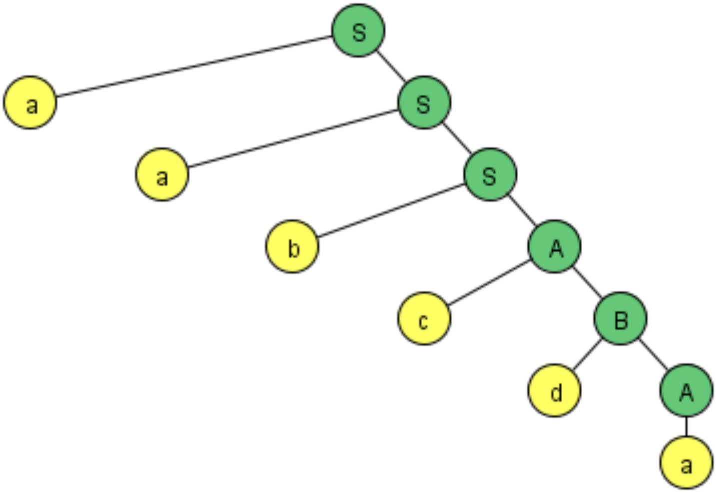
- the root represents the starting symbol;
- internal vertices represent nonterminals;
- leaves represent terminals.

Example:
$G = (V_N, V_T, S, P)$:

$V_N = \{ S, A, B\}$,
$V_T = \{a,b,c,d\}$,
$P = \{S \rightarrow aS, S\rightarrow a, S\rightarrow bA, A\rightarrow cB, A\rightarrow a,$
$B\rightarrow dA, B\rightarrow a\}$.

- Analisis if the word $w=aabcda$ can be generated by respectively grammar.
  $S\rightarrow aS\rightarrow aaS\rightarrow aabA\rightarrow aabcB\rightarrow aabcB\rightarrow aabcdA\rightarrow aabcda$

- Derivation tree:



Practical Tasks

Present the generated words but the given grammars and for the obtained word present the derivation trees:

- $G = (V_N, V_T, S, P)$: $V_N = \{ S\}$; $V_T = \{a, b\}$, $P = \{S \rightarrow aSb, S \rightarrow \varepsilon\}$.
- $G = (V_N, V_T, S, P)$: $V_N = \{ S, A\}$; $V_T = \{a,b\}$, $P = \{S \rightarrow aA, S \rightarrow b, A \rightarrow aa\}$.
- $G = (V_N, V_T, S, P)$: $V_N = \{ S, A, B\}$; $V_T = \{a,b, c\}$, $P = \{ S \rightarrow aAB, A \rightarrow Bba, B \rightarrow bB, B \rightarrow c\}$.
- Find a *grammar* for a simple arithmetic expression in a

programming language, and present the derivation tree for sample expressions (such as $(a+b)*(c-d)$).

1.5 Chomsky Classification of Grammars

Type 0. Recursively Enumerable Grammar.

No restrictions on productions $\alpha \rightarrow \beta$.

Example:

$G = (V_N, V_T, S, P)$:

$V_N = \{ S, C, D, E\}$;

$V_T = \{a, \$, \#\}$,

$P = \{S \rightarrow \$Ca\#|a|\varepsilon,\ Ca \rightarrow aaC,\ \$D \rightarrow \$C,\ C\# \rightarrow D\#|E,\ aD \rightarrow Da,\ aE \rightarrow Ea,\ \$E \rightarrow \varepsilon\}$.

- Analisis if the word $w=aa$ can be generated by respectively grammar.
  $S \rightarrow \$Ca\# \rightarrow \$aaC\# \rightarrow \$aaE \rightarrow \$aEa \rightarrow \$Eaa \rightarrow aa$

Type 1. Context-Sensitive Grammars.

All production are in the form $\alpha_1 A \alpha_2 \rightarrow \alpha_1 \beta \alpha_2$,

where $\alpha_1, \alpha_2 \in (V_N \cup V_T)^*, \beta \in (V_N \cup V_T)^+, A \in V_N$.

There are:

- left –context-sensitive grammar $\alpha_1 A \rightarrow \alpha_1 \beta$;
- right–context-sensitive grammar $A\alpha_2 \rightarrow \alpha_1 \beta$.

Example:

$G = (V_N, V_T, S, P)$:

$V_N = \{ S, A, B\}$;

$V_T = \{a, b\}$,

$P = \{S \rightarrow abc|aAbc,\ Ab \rightarrow bA,\ Ac \rightarrow Bbcc,\ bB \rightarrow Bb,\ aB \rightarrow aa|aaA\}$.

- The language generated by this grammar is

$L(G)=\{ a^n b^n c^n \mid n\geq 1\}$

Type 2. Context-Free Grammar.

The all productions of grammar G must be in form $A\to\beta$, where

$$A \in V_N, \beta \in (V_N \cup V_T)^*.$$

Example:

$G = (V_N, V_T, S, P)$:

$V_N = \{ S, A, B\}$;

$V_T = \{a, b\}$,

$P = \{S\to aSb|\varepsilon\}$.

Type 3. Regular Grammar.

Type 3 is most restricted grammar and and it has 2 representation as:

- Right linear grammar:

$$A \to \alpha B$$
$$A \to \alpha$$

Example:

$G = (V_N, V_T, S, P)$:

$V_N = \{ S\}$;

$V_T = \{a, b\}$,

$P = \{S \to aS, S \to a\}$.

- Left linear grammar:

$$A \to B\alpha$$
$$A \to \alpha$$

where $\alpha \in V_T^*$

Example:

$G = (V_N, V_T, S, P)$:

$V_N = \{ S, A, B\}$;
$V_T = \{a, b\}$,
$P = \{S \rightarrow Aab, A \rightarrow Aab|B, B \rightarrow a\}$.

Practical Tasks

Identify type of the given grammars:

1. $G= (V_N, V_T, S, P)$, $V_N = \{S\}$, $V_T = \{a, b\}$, $P=\{S\text{->}aS|b\}$.

2. $G= (V_N, V_T, S, P)$, $V_N = \{S, C\}$, $V_T = \{a, b\}$,
$P=\{S\text{->}aaC; C\text{->} \varepsilon; C\text{->}ab\}$.

3. $G= (V_N, V_T, S, P)$, $V_N = \{S\}$, $V_T = \{0,1\}$,
$P=\{S\text{->}0S1; S\text{->} \varepsilon \}$.

4. $G= (V_N, V_T, S, P)$, $V_N = \{S, C, D\}$, $V_T = \{a, b\}$,
$P=\{S\text{->}aaD| \varepsilon; D\text{->}Cb; C\text{->} \varepsilon \}$.

5. $G= (V_N, V_T, S, P)$, $V_N = \{S\}$, $V_T = \{a, b\}$,
$P=\{S\text{->}abS; S\text{->}a\}$.

6. $G= (V_N, V_T, S, P)$, $V_N = \{S, A\}$, $V_T = \{0,1\}$,
$P=\{S\text{->}1S|0A0S| \varepsilon; A\text{->}1A|\varepsilon \}$.

7. $G= (V_N, V_T, S, P)$, $V_N = \{S, A, B\}$, $V_T = \{a, b\}$,
$P=\{\mathrm{S}\text{->}Aab; A\text{->}Aab|B; B\text{->}a\}$.

8. $G= (V_N, V_T, S, P)$, $V_N = \{S\}$, $V_T = \{a, b\}$,
$P=\{S\text{->}Sa|b\}$.

9. $G= (V_N, V_T, S, P)$, $V_N = \{S, A, B, C\}$, $V_T = \{a, b\}$,
$P=\{S\text{->}ACA; AC\text{->}AACA|ABa|AaB; B\text{->}AB|A; A\text{->}\mathrm{a}|\mathrm{b}\}$.

10. $G= (V_N, V_T, S, P)$, $V_N = \{S\}$, $V_T = \{a, b\}$,
$P=\{S\text{->} \varepsilon; S\text{->}ba\}$.

11. $G= (V_N, V_T, S, P)$, $V_N = \{S, B, C\}$, $V_T = \{a, b\}$,
$P=\{S\text{->}aSBC; BC\text{->}B; C\text{->}a; B\text{->}b\}$.

12. $G = (V_N, V_T, S, P)$, $V_N = \{S\}$, $V_T = \{a, b, d\}$,
$P = \{S\text{->}aS|b|Sd|\ \varepsilon\}$.

13. $G = (V_N, V_T, S, P)$, $V_N = \{S, B, C\}$, $V_T = \{a, b\}$,
$P = \{S\text{->}aSBC;\ BC\text{->}Bab;\ C\text{->}a;\ B\text{->}b\}$.

14. $G = (V_N, V_T, S, P)$, $V_N = \{S\}$, $V_T = \{0,1\}$,
$P = \{S\text{->}00S1|\ \varepsilon\}$.

15. $G = (V_N, V_T, S, P)$, $V_N = \{S, C, D\}$, $V_T = \{0,1\}$,
$P = \{S\text{->}CD;\ C\text{->}0C|0;\ D\text{->}1D|1\}$.

16. $G = (V_N, V_T, S, P)$, $V_N = \{S, B, C\}$, $V_T = \{a, b\}$,
$P = \{S\text{->}aSBC;\ BC\text{->}Bab;\ C\text{->}a;\ B\text{->}b|\ \varepsilon\}$.

17. $G = (V_N, V_T, S, P)$, $V_N = \{S, A, B, C\}$, $V_T = \{a, b, c\}$,
$P = \{S\text{->}ABC|ABCS;\ AB\text{->}BA;\ AC\text{->}CA;\ BC\text{->}CB;$
$BA\text{->}AB;\ CA\text{->}AC;\ CB\text{->}BC;\ A\text{->}a;\ B\text{->}b;\ C\text{->}c\}$.

18. $G = (V_N, V_T, S, P)$, $V_N = \{S\}$, $V_T = \{0, 1\}$,
$P = \{S\text{->}0S1S|1S0S|\ \varepsilon\}$.

19. $G = (V_N, V_T, S, P)$, $V_N = \{S, T\}$, $V_T = \{0, 1\}$,
$P = \{S\text{->}1S|0T|\ \varepsilon;\ T\text{->}1T|0S\}$.

20. $G = (V_N, V_T, S, P)$, $V_N = \{S, A, B\}$, $V_T = \{a, b\}$,
$P = \{S\text{->}aAB;\ A\text{->}a;\ B\text{->}\ b\}$.