1 Exercises about introductory part

1. Pigeonhole Principle says: If you have more pigeons than pigeonholes, and each pigeon flies into some pigeonhole, then there must be at least one hole that has more than one pigeon.

   What happens, if you have as many pigeonholes as there are natural numbers, and as many pigeons, as there are integers? What about, if you have as many pigeons as there are natural numbers, but each pigeon tries to make nest with every other pigeon into a different hole? (Only one nest can be made into one hole.)

2. There is coming an infinite number of busses, with infinite number of people in each of them. How would you allocate the guests into rooms of Hilbert's Hotel?

3. In the quest book of Hilbert's Hotel there is only finite number of names in each page and new quests must always write their names into the next empty line. How many pages there must be in the book so that there is room for the new names (without reorganizing the names) as long as there is room in the hotel (maybe after reorganizing)?

4. In the Cantor's planet everything is measured with infinite precision. Each inhabitant has a unique name, which consists of infinite string in alphabet $\{a, b\}$.

   Inhabitants *aaaa*..... and *baaaa*.... have decided to arrange a group tour to Hilbert's Hotel, and they have invited all inhabitants whose names are in alphabetic order between their own names. For example *aaaa*...'s little sister *abaaa*... and *baaaa*...'s cousin's son *abbbb*... are invited. Can you accommodate everybody into Hilbert's Hotel? Justify your answer carefully! (Hint: Cantor's diagonalization method.)

5. Find the error in the following proof that $2 = 1$. Consider the equation $a = b$. Multiply both sides by a to obtain $a^2 = ab$. Subtract b^2 from both sides to get $a^2 - b^2 = ab - b^2$. Now factor each side, $(a - b)(a + b) = b(a - b)$, and divide each side by $(a - b)$, to get $a + b = b$. Finally, let a and b equal 1, which shows that $2 = 1$.

6. Let X be a set and X the size of $n = |X|$. Prove by induction that the size of the powerset of X is $|\mathcal{P}(X)| = 2^n$.

7. What is wrong in the following induction proof that all cats are of the same colour?

   Let n be the number of cats. If $n = 1$ the claim holds clearly (one cat is always of the same colour). Let's now suppose that for any group of n cats the claim holds. Then let's consider a group of $n + 1$ cats. By selecting any n cats from this group (which can be done in $n + 1$ different ways) we get by the induction assumption a group in which all the cats have the same colour. So all $n + 1$ cats must be of the same colour.

8. Prove the following claim. If there are $n (n \geq 2)$ people in the party, then at least two people have equal number of friends in the party.

9. Prove by contraposition: If c is an odd integer number, then the equation $n^2 + n - c = 0$ doesn't have any integer solution for n.

10. Construct a decision problem, which can be used for evaluating the difficulty of the following computational problem!

    (a) Calculate factorial of n, $n!$, when n is natural number.

    (b) Order a given word list into alphabetic order.

    (c) Search from a given graph all cliques i.e. such vertex sets that there is an edge between all vertices in the set.

    Consider also, how to code the input of decision problem as a string. What is the corresponding formal language like?

11. Invent examples about non-computational problems! Could you still use computers somehow to help in solving the problem?

12. Read the story about decision problems
    `http://www.cs.joensuu.fi/pages/whamalai/tepe04/story.html` and complete it!

13. Let's consider the logic school of Kissastan again. This time the topic is a more complicated MIAU-system, which consists of the following rules:

    xUAx $\rightarrow$ xAUy
    xUUx $\rightarrow$ xIUy

x → MxM
x → xUI
xx → x
xI → xUA

The task is to show that even an empty string can create a proper miaow (MIAU) by the rules of the system!

14. Let's consider alphabet $\Sigma = \{m, i, u\}$. We define the the "powers" of alphabet in the following way:

    $\Sigma^0 = \{\epsilon\}$ (empty string)
    $\Sigma^{k+1} = \Sigma \times \Sigma^k = \{ax | a \in \Sigma \text{ and } x \in \Sigma^k\}$.

    E.g. $\Sigma^1 = \{m, i, u\}$, $\Sigma^2 = \{mm, mi, mu, im, ii, iu, um, ui, uu\}$. How many items ("words") does Σ^n contain? What about the whole language $\Sigma^* = \bigcup_{i=0}^{\infty}$?

2 Exercises about regular languages

1. By the UNIX comman egrep (extended grep) you can search for patterns in the text, which can be defined as regular expressions. The basic syntax of `egrep` is following: `egrep <expression> <file>`, in which the expression can be

    - list of characters in the brackets, e.g. $[abcd]$: any of the characters a, b, c, d
    - $(expression1)(expression2)$: concatenation of two expressions
    - $(expression1)|(expression2)$: either expression1 or expression2
    - $(expression)*$: 0 or more times the expression (closure of the expression)
    - $\backslash b$: empty string in the edge of word $\backslash B$: empty string in the middle of word

    Notice! The expression might need single quotation marks ('expression') More information by command man egrep.

    Test the command egrep with input file
    `http://www.cs.joensuu.fi/pages/whamalai/tepe04/esim.c`!

What kind of patterns do you find with the following command?
egrep '(uu)|(sata)|(issa)|(oita)|(alla)' esim.c

Do you invent a simpler command, which finds exactly the poem lines?

2. Construct an egrep-command to find the following lines from esim.c:
   a) Rows, which consist numbers.
   b) Rows, which consist either word *while* or *for*
   c) Rows, which consist number 10
   d) Rows, which consist integers. (Notice! Your command should not accept desimal numbers.)

3. Construct an egrep command for finding email and street addresses and telephone numbers from a html file. Use the homepages of Kissastan Logic School as your input data (`http://cs.joensuu.fi/pages/whamalai/tepe04/kissastania/kissastania.htm`)!

4. Let's consider the following languages of the alphabet $\Sigma = \{a, b\}$. For each of the languages give two strings that are memebers and two strings that are not memebers!

   a) a^*b^*
   b) $a(ba)^*b$
   c) $a^* \cup b^*$
   d) $(aaa)^*$
   e) $(\epsilon \cup a)b$
   f) $\Sigma^*a\Sigma^*a\Sigma^*a\Sigma^*$

5. Which strings belong to the language described by the following expression?
   $(c \cup h \cup m \cup r)at((c \cup t)a \cup (s \cup t)o)ught(m \cup l \cup tw \cup r)ice$

6. Which strings belong to the language $L(\emptyset^*)$? What about $L(\epsilon^*)$?

7. Find a shortest string which belongs to the language described by the following expression!

a) $a^*(b \cup abb)b^*b$
b) $a^*b^*b(a \cup (ab)^*)^*b^*$
c) $(a \cup ab)(a^* \cup ab)^*b$

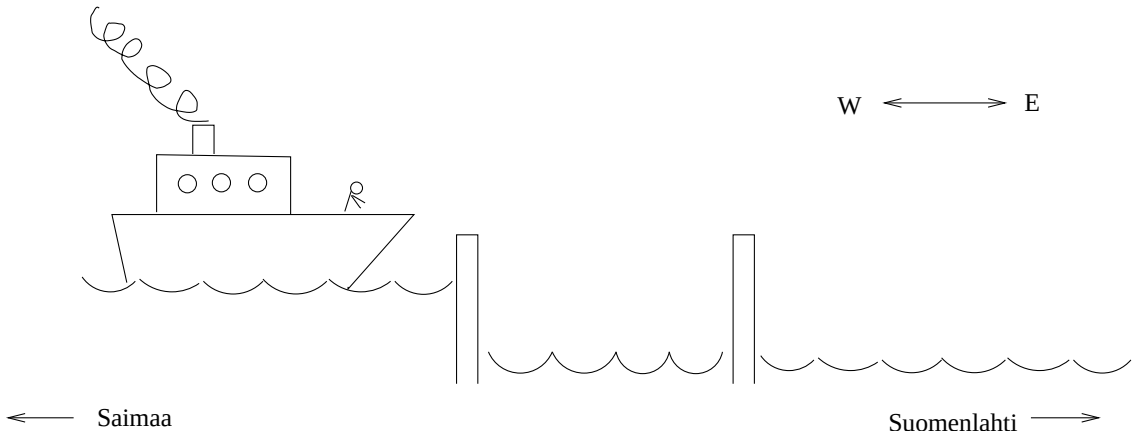
8. Construct the regular expressions corresponding the following languages:
   a) $\{w \in \{a,b\}^* | \text{the third last character of } w \text{ is } a\}$
   b) $\{w \in \{a,b\}^* | w \text{ contains either substring } ab \text{ or } ba\}$
   c) $\{w \in \{a,b\}^* | w \text{ does not cointain string } bab\}$

9. Construct the regular expressions corresponding the following languages:
   a) $\{w \in \{a,b\}^* | w \text{ contains an even number of characters } a\}$
   b) $\{w \in \{a,b\}^* | \text{the length of } w \text{ is odd}\}$
   c) $\{w \in \{a,b\}^* | \text{number of characters } b \text{ is multiple of } 3\}$

10. Let's have alphabet $\Sigma = \{a,b\}$. Construct a finite automaton, which recognizes the following language:

    a) $L(M) = L(\emptyset)$
    b) $L(M) = L(\emptyset^*)$
    c) $L(M) = L(\epsilon)$
    d) $L(M) = L(\epsilon^*)$
    e) $L(M) = L(\Sigma^*)$

11. Simulate the given finite automaton as a game! Each student corresponds to one state, which reads the next input character and transfers the control to the successor state. Accepting or rejecting state reports about result. (One point for each participant!)



12. Construct a lock automaton for a channel, which opens and closes the locks and increases and decreases the water level automatically. The Western lock gate can be opened only when the water is up and the Eastern gate only when the water is down. You can control the water level only when both gates are closed. The automaton gets the following input data from the control system:

    WD = water down
    WU = water up
    SW = a ship from West
    SE = a ship from East
    GC = gates closed
    SL = a ship in lock

    You may suppose that between the ships the lock automaton waits one gate open, until the next ship arrives.

    (Notice! The automaton doesn't have any special initial or final states.)

13. Create a finite automaton that describes the function of a lift moving between two floors. The lift may be either up or down. On both floors, there is a simple "come here" button, and in the lift there is an "up" and a "down" button. In addition, the lift has a door that can be opened and closed; the lift only moves if the door is closed. (The automaton does not have to have specific favorable states.)

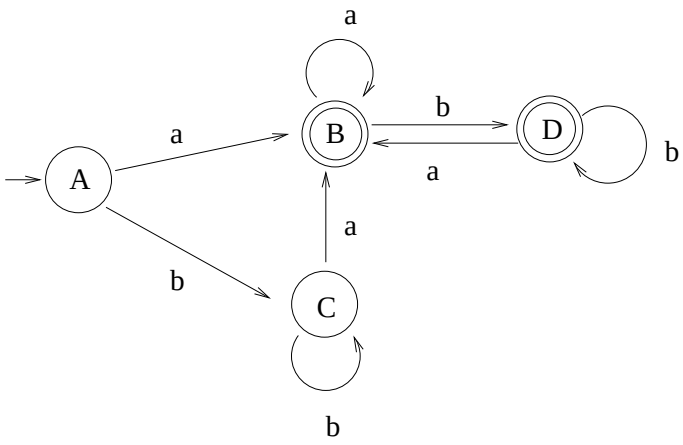
    Input: pushing the buttons, opening and closing the door. States: up/down; door open/closed. Notation: U = up, D = down, o = door open, c = door

closed.

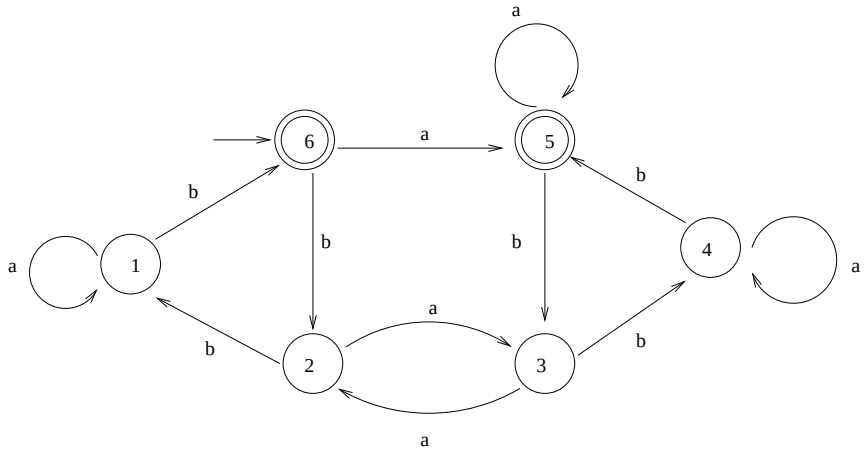
14. Let $\Sigma = \{0, 1\}$. Construct a deterministic finite automaton M, which recognizes the following language

    a) $L(M) = \{w|$the length of w is odd$\}$
    b) $L(M) = \{w|$number of 1s in w is multiple of three$\}$
    c) $L(M) = \{w|w$ has even number of 1s and 0s$\}$

15. Create a minimum automaton that is equivalent to the following deterministic finite automaton.



16. Create a minimum automaton that is equivalent to the following deterministic finite automaton.



17. Generating a regular language: "The Poem Automat"

    Implement a program that creates rows according to the regular expression (the)(ATTR)* SUBJ PRED the (ATTR)* OBJ(ATTRIB$\cup\epsilon$) Languages ATTR, SUBJ, OBJ, PRED and ATTRIB consist of the following words, for example:

    ATTR = {fat, black}
    SUBJ = {cat, moon, fish}
    OBJ = {cat, moon, fish}
    PRED = {shines, watches, swims}
    ATTRIB = {in the sky, in the lake}

    Your program can now produce 'verses' like the following:

    The black cat watches the fat fish in the lake
    The moon swims in the sky

    NB! Specify that not all predicates may be followed by an object (e.g. swims).

    Expand the example as you wish with words that fit into the poem! Add the exclamations

    "What a (ATTR) (SUBJ)!"

and the questions

"Are you a (ATTR) (SUBJ)?"

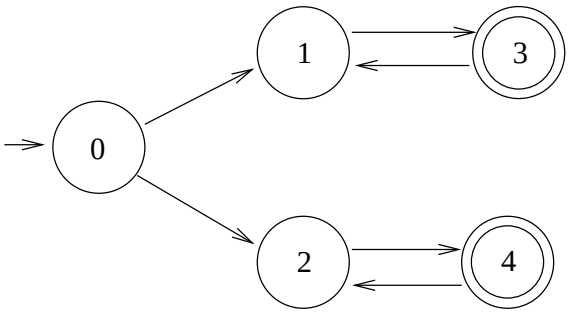
Return the code of your program and examples of your outputs (poems) to the instructor! Be ready to represent your program/poems in the art exhibition!

18. Let's consider the comic in the lecture material about a nondeterministic automaton consulting the doctor. Which middle phases does the operation consist of? Does everything go as it should?

19. Invent at least one phenomenon in your everyday life, which can be described as a nondeterministic automaton. Draw the transition diagram of your automaton and try to determinize it!

20. Create a non-deterministic finite automat that tests whether alphabet $\{a, b\}$ contains the character string "abaa" and make it deterministic.

21. Tranform the following automaton to deterministic!



22. Give nondeterministic finite automata to accept the following languages. Try to take advantage of nondeterminism as much as possible.

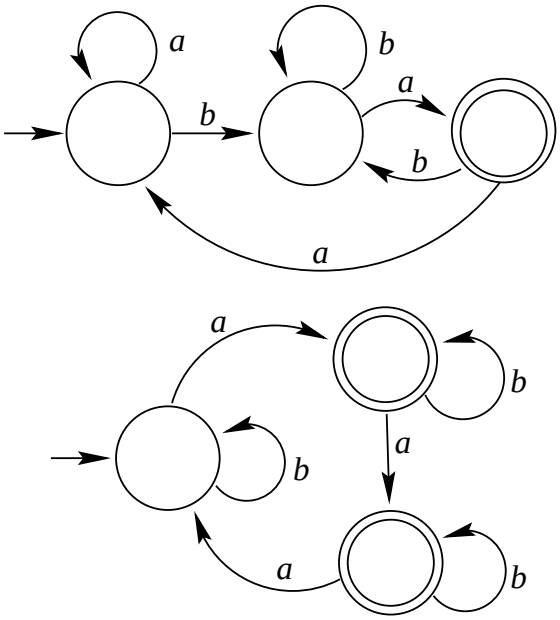
a) The set of strings over alphabet $\{0, 1, ..., 9\}$ such that the final digit has appeared before.
b) The set of strings over alphabet $\{0, 1, ..., 9\}$ such that the final digit has *not* appeared before.
c) The set of strings over alphabet $\{0, 1\}$ such that there are two 0's separated

by a number of 1's that is a multiple of 4. Note that 0 is an allowable multiple of 4.

23. Remove all ϵ-transitions from the Goblins' Gingerbread automaton and transform it to deterministic!
(See `http://cs.joensuu.fi/pages/whamalai/tepe04/redhood.pdf`)

24. What kind of language does the Goblins' Gingerbread automaton recognize? Describe the language as a regular expression!

25. Give the regular expressions corresponding the following automata!



26. Construct the finite automata corresponding the following regular expressions!
a)$(ab)^*(ba)^* \bigcup aa^*$
b)$((ab \bigcup abb)^* a^*)^*)$

27. Show that the set of regular languages is closed under cut and catenation! I.e. if L_1 and L_2 are regular languages, then also $L_1 \bigcap L_2$ and $L_1 L_2$ are regular. (Hint: automata.)

28. Are the following languages regular? Justify your answer carefully!

(a) $\{w|w$ is a character string of length 3 consisting of a's and b's$\}$

(b) $\{ww^*|w \in \{a,b\}^*\}$

(c) $\{w^*|$there are as many 1's as 0's in w

(d) $\{w|$there are twice as many 0's as 1's in w$\}$

29. What happens if you try to prove an actually regular language as non-regular by the Pumping Lemma? Consider for example languages $L_1 = \emptyset$, $L_2 = \{aa, bb\}$, $L_3 = \{ab^*a^*b\}$.

30. What is wrong in the following Pumping lemma proofs?

a) Let $L = \{(aa)^i(bb)^j|i,j \geq 0\}$.
Claim: L is nonregular.
Proof: Let $x = (aa)^n(bb)^k = a^{2n}b^{2n}$, $|x| = 4n$. Now x can be divided into parts uvw only in one way: $u = a^i$, $v = a^j$, $j \geq 1$, $w = a^{2n-i-j}b^{2n}$. Now $uv^0w = a^{2n-j}b^{2n} \notin L$, thus L is non-regular.

b) Let $L = \{c^ra^kb^k|r \geq 1, k \geq 0\} \cup \{a^kb^l|k,l \geq 0\}$.
Claim: L is regular.
Proof: $L = L_1 \cup L_2$, in which $L_1 = \{c^ra^kb^k|r \geq 1, k \geq 0\}$ and $L_2 = \{a^kb^l|k,l \geq 0\}$. For all $x \in L$ either $x \in L1$ or $x \in L2$. Let's consider both cases:
1) If $x \in L_1$, let's select $x = c^na^kb^k$. $|x| = n + 2k > n$. x can be divided into parts uvw only in one way:
$u = c^i$, $v = c^j$, $j \geq 1$, $w = c^{n-i-j}a^kb^k$. Now $uv^kw \in L$ for all $k = 0, 1, 2, ...$, i.e. we can always pump x.
2) If $x \in L_2$, let's select $x = a^nb^l$, $|x| = n + l$. x can be divided into parts in only one way $u = a^i$, $v = a^j$, $j \geq 1$, $w = a^{n-i-j}b^l$. $uv^kw \in L$ for all $k = 0, 1, 2, ...$
$\Rightarrow L$ is regular.

31. We know that in general the problem $REG(L)$ (i.e. is the given language L regular or not) is unsolvable. What is the reason for that? Could you still invent a program, which would help people to prove by the Pumping Lemma that the given language is nonregular?

32. The stronger version of the Pumping Lemma gives both sufficient and necessary conditions for the regularity of a language:

Pumping Lemma 2: The language $A \in \Sigma^*$ is regular if and only if there is a constant $n \geq 1$ such that for all $x \in \Sigma^*$, if $|x| \geq n$ then there exists u, v, w such that $x = uvw$ and $|v| \geq 1$, and for all $i \geq 0$ and all $yin\Sigma^*$ $xy \in A$ if and only if $uv^iwy \in A$.

Based on this, could you make a program, which decides for any given language A, if it is regular or not?

33. Prove with the Pumping lemma that the following languages are non-regular:

    (a) $\{a^n b^n c^k | n, k = 0, 1, ...\}$

    (b) $\{a^n b^k c^k | n, k = 0, 1, ...\}$

    (c) $\{a^n b^n a^m b^m | n, k = 0, 1, ...\}$

34. Let's denote by w^R the string w written backwards (i.e. if $w = a_1 a_2 \dots a_n$, then $w^R = a_n \dots a_2 a_1$). The string is a palindrome if $w = w^R$ (e.g. "madamimadam"). Consider the palindrome language $L_{pal} = \{ww^R | w \in \{a, b\}^*\}$.

    Prove with the Pumping Lemma that L_{pal} is non-regular!

35. Write a program, which transforms a simple HTML file into corresponding latex file. HTML file `<html><body> text </body></html>` corresponds to latex file `\documentclass[a4paper,12pt]{article} \begin{document} text \end{document}`. Text contains only normal text and title definitions. Title definitions `<h1>title</h1>`, `<h2>title</h2>`, `<h3>title</h3>` are generated in latex by `\section{title}`, `\subsection{title}` and `\subsubsection{title}`.

    Is an application based on finite automata powerful enough for checking, if the HTML file is also correct?

36. How could you use regular expressions and finite automata in practice? Give at least three applications!

37. What can you say about time requirements of problems, which belong to the class of regular languages? What about their space requirements? (Hint: Consider the program representation of a finite automaton.) Is there any difference if the corresponding automaton is deterministic or undeterministic?

3 Exercises about context-free languages

1. Let's have a context-free grammar $G = (V, \Sigma, P, S)$, $V = \{A, B, C, D, S\} \cup \Sigma$, in which $\Sigma = \{Jim, big, green, cheese, ate\}$, and rules P are:
   $A \rightarrow B|CA$
   $S \rightarrow ADA$
   $C \rightarrow big|green$
   $B \rightarrow cheese|Jim$
   $D \rightarrow ate$

   What kind of strings belong to the language described by G? Give some exaples! Do the following sentences belong to the language $L(G)$: "big big green cheese", "ate Jim big cheese"?

2. Construct a finite automaton, which recognizes the language described by the following grammar. What is the corresponding regular expression?

   $S \rightarrow aA|bB$
   $A \rightarrow aS|bA$
   $B \rightarrow bB|\epsilon$

3. Give a right-linear grammar that describes the language

   $L = \{w \in \{a, b\}^* | w$ does not contain substring $abaa\}$

4. All words of the language of outerspace aliens follows the Blurbs normalform. Blurb is a Whoozit, followed by one or more Whatzit. Whoozit is letter 'x', which can be followed by any number of letters 'y' (also zero). Whatzit is letter 'q', which is followed by either letter 'z' or 'd', followed by Whoozit.

   Give the context-free grammar, which describes the Blurbs language. Can you now give the corresponding regular expression? (Hint: construct first the automaton from the grammar!)

5. For regular expression holds the following rule::
   If $r = rs \bigcup t$, then $r = ts^*$, when $\epsilon \notin L(s)$.
   Show, why this rule holds with the help of context-free grammars!

6. Create a context-free grammar that produces programming language expressions consisting of an infinite number of 'for' loops inside each other and elementary operations a, in the same way as in the example.

```
for (i = N; i < N; i++) {
    for (j=N; j<N; j++) {
        a;
    }
}
```

in which N is an integer.

7. Describe the following type of C-programs as context-free grammar!

   - the function takes the form

     ```
     fcttype fctname(partype par (,partype par)*)
     {
       fcttrunk
     }
     ```

     in which fcttype may be void or int and partype is int.
   - fcttrunk is an ifexpression or assignment
   - the assignment takes the form variable=value;
   - par and the variable are variable names consisting of one letter
   - the value is an integer
   - the ifexpression takes the form
     if (condition) ifexpression assignment; or
     if (condition) ifexpression assignment; else ifexpression assignment; or
     an empty character string
   - the condition takes the form (variable=value)

8. Draw the following pushdown automaton M:

   $M = (\{q_0, q_1, q_2\}, \{a, b\}, \{A\}, \delta, q_0, \{q_1, q_2\})$, in which
   $\delta(q_0, a, \epsilon) = \{(q_0, A)\}$
   $\delta(q_0, \epsilon, \epsilon) = \{(q_1, \epsilon)\}$

$\delta(q_0, b, A) = \{(q_2, \epsilon)\}$
$\delta(q_1, \epsilon, A) = \{(q_1, \epsilon)\}$
$\delta(q_2, b, A) = \{(q_2, \epsilon)\}$
$\delta(q_2, \epsilon, A) = \{(q_2, \epsilon)\}$

What kind of language does M recognize? Trace all computation paths with input strings *aab*, *abb* and *aba*! Does M accept strings *aabb* and *aaab*?

9. Write that w^R = the character string w backwards (i.e. if $w = a_1 a_2 \dots a_n$, then $w^R = a_n \dots a_2 a_1$). The character string is a palindrome if $w = w^R$ (e.g. "madamimadam"). Consider the language PAL = $\{ w \in \{ a, b \} \mid w = w^R \}$. that is formed by palindromes in the alphabet $\{a, b\}$.

   a) Give a context-free grammar, which produces the language.
   b) Construct a pushdown automaton, which recognizes the language.

10. Create a context-free grammar that produces Roman numbers that are less than 90. The numbers are L=50, X=10, V=5, I=1. XII=12, for example, and XIV=14, XL=40, XLIX=49. For thew sake of simplicity, you may assume that an empty character string signifies 0 and the character string LXL the number 90.

11. Construct a pushdown automaton, which recognizes the Roman numbers that are less than 90, as described in the previous task!

12. In the Logic school of Kissastan the current topic is context-free languages. In this class the cats consider the following language:

    One miaow can begin with one or more *miu* and in the end there is equal number of *mau*'s. Between them can be 0 or more *u*'s. Or the miaow is maow, which is followed by one or more *au*. Maow begins with *miau* or *mau*, which is followed by 0 or more *u*'s.

    a) Give a context-free grammar, which produces the language.
    b) Give some examples words of this language.
    c) Construct a pushdown automaton, which recognizes this language.

13. What kind of language does the following pushdown automaton M recognize?

    $M = (\{q_0, q_1\}, \{a, b\}, \{\underline{A}, \underline{A_1}, \underline{B}, \underline{B_1}, A, A_1, B, B_1\}, \delta, q_0, \{q_0\})$

$\delta(q_0, a, \epsilon) = \{(q_1, \underline{A_1})\}$
$\delta(q_0, b, \epsilon) = \{(q_1, \underline{B})\}$
$\delta(q_1, a, \epsilon) = \{(q_1, \underline{A_1})\}$
$\delta(q_1, a, B) = \{(q_1, B_1)\}$
$\delta(q_1, a, B_1) = \{(q_1, \epsilon)\}$
$\delta(q_1, a, \underline{B}) = \{(q_1, \underline{B_1})\}$
$\delta(q_1, a, A_1) = \{(q_1, A)\}$
$\delta(q_1, a, \underline{A_1}) = \{(q_1, \underline{A})\}$
$\delta(q_1, b, \epsilon) = \{(q_1, B)\}$
$\delta(q_1, b, A) = \{(q_1, \epsilon)\}$
$\delta(q_1, b, A_1) = \{(q_1, B_1)\}$
$\delta(q_1, b, \underline{A_1}) = \{(q_1, \underline{B_1})\}$

14. Show that the following grammars are ambiguous! Can you give an unambiguous grammar which describes the same language?

    (a) $S \to aSb|SS|\epsilon$

    (b) $S \to aSb|aaSb|\epsilon$

15. Construct a grammar which produces language $\{a^m b^n c^{m+n} | m, n \geq 0\}$. Is your grammar ambiguous or unambiguous?

16. What does it matter if a language is inherently ambiguous?

17. Show that the following languages are context-free!:

    a)$\{a^m b^n | m \geq n\}$
    b)$\{a^m b^n c^p d^q | m + n = p + q\}$
    c)$\{uawb | u, w \in \{a, b\}^*, |u| = |w|\}$
    d)$\{a^m b^n | n \leq m \leq 2n\}$

18. Let $\Sigma = \{m, i, u, a\}$. Let's consider the grammar:
    $S \to miuNmauS|SmiauA|\epsilon$
    $N \to miuNmau|U$
    $U \to uU|\epsilon$
    $A \to Aau|\epsilon$

    Give the leftmost and the rightmost derivation and the parsing tree for the string

miumiuuumaumaumiaumiauau! Is the grammar unambiguous or ambiguous?

19. Does the following grammar satisfy the LL(1) condition?

    $S \rightarrow (L)|p|q$
    $L \rightarrow LandS|LorS|S$

    What kind of language does it decribe?

20. Give (as pseudocode) a recursive parser for the language described by the previous grammar. (N.B. Transform it first to LL(1) if needed!)

21. Let L be a language, the words of which are constructed from any text and legal parantheses structures. I.e. as in the problem 8, but now there can be text in the middle of parantheses. E.g. sentence "a (big) cat animal (either lion or tiger {which are rare } or saber-toothed tiger {which extincts [read further Kurten] – shame –} or leopard) is an attractive (but dangerous!) friend" belongs to the language. You can suppose for simplisity that the text parts consist of only lowercase letters a..z and spaces.

    Give a LL(1) grammar, which describes the langauge, and design for it a recursive parser!

22. Let's consider a simple programming language, in which all programs are of form *begin* SENTENCE; *end*. SENTENCE can be a simple assignment VARIABLE:=FACTOR;, VARIABLE:=FACTOR+FACTOR; or VARIABLE:=FACTOR-FACTOR;, in which FACTOR is either variable $a, ..., z$ or unsigned integer. Or SENTENCE is *while* structure: *while* (CONDITION) *do begin* SENTENCE *end*. CONDITION compares two factors with operator <,<=,>,>=,==,<>.

    Construct a grammar for the language and describe, how it is parsered recursively!

23. Create a context-free grammar, which describes the polynoms of a variable x. For the sake of simplicity, you may assume that the coefficients and the exponents of the terms are integers of one digit and that the first term is without prefix. The terms do not have to be in a specific order, and there may be temrs of the same denomination in the polynom. Give the derivation trees of the following strings in your grammar: $2 * x\text{^}2 - 2 * x + 1$, $x - x\text{^}2 - 1 + 2 * x\text{^}2 - 3x$, x, 5! Can you construct a recursive parser for the language?

24. Write a program which transforms HTML list structures into corresponding latex lists. HTML lists are of form `<ul>(<li>text)+</ul>` and `<ol>(<li>text)`$^+$`</ol>` and in latex the corresponding lists are `\begin{itemize}(\item text)`$^+$`\end{itemize}` ja `\begin{enumerate} (\item text)`$^+$`\end{enumerate}`. Notice that there can be several nested lists!

25. Show that the following languages are deterministic:
    a) $\{a^m b^n | m \neq n\}$
    b) $\{wcw^R | w \in \{a, b\}^*\}$
    c) $\{a^m cb^m\} \bigcup \{a^m db^{2m}\}$

26. Transfor the grammar

    $S \rightarrow (S)|A$
    $A \rightarrow SS|\epsilon$

    into Chomsky normal form. Give also the middle steps (removing ϵ-productions and unit productions)!

27. Transfor the grammar

    $S \rightarrow ABC|a$
    $A \rightarrow aAaa|\epsilon$
    $B \rightarrow bBbb|\epsilon$
    $C \rightarrow cCa|c$

    into Chomsky normal form. Give also the middle steps!

28. Simulate the CYK-algorithm, when it decides, if the strings bbaab, ababab and aabba belong to the language described by the grammar

    $S \rightarrow AS|b$
    $A \rightarrow SA|a$

    If the answer is yes, give also the corresponding parse tree.

29. Program the CYK algorithm for grammar

    $S \rightarrow AB|BC$
    $A \rightarrow BA|a$
    $B \rightarrow CC|b$
    $C \rightarrow AB|a$

    Test your program with different strings!

30. How would you parse the following language? It is enough to give the basic strategy.
    a) HTML
    b) Clauses of propositional logic, which consist of atomic clauses $A, B, ..., Z$, operations $\neg$, $\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$, and parantheses.
    c) C- or Pascal-source code
    d) SQL-query

31. Let $L_1 = \{a^n b^{2n} c^m | n, m \geq o\}$ and $L_2 = \{a^n b^m c^{2m} | n, m \geq o\}$. Is $L_1 \bigcap L_2$ context-free language? Justify your answer!

32. Design an efficient algorithm, which recognizes, if the given nonterminal symbol is nullable (i.e. can it produce ϵ in some derivation)!

33. Prove that context-free langauges are closed under union, concatenation and closure. I.e. if L_1 and L_2 are context-free, then

    a) $L_1 \bigcup L_2$
    b) $L_1 L_2$
    c) L_1^*

    are context-free. (Hint: Suppose that there exists pushdown automata $M(L_1)$ and $M(L_2)$ and construct pushdown automata for combined languages.)

34. Language $\{a^n b^n c^n | n \geq 0\}$ cannot be recognized by a common pushdown automaton. Could it be recognized, if you had two stacks available? If it could, draw the transition diagram of the automaton and simulate its behaviour! If not, then justify, why not!

35. Prove by the Pumping Lemma for context-free languages that $\{ww \; w \in \{a, b\}^*\}$ is not context-free!

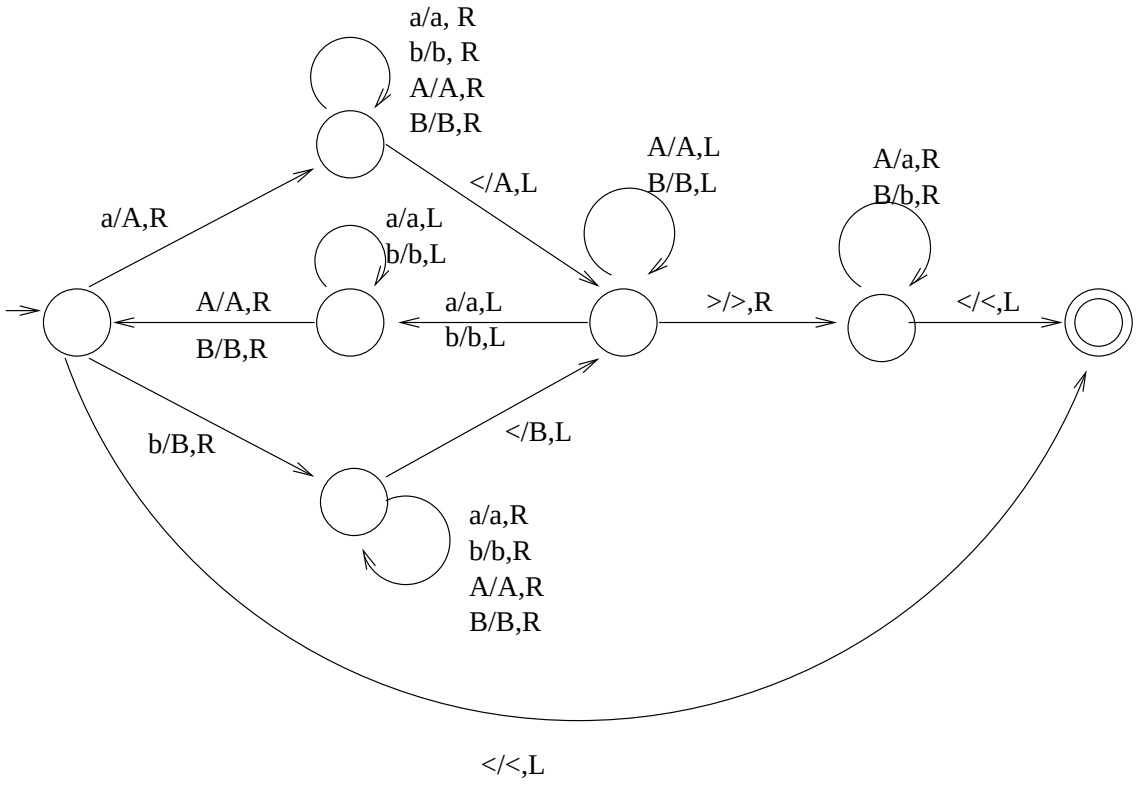
36. Prove by the Pumpinh Lemma for context-free languages that $\{a^p | p$ is prime$\}$ is not context-free!

4 Exercises about Turing machines

1. Simulate the behaviour of a given Turing machine as a game. (Your exercise teacher will have some transition diagrams of different size of Turing machines.) If there are enough students in the group, you can make a competition

between the machines in recognizing a language. Each student corresponds a state, performs a required writing operation on the tape, moves the reading head and gives the control to the correct successor state. The accepting or rejecting final state reports the result. (One point for each participant of the game!)

2. What does the following Turing machine do? Simulate its behaviour with different input strings of as and bs!



3. Construct a Turing machine, which reads the input string, until it finds two consecutice as. The input alphabet is $\{a, b\}$.

4. Construct a standard Turing machine, which subtracts one from the input binary string. I.e. an integer n is given as a binary string x, in which the most significant bits are left and least significant right (e.g. 8=1000). If $n > 0$, the

machine replaces x by the binary representation of integer $n-1$. If $n=0$, the tape remains the same, and the machine moves to the rejecting final state.

5. Construct a Turing machine, which divides the input number represented as binary number by two, if it is even. With odd numbers the machine transfers to the rejecting final state. The binary numbers are represented

   a) the most significant bit on right
   b) the most significant bit on left

6. Construct a standard Turing machine, which recognizes the language $\{ww^R | w \in \{a,b\}^*\}$.

7. Construct a standard Turing machine, which transform the string w into string ww^R ($w \in \{a,b\}^*$).

8. Construct a standard Turing machine, which recognizes the language $\{w \in \{a,b\}^* | w$ contains equal number of a's and b's$\}$.

9. Construct a standard Turing machine, which lists all words in language 1^n, $n = 0, 1, ...$ The machine begins with emty tape, and generates unary numbers 1, 11, 111, 1111, ... (Notice! Your machine will never halt.)

10. Construct a Turing machine, which generates binary representations of all natural numbers 0,1,00,01,10,11,000,001,... You can represent the binary numbers as the least significant bit on left.

11. Construct a 2-tape Turing machine, which gets input string $w \in \{a,b\}^*$ on its 1. tape, and writes string w^R ($=w$ in reversed order) onto 2. tape. You can use a special direction S=stay.

12. Construct a 3-tape Turing machine, which calculates bit-and operation. Binary strings p and q are given on the first two tapes and the machine calculates the result $p\&q$ onto thrid tape. You can assume that p and q are of equal length.

13. Construct a 3-tape Turing machine for multiplying unary numbers. Unary presentations of numbers p and q are given on the first two tapes (e.g. $p=5$ is coded as 11111) and the machine calculates the unary presentation of the

product $n = p * q$ onto third tape. You can use a special direction S=stay. How would you implement the same with 1-tape standard machine?

14. Let's consider the following nondeterministic Turing machine:

    $M = (\{q_0, q_1, q_2, q_f\}, \{0, 1\}, \{0, 1\}, \delta, q_0, q_f, q_{no})$,

    whose transition diagram is defined as

    $\delta(q_0, 0) = \{q_0, 1, R), (q_1, 1, R)\}$
    $\delta(q_1, 1) = \{q_2, 0, L)\}$
    $\delta(q_2, 1) = \{q_0, 1, R)\}$
    $\delta(q_1, <) = \{q_f, <, R)\}$

    What does the machine do? Hint: simulate its behaviour with different binary strings. (You can use JFLAP, if you want.)

15. Construct a nondeterministic Turing machine, which recognizes the language $\{ww | w \in \{a, b\}^*\}$.

16. Consider the nondeterministic Turing machine TEST_COMPOSITE (look at the English material given to you), which recognizes composite numbers. Could you make a prime tester machine by changing the accepting and rejecting states of the machine? Justify your answer!

17. Describe (unformally) a nondeterministic Turing machine, which recognizes the following language: The words of the language are of form $w_1\#w_2\#...\#w_n$ for any n such that for all i $w_i \in \{a, b\}^*$ and for some j w_j is the binary representation of integer j. N.B.! Utilize the nondeterminism as much as possible, i.e. prefer much branched but short paths. (The machine guesses the correct path nondeterministically.)

18. Construct a nondeterministic 2-tape Turing machine, which solves subset sum problem. Factors $x_1, ..., x_n$ are given on the first tape, as unary numbers, separated by character $\#$, and number K is given on the second tape as unary number. The machine should check if the first tape contains subset sum, which is K. E.g. in 1#1111#11 we can find subset sum $K = 111$. (I.e. we should select non-deterministically the factors, which sum up to K, if such exist.)

19. Construct a standard Turing machine, which begins with empty tape and generates as many 1s as possible on its tape before halting.(The machine must halt finally!) The machine can be composed of at most 3 states and the accepting final state.

20. Give an unrestricted (or context-sensitive) grammar, which produces language $L = \{a^i b^{2i} a^i | i > 0\}$. Give the derivation for string *aabbbbaa* in the grammar!

21. Chomsky thought that natural language can be described by context-sensitive grammars, in which all productions can be represented in form:

    $S \rightarrow \epsilon$ or $\alpha A \beta \rightarrow \alpha \omega \beta$,

    in which A is non-terminal symbol and $\omega \neq \epsilon$. Give some example of (syntactic) structure in natural language, which requires such "context" α_β! (You can consider examples in your own native language, but explain the feature also in English.) Do you believe that all structures of natural language can be described with context-sensitive grammars?

5 Exercises about solvability and unsolvability

1. The Universal Turing machine has invited all Turing machines of the universum into Universal Truring Symposium, which is hold in Hilbert's Hotel. The room reservations have been done with the codes c_M of the Turing machines. Is there enough rooms for all Turing machines, if there are no other guests in the hotel? How would you allocate the rooms?

2. Draw the Turing machine described by the following code!

    11101010010100110100101001001101000010000100001011001010101001100100100100100110010000100010000101111.

3. What are the codes of the following Turing machines?

    (a) $\delta(q_0, 0) = (q_0, 0, R)$
    $\delta(q_0, 1) = (q_0, 1, R)$
    $\delta(q_0, <) = (q_{yes}, <, L)$

    (b) $\delta(q_0, 0) = (q_1, 0, R)$
    $\delta(q_0, 1) = (q_{no}, 1, R)$

$\delta(q_0, <) = (q_{no}, <, L)$
$\delta(q_1, 0) = (q_{yes}, 0, R)$
$\delta(q_1, 1) = (q_0, 1, R)$
$\delta(q_1, <) = (q_{yes}, <, L)$

4. Write a program, which gets a transition function of a Turing machine as its input, and outputs its code. You can give the transition function in text file in the following form: on the 1. row the number of states n (thus $q_{no} = q_n$) and on the following rows the transitions $\delta(q, a) = (q', b, \Delta)$ written as $qaq'b\Delta$.

5. Simulate the behaviour of the universal Turing machine, when it is given as its input $c_M w$ the code of the machine in b-part of the previous task and a string 0101!

6. Construct a finite automaton, which recognizes the legal codes of Turing machines. What is the corresponding regular expression?

7. Construct a standard Turing machine, which recognizes the legal codes of Turing machines.

8. Let the languages A and B be recursive enumerable. (i.e. there exists Turing machines M_A and M_B, which recognize the given languages and halt in "Yes"-case, but not necessarily in "No"-case.) Prove that languages $A \cup B$ and $A \cap B$ are also recursive enumerable i.e. you can construct for them Turing machines, which halt at least in "Yes"-case for all input strings!

9. Let A be a recursive enumerable language. Can you construct a Turing machine for its complement language $\overline{A}$ from the machine M_A by changing the accepting and rejecting final states?

10. Prove that language $\{c_M | M \text{ halts on input } \epsilon\}$ is recursive enumerable but not recursive! I.e. invent for it a Turing machine, which halts in "Yes"-case, but not necessarily in "No"-case!

11. Give an example of a Turing machine, which

    a) accepts its own code.
    b) does not accept its own code.

12. Are the following properties of Turing machines syntactic or semantic? If the property is syntactic, how it can be solved?

    a) Fron M's initial state there is at least one transition with a, which does not lead to error state.
    b) If M is started with input a, it transfers to rejecting state.
    c) The computation of M takes with any input at most 10 steps.
    d) M has at most 10 transitions.
    e) From M's initial state you can get into the final state without writing anything.

13. Show that the following semantic properties of recursive enumerable languages are nontrivial (trivial property holds for none or for all languages).

    a) L contains string w.
    b) L is finite.
    c) L is regular.
    d) L is $\{0,1\}^*$.

14. Give two examples of trivial properties: one, which doesn't hold for any recursive enumerable language and the other, which holds for all!

15. Which of the following properties of Turing machines are solvable? (Hint: either invent an algorithm idea or consider, if the property is semantic.)

    a) M halts on all even binary numbers.
    b) If M is started with empty input, it reaches state q on at most 10 steps, or if it is started with input a it reaches state p on at most 20 steps.
    c) M doesn't contain any transition into state q, in which it would write end character $<$.
    d) State q can be reached from state p on at most 3 steps.
    e) M has less than 100 states and M halts on input 0.

16. We know that for recursive reduction of problems $\leq_m$ holds:

    i) If $A \leq_m B$ and B is recursive enumerable, then A is recursive enumerable.

    i) If $A \leq_m B$ and B is recursive, then A is recursive.

    Is language X recursive, recursive enumerable or totally unsolvable in the following cases?

a) For universal language U holds $U \leq_m X$.
b) For language H and for an unknown recursive enumerable language Y holds $H \leq_m Y$ and $Y \leq_m X$. ($H = \{c_M w | M$ halts on input $w\}$)
c) For H's complement $\overline{H}$ holds $\overline{H} \leq_m X$.
d) For Hamiltonian Cycle -language $HC = \{x | x$ codes graph G, which contains Hamiltonian cy
holds $X \leq_m HC$.
e) $HC \leq_m X$

N.B.! Remember the contraposition rule: $A \Rightarrow B \equiv \neg B \Rightarrow \neg A$.

17. Prove that the following property is unsolvable: Given a code of a Turing machine c_M, state q and input w. Does M enter by input w into state q?

18. Are the following languages recursive, recursive enumerable or totally unsolvable?

    a) $L_1 = \{c_M |$The code of machine M c_M is palindrom$\}$.
    b) $L_2 = \{c_M | c_M$ is code of machine M and M recognizes all palindroms in alphabet $\{0, 1\}\}$.

    (Definition of palindroms: Denote $w = a_0 a_1 ... a_n$, if $a_0 a_1 ... a_n = a_n a_{n-1} ... a_0$, then w is palindrom.)

19. Are the following problems solvable, partially solvable or totally unsolvable?

    a) Does the given Turing machine halt on all input?
    b) Does the given Turing machine halt on no input?
    c) Does the given Turing machine halt on at least one input?
    c) Does the given Turing machine fail to halt on at least one input?

20. The following problems are known to be unsolvable, but are they partially solvable or totally unsolvable?

    a) Does the language $L(M)$ contain at least two strings?
    b) Is $L(M)$ finite?
    c) Is $L(M)$ context-free language?

[1] Hamiltonian cycle=cycle, which goes through all vertices exactly once.

21. Invent some concrete examples of unsolvable problems! (Other than halting problem.) N.B.! Justify the unsolvability. Is the problem partially solvable (i.e. recursive enumerable language) or totally unsolvable problem?

22. Are the following problems solvable or unsolvable? If they are unsolvable, tell if they are partially solvable or totally unsolvable. If they are solvable, give the weakest type of machine (finite automaton, pushdown automaton, Turing machine), which solves the problem!

    a) Give the integer nollakohta? for arbitrary polynom $a_1x^n + a_2x^{n-1} + ... + a_1x + a_0$!

    b) Program for compiler an a error checker, which gets as its input a code of a program and checks that the program doesn't perform division by 0 during it execution.

    c) Seach if the given text file contains either words **cat** and **black** or words **bird** and **white**, but not both.

    d) Check that in the given text file word "animal" is followed by word "wise", only if word "cat" appears in the same sentence.

    e) Construct a general virus tester, which recognizes all "dangerous" programs i.e. such programs, which can change system files during their execution.