Exam Paper
Formal Languages and Compilers
(A.Y. 2014/2015)

February 24^{th}, 2015

First name: **Last name:**

Matriculation n.: **e-mail:**

Lexical Analysis

Ex. 1 (8pts)

Derive automata and write regular expressions (RegExp) for the following languages:

- All strings on the alphabet $\Sigma = \{a, b, c, \cdots, z\} \cup \{-, /, +, \cdot\}$ representing correct arithmetic expressions where letters are used to represent variables ("$a + b$, $a + b + c$", "$a \cdot b + c - z$" – OK; "$a+$", "$a - bc$", "$a + \cdot b$" – NOK)
- All strings on the alphabet $\Sigma = \{-, 1, 0\}$ representing a sequence of consecutive triples of numbers for which the sum is 0 for each triple (e.g. "-101", "1-10", "000", "-11010-1" – OK; "1101", "1-01", "10-1101" – NOK).
- All strings on the alphabet $\Sigma = \{a, b\}$ not containing a subsequence constituted by the two substring aa and bb. (e.g. "ababba", "a", "bb", "aab", "baabab" – OK; "aababab", "babaababba", "aabb", "aaabbc" – NOK).

Syntax Analysis

Ex. 2 (12pts)

Let's G the grammar defined by the following productions:

$$S \longrightarrow Ac \quad A \longrightarrow aA \mid acB \quad B \longrightarrow bB \mid bA \mid b \tag{1}$$

1. Discuss the applicability of parsing LL(1) clearly stating why in case the parser is not applicable why is not applicable. In the last case modify the grammar in order to be parsed by a parser LL(1).
2. derive the FIRST, FOLLOW and nullable sets for the original grammar G.
3. Discuss the applicability of parser LR(0) and SLR(1) for the original grammar G.

Semantic Analysis

Ex. 3 (13pts)

Let's G the grammar defined by the following productions that permit to define words representing nested lists of numbers:

$$S \longrightarrow N \quad N \longrightarrow N; M \mid M \quad M \longrightarrow (N) \mid digit \tag{2}$$

Therefore sentences that can be generated by the grammar are: `((12;11);14)`, `((5);(8;3);15)`

Answer to the following requests:

1. define attributes and semantic rules (SDD) for the grammar, in order to permit:
   - the calculation of the sum of all numbers appearing in a sentence
   - the calculation of the number of parenthesis opened in the sentence
   - the printing of the position in the list for each number in the sentence (consider that to implement this feature you could need accessory attributes).
2. show the evaluation tree for the sentence `(5;(8))`
3. modify the grammar in order to be parsable by an LL(1) parser modifying the productions and rules to obtain a suitable L-attributed translation scheme.