Mathematical linguistics – laboratory

task 5: Simple syntax scanner using regular expressions – continuation

During last laboratory we designed a program which used regular expressions to check the compliance of input sentence.

Today we are going to use regular expressions to check the compliance of a sentence with given grammar.

Firstly:

1. **How to migrate from grammar definition to regular expression:**

We should use MBNF notation to design a grammar. It encloses some additional symbols:

[ ] a symbol appears 0 or 1 time

{ } a symbol appears unspecified number of times

( ) means a group of symbols

The migration from a grammar to regular expression will be introduced on the basis of simple grammar valid for an e-mail address.

We design the grammar:

```
S::= A@A.W
A::= W{.W}
W::= L{L}
L::= a | b | c | d | e | … | x | y | z
```

Next, we substitute symbols to each other. As a result we get:

```
S::=(a|b|c|…|z){(a|b|c|…|z)}{.(a|b|c|…|z){(a|b|c|…|z)}}
@(a|b|c|…|z){(a|b|c|…|z)}{.(a|b|c|…|z){(a|b|c|…|z)}}
.(a|b|c|…|z){(a|b|c|…|z)}
```

Such form is almost a regular expression in itself:

```
^[a-z]+(\.[a-z]+)*@[a-z]+(\.[a-z]+)*\.[a-z]+$
```

Please notice: if there is a recurrence call of a symbol, there is no possibility to substitute it. We can not build a regular expression to check given grammar.

Example:

```
S::= W;{W;}
W::= POP{OP}
P::= L|(W)
L::= C{C}
C::= 1| 2 | 3| 4 | 5 | 6 | 7 | 8 | 9 | 0
O::= *| / | + | - | ^
```

If we try to substitute P in W, there will be recurrent call of W. We will not be able to substitute followed symbols.

Task:

Design a grammar.
Design a regular expression to check the compliance of input expression with given grammar.
Build a syntax scanner which will use regular expression to check compliance of input sentences.

Requirements:

Level 1:
Design a grammar which will allow to input following sentences. Sentences should be arithmetic operations involving integers and following operations: addition, subtraction, multiplication, dividing and exponentiation.
Program a simple syntax scanner for that grammar using one of high level languages which will take as input a sentence. As a result there should be a message displayed about the compliance (or not) of the input sentence with designed grammar. A console version is possible.

Level 2:
A GUI version is necessary. It is required to input the sentence via keyboard, trigger the start of the check with a button. As a result there should be a message displayed about the compliance (or not) of the input sentence with designed grammar.

example sentences for input: **`12+2*9; 3*8^12-2/3;`**

Level 3:
Design a grammar which additionally allows the use of fraction numbers and brackets.

example sentences for input: (**`1.2*3)+5-(23.4+3)^3; 8.2/4;`**

All cases require a grammar and a regular expression written enclosed in a text file within the project directory.