Automaten en Formele Talen Assignment 2
Syntax Analysis

Deadline: 23:59, Wednesday, May 9, 2018

Inge Bethke

Created by Daan de Graaf

Last changes on: April 6, 2018 (Bas van den Heuvel)

Background

As seen in the previous assignment, using an automaton to do syntax analysis might cause problems. In order for us to parse our source-code correctly, we will use a Push-Down Automaton (PDA). In the framework you will find the Context Free Grammar (the `CFG` class in `CFG.py`) for the arithmetic expressions and a parse table. The parse table shows which production to use based on the stack-symbol and the input symbol.

Assignment 2 (100 points)

In order for our new parser to work, you will need to program a stack class and a parse function.

Assignment 2.1: The Stack (35 points)

Since the PDA uses a stack, we want to recreate this using a class. Create some methods which you will need to use later on, for example one to determine the length of the stack or to push items on top of it.
N.B.: Instead of a parse tree, the framework uses *semantic actions*. They are embedded in the CFG and can be recognised by the square brackets. We will use them in the third assignment to determine which actions to take.

Assignment 2.2: The Parser (65 points)

The parser should work as follows. First it should take the stack symbol (initially `START`) and the first item of the input stream. Use the parse table to determine which production to use. Those symbols that can be replaced by other symbols are non-terminals. Non-terminals can be found on the left-hand side of the equation in the grammar. You can use the `isNonTerminal()` and `isTerminal()` functions from the `CFG` class (in `CFG.py`).
Terminals are symbols that can not be replaced by other symbols. They are characters and semantic actions. Once you find a terminal on top of the stack, you can push it to a list as a tuple (of its value and type). This list will eventually be the output. Once the stack is empty, the parse was successful and the syntax is correct.

Submission

Put your version of `PDA.py` in a zip or tar with your name and student number and upload it to Canvas **before 23:59, Wednesday, May 9, 2018**. *If your code does not work with Python 3, your work will not be graded and you will receive 0 points.*