# Interpreter LISP - Michał Flak

## Wstęp

Lispy są wyjątkowo prostymi językami do parsowania. W tym projekcie implementuję interpreter podzbioru Scheme.
Użyłem bibloteki SLY oraz języka Python.

## 1. Lekser

```
class LispLexer(Lexer):
    # Set of token names.   This is always required
    tokens = { SYMBOL, STRING, NUMBER }

    SYMBOL = r'[a-zA-Z_+=\*\-\<\>][a-zA-Z0-9_+\*\-\<\>]*'

    @_(r'"(.*?)"')
    def STRING(self, t):
        t.value = t.value[1:-1] # Strip quotes
        return t

    @_(r'\d+')
    def NUMBER(self, t):
        t.value = int(t.value)   # Convert to a numeric value
        return t

    literals = { '(', ')' }

    # String containing ignored characters between tokens
    ignore = ' \t'

    ignore_comment = r'\#.*'
    ignore_newline = r'\n+'
```

## 2. Parser

Gramatyka (EBNF):

```
seq = expr , seq | ;
expr = SYMBOL
       | STRING
       | NUMBER
       | list ;
list = "(" , seq , ")" ;
```

Przykładowe wywołanie:

```
(venv) work@pop-os:~/repos/lisp-interpreter$ python3 lisp-interpreter.py -f tests/fizzbuzz.scm
Parser debugging for LispParser written to parser.out
(do ((i 1 (+ i 1)))
    ((> i 100))
    (display
      (cond ((= 0 (modulo i 15)) "FizzBuzz")
            ((= 0 (modulo i 3))  "Fizz")
            ((= 0 (modulo i 5))  "Buzz")
            (else                i)))
    (newline))
[['do', [['i', 1, ['+', 'i', 1]]], [['>', 'i', 100]], ['display', ['cond', [['=', 0, ['modulo', 'i', 15]], 'F
```

Kod źródłowy:

```
from sly import Parser
from lexer import LispLexer

class LispParser(Parser):
    # Get the token list from the lexer (required)
    tokens = LispLexer.tokens
    debugfile = 'parser.out'

    # Grammar rules and actions
    @_('expr seq')
    def seq(self, p):
        return [p.expr, *p.seq] if p.seq else [p.expr]

    @_('')
    def seq(self, p):
        pass

    @_('SYMBOL',
       'STRING',
       'NUMBER',
       'list_'
      )
    def expr(self, p):
        return p[0]

    @_('"(" seq ")"')
    def list_(self, p):
        return p.seq
```

## 3. Ewaluacja drzewa

TODO

## Źródła


gramatyka lisp: https://theory.stanford.edu/~amitp/yapps/yapps-doc/node2.html
dokumentacja SLY: https://sly.readthedocs.io/en/latest/sly.html
programy testowe: http://rosettacode.org/wiki/Category:Scheme