CSC402 - Assignment #1

Introduction

Consider the following grammar that defines a simple Lisp-like language[1], call it μLisp:

```
prog : sexp prog
     | ""

sexp : '(' exp ')'
     | var
     | num

exp : '+' sexp sexp
    | '-' sexp sexp
    | '*' sexp sexp
    | '/' sexp sexp
    | 's' var sexp
    | 'p' sexp

var : 'x' | 'y' | 'z'

num : '0' | '1' | '2' | '3' | '4' | '5' | '6' | '7' | '8' | '9'
```

Programs in this language are sequences of (possibly nested) parenthesized expressions:

1. (p (+ (* 3 2) 1))
2. ```
   (s x 1)
   (s y x)
   (p (+ x y))
   ```
3. ```
   (s x 1)
   (p x)
   (s y 2)
   (p y)
   (p (+ x y))
   ```

Your processor should *reject* the following program:

```
(s x 1)
(s y x)
(p (+ x p))
```

[1] https://en.wikipedia.org/wiki/Lisp_(programming_language)

Problems

1. Develop a grammar extended with the appropriate lookahead sets for the rules in the grammar above (10 pts).

2. Write a recursive descent sLL(1) parser for the μLisp language defined above in Python based on your extended grammar (Hint: you can use the code exp0_recdesc.py as a starting point. This code is available in the 'code' folder of the Plipy book) (10 pts).

3. Develop a Jupyter notebook that demonstrates that your parser works by parsing the four example programs above - note, your parser should reject the last program (10 pts).

Hand in your extended grammar (as a text file or notebook) and source code together with a Jupyter Notebook that shows that your program works. To submit your work create a zip file of your sources and the notebook and submit it through BrightSpace. Assignments submitted in formats other than Jupyter Notebooks will not be graded and a failing grade will be recorded. An example assignment notebook can be found in the code folder in the plipy github repository (https://github.com/lutzhamel/plipy).