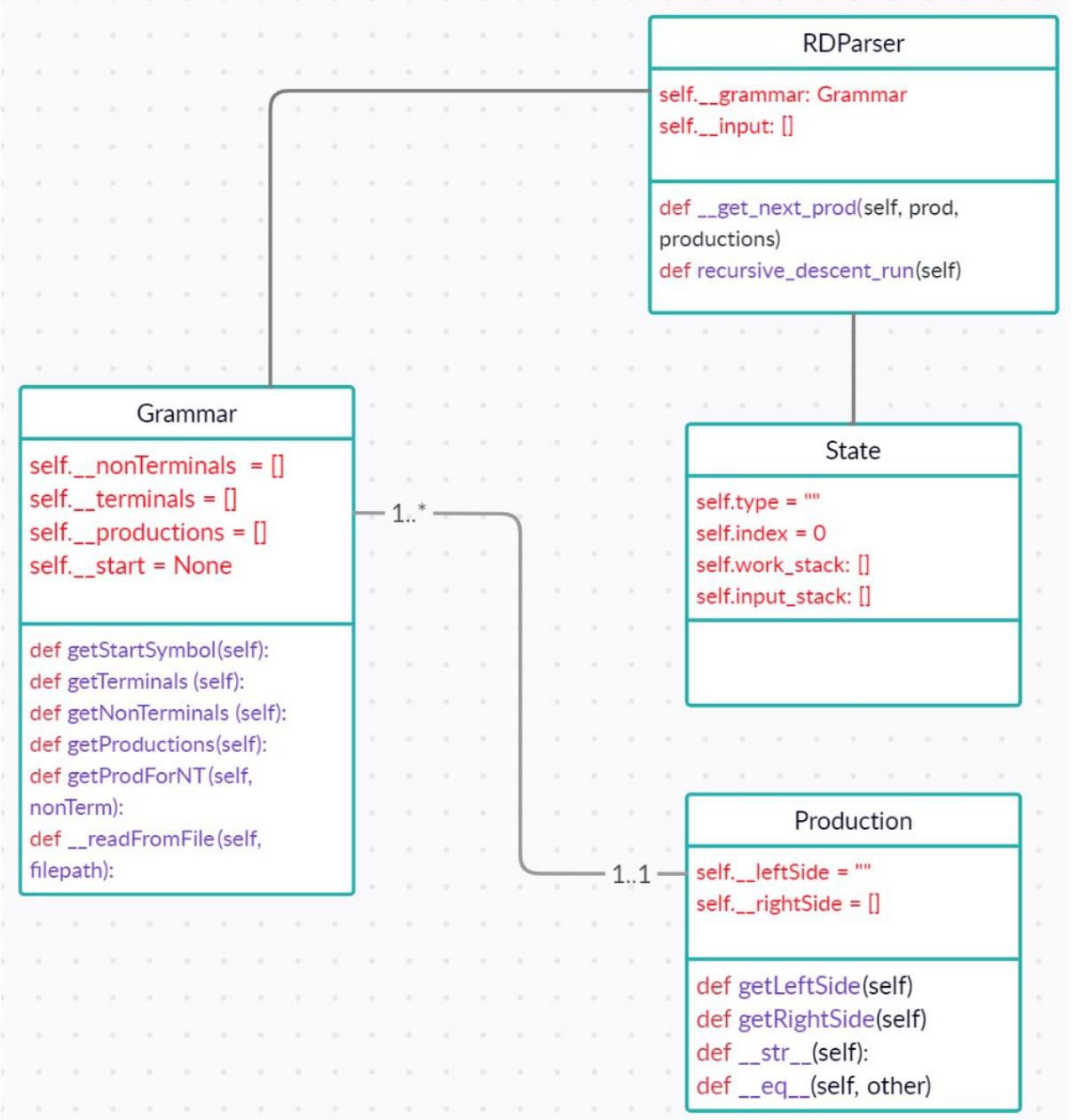
Miclea George
Mihalcea Leonard

Documentation - Lab 5-6

Link Github: https://github.com/Geromic/flcd/tree/master/lab5



Miclea George
Mihalcea Leonard

Input: g1.txt (https://github.com/Geromic/flcd/blob/master/resources/grammars/g1.txt)

Structure:

-1st line: space separated non-terminals (N)

-2nd line: space separated terminals (T)

-3rd line: start (S)

-next N lines: productions

```
S

        a b c

        S

        S ->
        aSbS|aS|c
```

Grammar

This class has multiple roles, firstly, it reads the grammar from the g1.txt file. It stores the non-terminals, the terminals, the start and the productions.

State

Stores the overall state of the program. We have 4 actions we can take at a given moment. NORMAL, ERROR, BACK, FINAL.

NORMAL: everything is normal, keep going
ERROR: something went wrong
BACK: we have to backtrack
FINAL: job's done

Production

Stores a production, its left side and right side. Also has getters for them.

Miclea George
Mihalcea Leonard

RDParser

Where the magic happens.
Initializes the state.
It has 2 main branches, namely the “continue” branch and the “backtrack” branch.

In the “continue” branch, it checks if it has reached the end, if so, sets the state type to FINAL. If not, if the input stack is emtpy, it sets the state to BACK.
If it’s not FINAL or BACK, we check if the first element in the input stack is in the non terminals list. If it is, we append to the work stack a pair made from the left side of the first production and the right side of the first production and we modify the input stack accordingly.

In the “backtrack” branch, we check if we have an error and if not, we undo our last decision and check the next production.

Miclea George
Mihalcea Leonard

Examples:

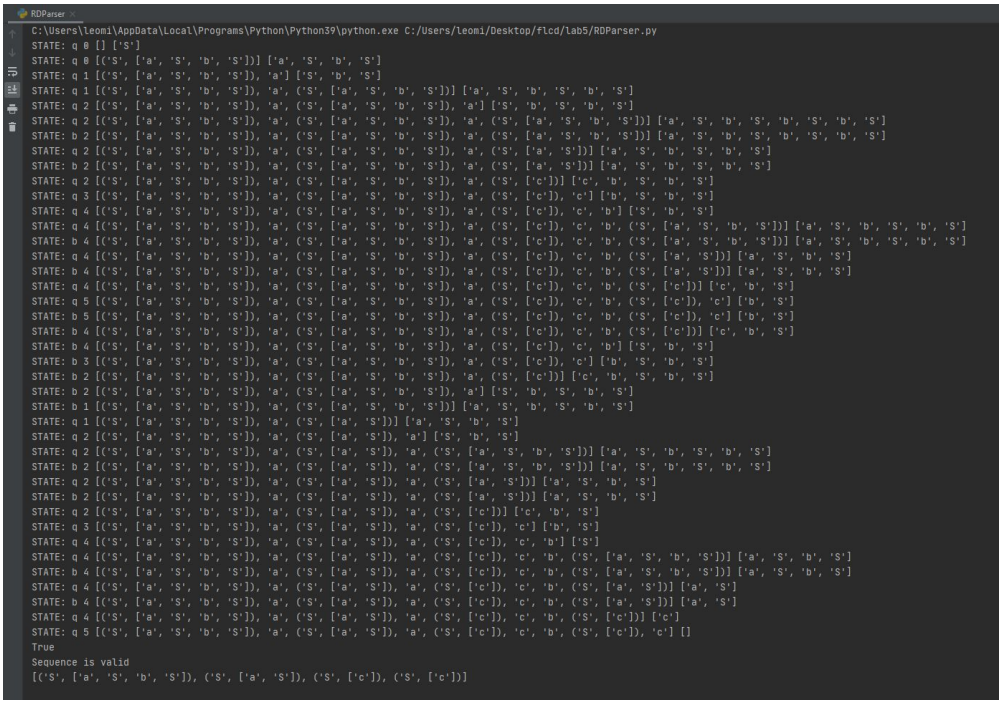
Grammar:

```
S
a b c
S
S -> aSbS|aS|c
```

“Happy” case:

Input: `"aacbc"`

Output



STATE: q 0 [] ['S']
STATE: q 0 [('S', ['a', 'S', 'b', 'S'])] ['a', 'S', 'b', 'S']
STATE: q 1 [('S', ['a', 'S', 'b', 'S']), 'a'] ['S', 'b', 'S']
STATE: q 1 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S', 'b', 'S'])] ['a', 'S', 'b', 'S', 'b', 'S']
STATE: q 2 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S', 'b', 'S']), 'a'] ['S', 'b', 'S', 'b', 'S']
STATE: q 2 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S', 'b', 'S'])] ['a', 'S', 'b', 'S', 'b', 'S', 'b', 'S']
STATE: b 2 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S', 'b', 'S'])] ['a', 'S', 'b', 'S', 'b', 'S', 'b', 'S']
STATE: q 2 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S']), 'a', ('S', ['a', 'S'])] ['a', 'S', 'b', 'S', 'b', 'S']
STATE: b 2 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S']), 'a', ('S', ['a', 'S'])] ['a', 'S', 'b', 'S', 'b', 'S']
STATE: q 2 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S']), 'a', ('S', ['c'])] ['c', 'b', 'S', 'b', 'S']
STATE: q 3 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S']), 'a', ('S', ['c']), 'c'] ['b', 'S', 'b', 'S']
STATE: q 4 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S']), 'a', ('S', ['c']), 'c', 'b'] ['S', 'b', 'S']

STATE: q 4 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['c']), 'c', 'b', ('S', ['a', 'S', 'b', 'S'])] ['a', 'S', 'b', 'S', 'b', 'S']
STATE: b 4 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['c']), 'c', 'b', ('S', ['a', 'S', 'b', 'S'])] ['a', 'S', 'b', 'S', 'b', 'S']
STATE: q 4 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['c']), 'c', 'b', ('S', ['a', 'S'])] ['a', 'S', 'b', 'S']
STATE: b 4 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['c']), 'c', 'b', ('S', ['a', 'S'])] ['a', 'S', 'b', 'S']
STATE: q 4 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['c']), 'c', 'b', ('S', ['c'])] ['c', 'b', 'S']
STATE: q 5 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['c']), 'c', 'b', ('S', ['c']), 'c'] ['b', 'S']
STATE: b 5 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['c']), 'c', 'b', ('S', ['c']), 'c'] ['b', 'S']
STATE: b 4 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['c']), 'c', 'b', ('S', ['c'])] ['c', 'b', 'S']
STATE: b 4 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['c']), 'c', 'b'] ['S', 'b', 'S']
STATE: b 3 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['c']), 'c'] ['b', 'S', 'b', 'S']
STATE: b 2 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['c'])] ['c', 'b', 'S', 'b', 'S']
STATE: b 2 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S', 'b', 'S']), 'a'] ['S', 'b', 'S', 'b', 'S']
STATE: b 1 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S', 'b', 'S'])] ['a', 'S', 'b', 'S', 'b', 'S']
STATE: q 1 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S'])] ['a', 'S', 'b', 'S']
STATE: q 2 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S']), 'a'] ['S', 'b', 'S']
STATE: q 2 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S']), 'a', ('S', ['a', 'S', 'b', 'S'])] ['a', 'S', 'b', 'S', 'b', 'S']
STATE: b 2 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S']), 'a', ('S', ['a', 'S', 'b', 'S'])] ['a', 'S', 'b', 'S', 'b', 'S']
STATE: q 2 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S']), 'a', ('S', ['a', 'S'])] ['a', 'S', 'b', 'S']
STATE: b 2 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S']), 'a', ('S', ['a', 'S'])] ['a', 'S', 'b', 'S']
STATE: q 2 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S']), 'a', ('S', ['c'])] ['c', 'b', 'S']
STATE: q 3 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S']), 'a', ('S', ['c']), 'c'] ['b', 'S']
STATE: q 4 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S']), 'a', ('S', ['c']), 'c', 'b'] ['S']
STATE: q 4 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S']), 'a', ('S', ['c']), 'c', 'b', ('S', ['a', 'S', 'b', 'S'])] ['a', 'S', 'b', 'S']
STATE: b 4 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S']), 'a', ('S', ['c']), 'c', 'b', ('S', ['a', 'S', 'b', 'S'])] ['a', 'S', 'b', 'S']
STATE: q 4 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S']), 'a', ('S', ['c']), 'c', 'b', ('S', ['a', 'S'])] ['a', 'S']
STATE: b 4 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S']), 'a', ('S', ['c']), 'c', 'b', ('S', ['a', 'S'])] ['a', 'S']
STATE: q 4 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S']), 'a', ('S', ['c']), 'c', 'b', ('S', ['c'])] ['c']
STATE: q 5 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S']), 'a', ('S', ['c']), 'c', 'b', ('S', ['c']), 'c'] []
True
Sequence is valid
[('S', ['a', 'S', 'b', 'S']), ('S', ['a', 'S']), ('S', ['c']), ('S', ['c'])]

Process finished with exit code 0

Miclea George
Mihalcea Leonard

Worst-Case scenario:

Input: `"abca"`

Output:

```
STATE: q 0 [] ['S']
STATE: q 0 [('S', ['a', 'S', 'b', 'S'])] ['a', 'S', 'b', 'S']
STATE: q 1 [('S', ['a', 'S', 'b', 'S']), 'a'] ['S', 'b', 'S']
STATE: q 1 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S', 'b', 'S'])] ['a', 'S', 'b', 'S', 'b', 'S']
STATE: b 1 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S', 'b', 'S'])] ['a', 'S', 'b', 'S', 'b', 'S']
STATE: q 1 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S'])] ['a', 'S', 'b', 'S']
STATE: b 1 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S'])] ['a', 'S', 'b', 'S']
STATE: q 1 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['c'])] ['c', 'b', 'S']
STATE: b 1 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['c'])] ['c', 'b', 'S']
STATE: b 1 [('S', ['a', 'S', 'b', 'S']), 'a'] ['S', 'b', 'S']
STATE: b 0 [('S', ['a', 'S', 'b', 'S'])] ['a', 'S', 'b', 'S']
STATE: q 0 [('S', ['a', 'S'])] ['a', 'S']
STATE: q 1 [('S', ['a', 'S']), 'a'] ['S']
STATE: q 1 [('S', ['a', 'S']), 'a', ('S', ['a', 'S', 'b', 'S'])] ['a', 'S', 'b', 'S']
STATE: b 1 [('S', ['a', 'S']), 'a', ('S', ['a', 'S', 'b', 'S'])] ['a', 'S', 'b', 'S']
STATE: q 1 [('S', ['a', 'S']), 'a', ('S', ['a', 'S'])] ['a', 'S']
STATE: b 1 [('S', ['a', 'S']), 'a', ('S', ['a', 'S'])] ['a', 'S']
STATE: q 1 [('S', ['a', 'S']), 'a', ('S', ['c'])] ['c']
STATE: b 1 [('S', ['a', 'S']), 'a', ('S', ['c'])] ['c']
STATE: b 1 [('S', ['a', 'S']), 'a'] ['S']
STATE: b 0 [('S', ['a', 'S'])] ['a', 'S']
STATE: q 0 [('S', ['c'])] ['c']
STATE: b 0 [('S', ['c'])] ['c']
False
Sequence is not valid
```

STATE: q 0 [] ['S']
STATE: q 0 [('S', ['a', 'S', 'b', 'S'])] ['a', 'S', 'b', 'S']
STATE: q 1 [('S', ['a', 'S', 'b', 'S']), 'a'] ['S', 'b', 'S']
STATE: q 1 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S', 'b', 'S'])] ['a', 'S', 'b', 'S', 'b', 'S']
STATE: b 1 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S', 'b', 'S'])] ['a', 'S', 'b', 'S', 'b', 'S']
STATE: q 1 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S'])] ['a', 'S', 'b', 'S']
STATE: b 1 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['a', 'S'])] ['a', 'S', 'b', 'S']
STATE: q 1 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['c'])] ['c', 'b', 'S']
STATE: b 1 [('S', ['a', 'S', 'b', 'S']), 'a', ('S', ['c'])] ['c', 'b', 'S']
STATE: b 1 [('S', ['a', 'S', 'b', 'S']), 'a'] ['S', 'b', 'S']
STATE: b 0 [('S', ['a', 'S', 'b', 'S'])] ['a', 'S', 'b', 'S']
STATE: q 0 [('S', ['a', 'S'])] ['a', 'S']
STATE: q 1 [('S', ['a', 'S']), 'a'] ['S']
STATE: q 1 [('S', ['a', 'S']), 'a', ('S', ['a', 'S', 'b', 'S'])] ['a', 'S', 'b', 'S']
STATE: b 1 [('S', ['a', 'S']), 'a', ('S', ['a', 'S', 'b', 'S'])] ['a', 'S', 'b', 'S']
STATE: q 1 [('S', ['a', 'S']), 'a', ('S', ['a', 'S'])] ['a', 'S']
STATE: b 1 [('S', ['a', 'S']), 'a', ('S', ['a', 'S'])] ['a', 'S']
STATE: q 1 [('S', ['a', 'S']), 'a', ('S', ['c'])] ['c']
STATE: b 1 [('S', ['a', 'S']), 'a', ('S', ['c'])] ['c']
STATE: b 1 [('S', ['a', 'S']), 'a'] ['S']
STATE: b 0 [('S', ['a', 'S'])] ['a', 'S']
STATE: q 0 [('S', ['c'])] ['c']
STATE: b 0 [('S', ['c'])] ['c']
False
Sequence is not valid