German University in Cairo
Media Engineering and Technology
Assoc. Prof. Haythem Ismail

Compilers Lab, Spring term 2019

Task 4

Elimination of left recursion & left factoring

Please read the following instructions carefully:

- Read Rules & regulations first
- It is **YOUR responsibility** to ensure that you have:
  - Submitted before the deadline.
  - Submitted the correct file(s).
  - Submitted the correct file(s) names.
  - Submitted correct logic of the task as it will be tested both publicly & privately.
  - Submitted your code in the format XX_XXXX_lab_4.zip where XX_XXXX is your ID for example 34_8000_lab_4.zip if your ID is 3 digits, append a zero to the left to be 34_0800_lab_4.zip to the correct google form link https://goo.gl/forms/rQz9Mz26NiBB73M72.

  .
- Good luck! =D

1 Elimination of left recursion

In this part, you are required to implement the elimination of left recursion for any given grammar. **Following the exact output file name for each one** & with the following format, you must follow the sample file on MET as well.

Follow the exact file name “task_4_1.py” & output file name “task_4_1_result.txt”.

For example, the input file will contain the grammar as follows:
(Assume that grammars given have no cycles & no epsilon productions)
(All tokens must be separated by space characters)
(Epsilon will be represented with the word "epsilon")

Listing 1: Format

```
Line #1 non terminal colon set of production rules.
e.g.: A : A c | A a d | b d
...
```

```
S : A a | b
A : S c | b d
```

The output would be:

```
S : A a | b
A : b c A' | b d A'
A' : a c A' | epsilon
```

2 Elimination of left factoring

In this part, you are required to implement the elimination left factoring for any given grammar. **Following the exact output file name for each one** & with the following format, you must follow the sample file on MET as well.

Follow the exact file name “task_4_2.py” & output file name “task_4_2_result.txt”.

For example, the input file will contain the grammar as follows:
(All tokens must be separated by space characters)
(Epsilon will be represented with the word "epsilon")

Listing 2: Format

```
Line #1 non terminal colon set of production rules.
e.g.: A : A c | A a d | b d
...
```

```
S : S + S | S S | ( S ) | S * | a
A : A c | A a d | b d | epsilon
```

The output would be:

```
S : S S' | ( S ) | a
S' : + S | S | *
A : A A' | b d | epsilon
A' : c | a d
```