Computer Science 312
Principles of Programming Languages
Spring 2018
Assignment 2 – Derive

Due: 11:59 p.m., Tuesday, 2/20

Description

For this project, you are to write a program in Python3 that reads in a grammar file and generates all terminal strings up to a specified length. In the file, the grammar rules will appear one rule per line, with each symbol separated from the next symbol by white space. For example:

```
N = D
N = N D
D = 0
D = 1
```

With a specified length of 2, your program should generate:

```
0
1
0 0
0 1
1 0
1 1
```

Specifications

Your program should begin by reading the file containing the productions of a grammar. The following specifications describe the format of the file:

- only one production appears per line
- the start symbol is the *lhs* of the first production, in the above example, **`N`**
- the set of nonterminal symbols is precisely the set of symbols that appear at least once as a *lhs* symbol; all the remaining symbols are terminals
- all productions for a given nonterminal are grouped together
- the metasymbol for "derives" or "can be replaced by" is always the second symbol and is the same for all productions (but not all grammars)
- all symbols, including metasymbols (e.g., **`=`**), nonterminals (e.g., **`N D`**), and terminals (e.g., **`9`**), must be separated from each other by one or more spaces

The output generated by your program should adhere to the following guidelines:

- output should be limited to generated strings only
- each generated string should appear on a separate line
- generated strings may appear in any order
- in any generated string, all characters should be delimited by spaces

Do not make any of the following assumptions:

- the metasymbol for "derives" is "=" or one character in length
- any symbol (metasymbol, terminal, or nonterminal) is restricted to one character
- a grammar test file contains errors
- a grammar will cause the algorithm to cycle

Algorithm

Your program should be written using the following algorithm:

```
read length N from the command line argument
read the grammar file and store all productions

push the start symbol onto the worklist
while the worklist is not empty:
      get and delete one potential sentence s from the worklist

      if | s | > N, continue

      if s has no nonterminals, print s and continue

      choose the leftmost nonterminal NT
      for all productions NT -> rhs:
            replace NT in s with rhs and store in tmp
            push tmp onto the worklist
```

Python Notes

Your program must be written in Python3 and must compile and run on the department's machines. Additionally, the following guidelines should be followed:

- store the candidate terminal strings (worklist) as a simple list
  - reading a line from a file, one at a time:

    ```
    for line in open(filename, "r"):
        ...
    ```
  - splitting a string into an array with whitespace delimiters:

    ```
    import string
    ...
    list = astring.split( )
    ```
  - adding a new value to a list: **`array.append(stringval)`**

- store each production in a dictionary with the *lhs* nonterminal as the key and a list of strings as the value
  - checking a dictionary for a given key: **`if dict.has_key(key)`**
  - creating a list and assigning it to a dictionary: **`dict[key] = [ ]`**

Invocation

The program should be invoked with the following command:

```
python3 derive.py [-llength] grammarfile
```

where

- **`-l`** is an optional parameter which gives the maximum string length (default: 3); there should be no space between the **`-l`** and the number, e.g., **`-l3`**
- **`grammarfile`** is the name of the file containing the grammar

Submission

Your python source file, named **`derive.py`**, should be submitted through the assignment link in Blackboard. You may also be required to submit a hardcopy of your program.fffkkkhggg