Derive

Due: 11:59 PM, Feb 1
Submit Project: derive
Source Files: derive.py
Test Files: Binary.txt Ambig.txt

You are to read in a grammar and generate all terminal strings up to a specified length. The grammar rules will appear one rule per line, with each symbol separated from the next symbol by white space.

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

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

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

Requirements

1. Output:
    a. The order of the lines shown is irrelevant.
    b. There should be no other output, especially on the lines with the digit sequences on them.
    c. In printing a digit sequence, there should be a space between successive digits.
2. Input:
    a. Productions are required to appear one per line.
    b. All the productions for a given nonterminal will be grouped together.
    c. The start symbol is the *lhs* of the first production, in this case N.
    d. The meta symbol for "derives", or "can be replaced by" is always the second symbol and is the same for all productions (but **not** all grammars).
    e. All symbols including metasymbols (ex. =), nonterminals (ex: N D), and terminals (ex: 9) must be separated from each other by 1 or more spaces.
    f. 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.
3. Hints:
    a. Store the productions as a dictionary with the key being the lhs nonterminal and the value being a list of strings.
    b. Store the candidate terminal strings (worklist) as a simple numeric list.

Algorithm

```
Read the length N from the command argument (or from input).
```

```
Read and store all the 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 the | 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; call it tmp.
                Store tmp on worklist.
```

Bad Assumptions

1. The metasymbol for "derives" is "=" or one character in length.
2. Any symbol is restricted to being one character.
3. There is an error in a grammar test file.
4. A grammar will cause the above algorithm to cycle.

Python Idioms

- Checking if a Dictionary has a given key: `if dict.has_key(key)`
- Creating a list and assigning it to a dictionary: `dict[key] = [ ]`
- Adding a new value to a list: `array.append(stringval)`
- Splitting a string into an array using whitespace:

```
import string
...
list = astring.split( )
```

- Reading a line from a file, one at a time:

```
for line in open(filename, "r"):
    ...
```

Invocation

```
python  derive.py  [-l length]  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.
- Windows users may prompt and read each of the above instead.
  the command line `python derive.py -l5 Binary.txt` would appear as the prompts:

```
Length? 5
Filename: Binary.txt
```

Python3

If you are using Python3, then the first line of each Python file must contain:

```
#! /usr/bin/python3
```

Submision

This project should be submitted through the assignment link in the Blackboard. If you have more than one files to submit, you are required to put all files into a single tar file for the submission.