

## CS240 Program 2

### Random Sentence Generator

In the past decade or so, computers have revolutionized student life. In addition to providing no end of entertainment and distractions, computers also have also facilitated all sorts of student work from English papers to calculus. One important area of student labor that has been painfully neglected is the task of filling up space in papers, Ph.D. dissertations, extension requests, etc. with important sounding and somewhat grammatically correct random sequences. An area that has been neglected that is, until now.

The Random Sentence Generator is a handy and marvelous piece of technology to create random sentences from a context-free grammar. You can choose from the variety of grammars that come with the assignment free of charge, and generate James Bond movie plots, sound bites for your future in politics, Dear John and college rejection letters, etc. Or, create your own grammar for any application you want!

#### The Grammar

Grammar rules follow a simple format, exemplified below in the definition of “<verb>”:

```
{ <verb> sigh <adverb> ; portend like <object> ; die <adverb> ; }
```

This rule has three productions. Each string in brackets is a non-terminal. A production can be empty (i.e. just consist of the terminating semi-colon). The entire definition is enclosed in curly braces ‘{’ ‘}’. Comments and other irrelevant text may be outside the curly braces and should be ignored. All the components of the input file—braces, words, and semi-colons—will be separated from each other by some sort of white space (spaces, tabs, newlines), so you will be able to use those as delimiters when parsing the grammar. You can discard the white-space delimiters since they are not important. No string will be larger than 128 characters long, so you have an upper bound on the buffer needed when reading a word.

Once you have read in the grammar, you will be able to produce random expansions from it. You will always begin with the single non-terminal `<start>`. For a nonterminal, consider its definition, which will contain a set of productions. Choose one of the productions at random. Take the words from the chosen production in sequence, (recursively) expanding any that are themselves non-terminals as you go. For example:

```
<start>
The <object> <verb> tonight.
The big yellow flowers <verb> tonight.
The big yellow flowers sigh <adverb> tonight.
The big yellow flowers sigh warily tonight.
```

Since we are choosing productions at random, doing the derivation a second time might produce a different result, and running the entire program again should also result in different patterns.

#### Choosing The Grammar File

Your program should take one argument, which is the name of the grammar file to read. Your program should create three random expansions from the grammar and exit.

**Getting Started**

The directory `cs240/assignments/rsg` contains a subdirectory of grammar files (files named with the extension `".g"`).