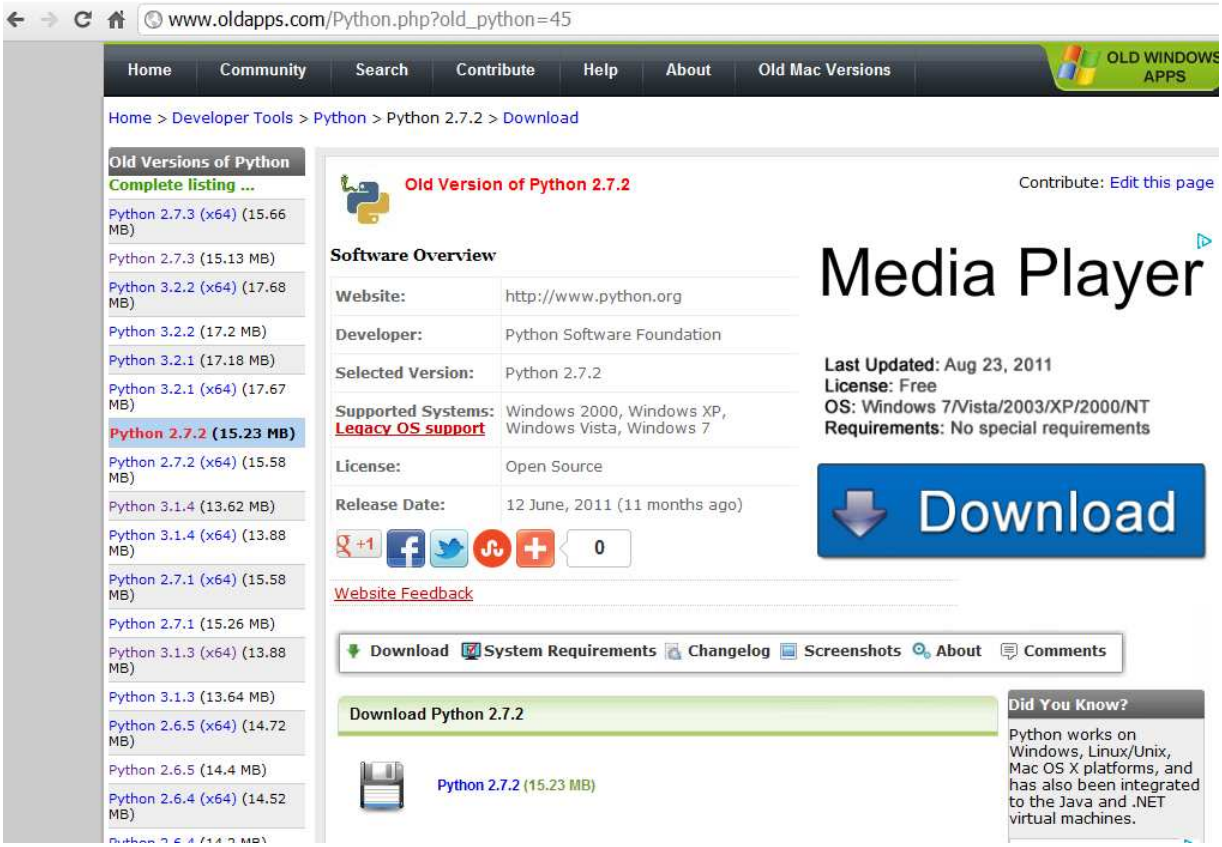
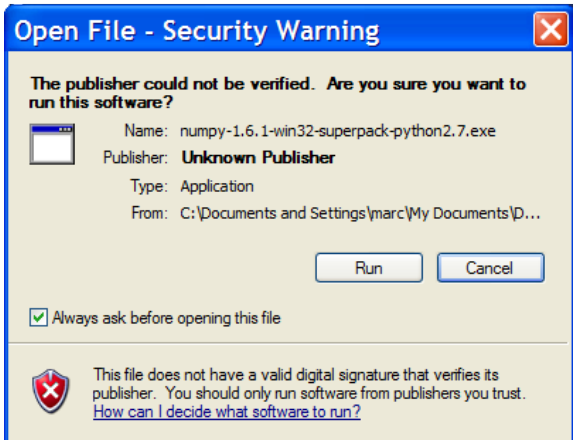
A Python program to calculate the Good-Turing frequencies

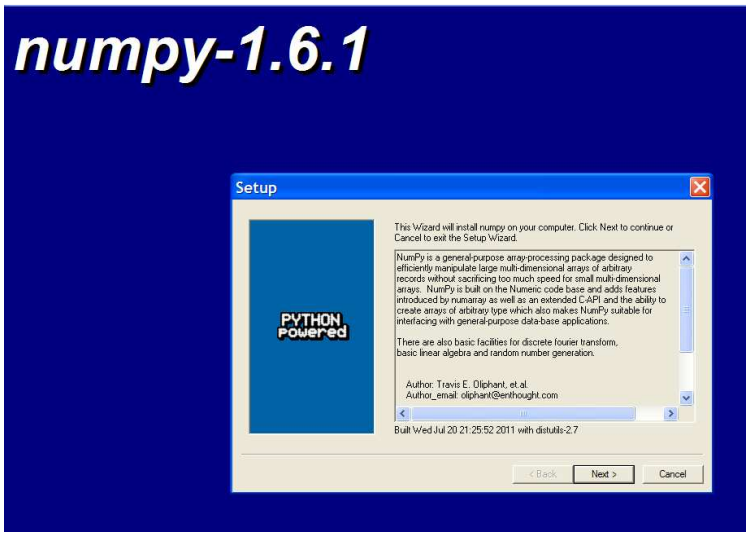
Python is a computer programming language. A complicating factor with Python is that there are many different versions, which are not 100% compatible. In particular, Python 3 is different from Python 2. Because our algorithm has been written in Python 2, we must install this version. Because this is not the most recent release, it is better not to go to the official Python site (where installing old releases is rather cumbersome), but to another site such as http://www.oldapps.com/ (use a google search ***download old releases python 2*** if you can't easily find what you are looking for). Let's install Python 2.7. If you're working under Windows, you need the 32 bit version.



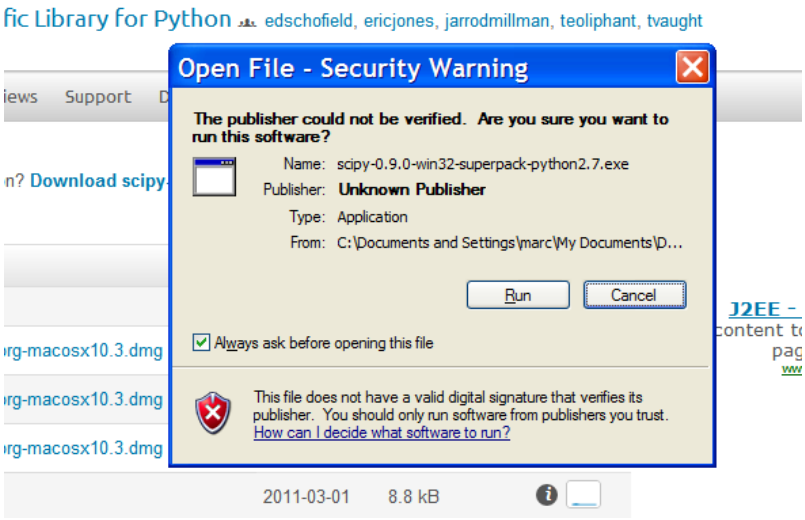
Follow the full installation instructions until you have Python 2.7 installed.

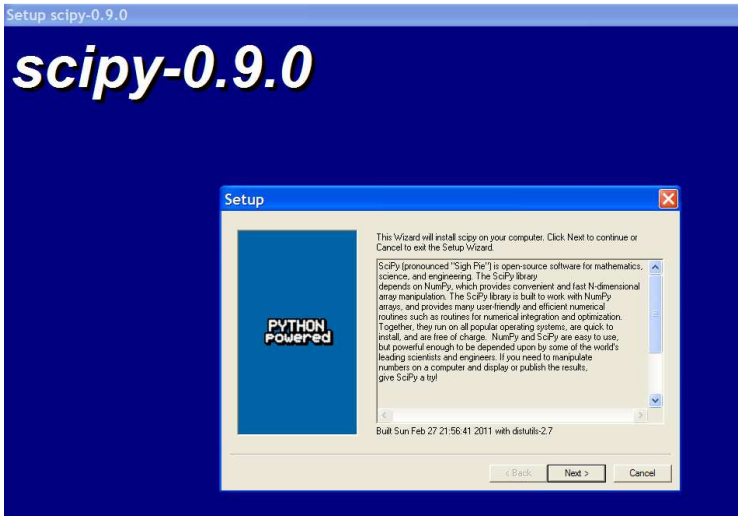
Because the Good-Turing algorithm works with mathematical libraries, you need to install two additional packages Numpy and Scipy. Be careful because these libraries are version dependent. So, make sure you use the correct version. For Python 2.7, this is currently at http://sourceforge.net/projects/numpy/files/NumPy/1.6.1/. The name of the file is *numpy-1.6.1-win32-superpack-python2.7.exe*



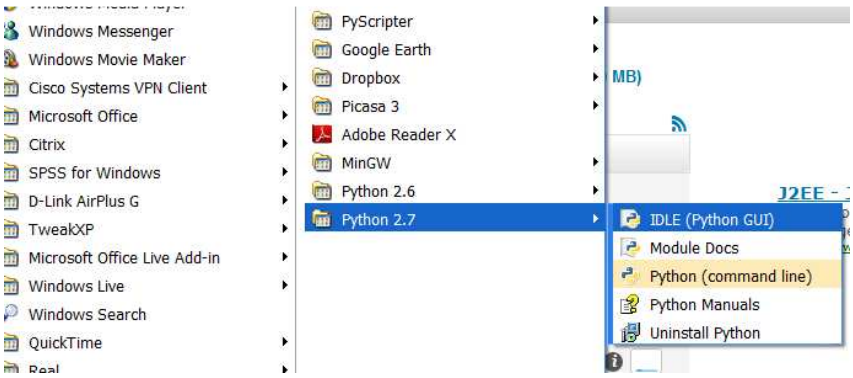


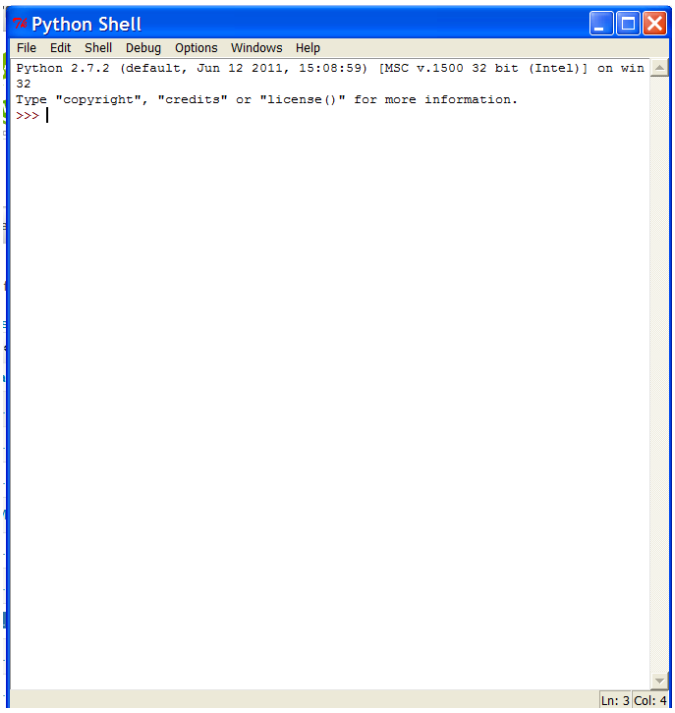
Finally, we need to install Scipy from (at the time of writing) http://sourceforge.net/projects/scipy/files/scipy/0.9.0/ (name of the file *scipy-0.9.0-win32-superpack-python2.7.exe*)



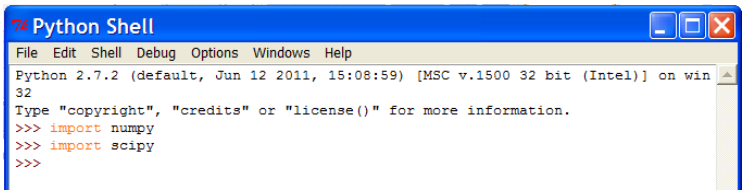


You can now open the Python shell using Programs under the Windows Start button.



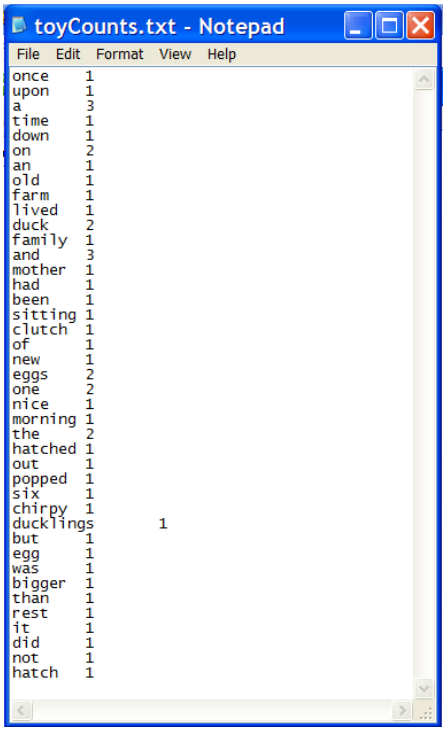


Write ***import numpy*** and enter; write ***import scipy*** and enter

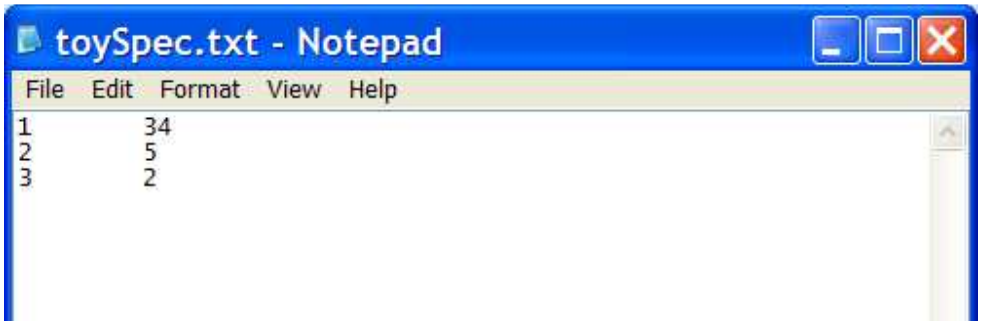


Now that you have python and the associated libraries installed, you can run the Good-Turing algorithm. To install the algorithm, save the file sgt.tar.gz or sgt.zip in the directory you want to. These are compressed files. The former is in a much-used format in the programming world, but which MS Windows does not recognize. If you have no special decompressor installed, you'll better work with the latter.

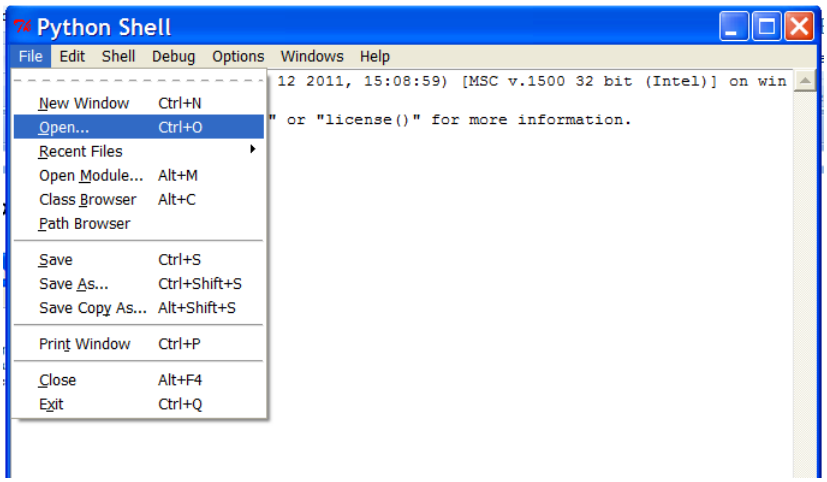
If everything went well, you should have four files in your directory: sgt.py, sgtInp.py (be careful: the middle letter is the capital I), tmpCounts.txt, and tmpSpec.txt. To these, we can add two more files related to the toy corpus we are working with in the article: toyCounts.txt and toySpec.txt. To create these, open Notepad or some other basic text processor (not MS Word!) and enter your data. Make sure there is a tab between the entries of the two columns. This is how the toyCounts.txt file looks like:

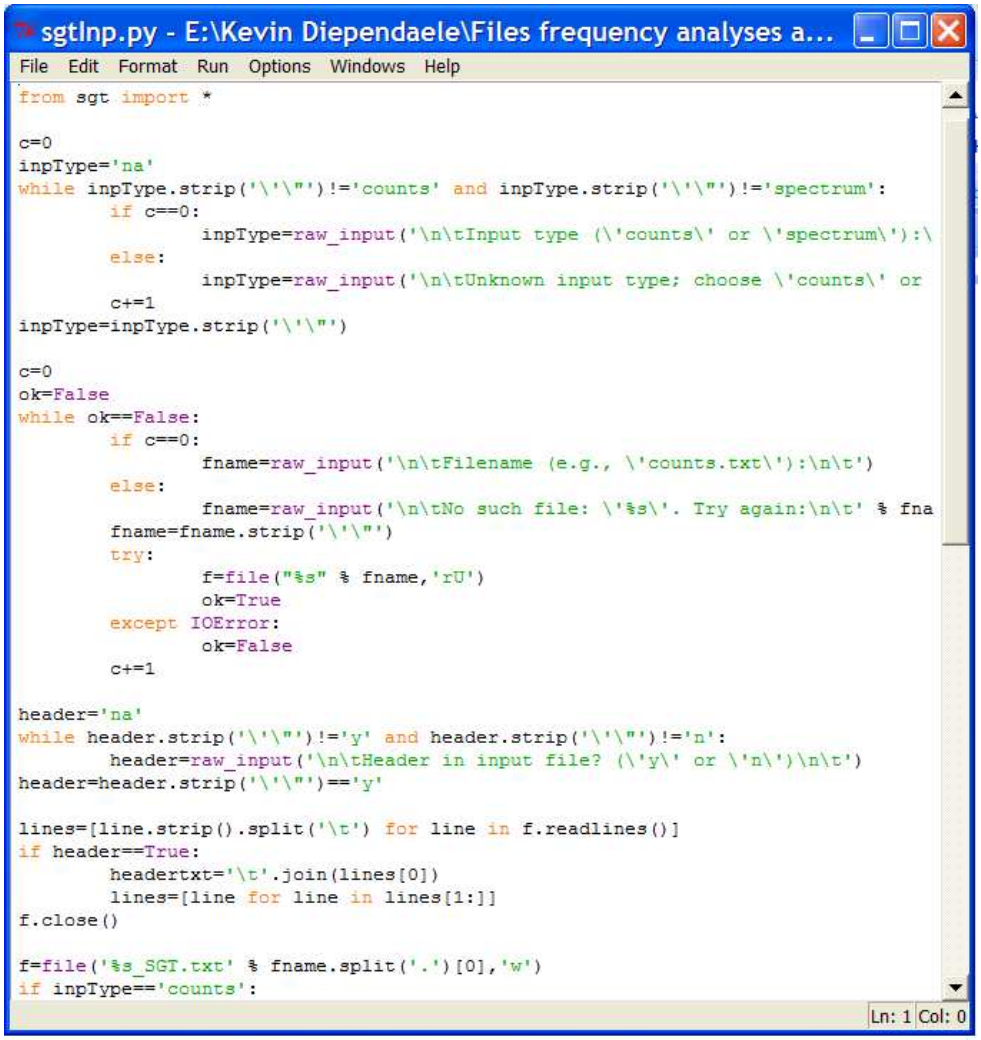


Save the file in the same directory as the file sgtInp.py is located in. Now make the toySpec.txt file. Again make sure that there is a tab between the entries on a line.

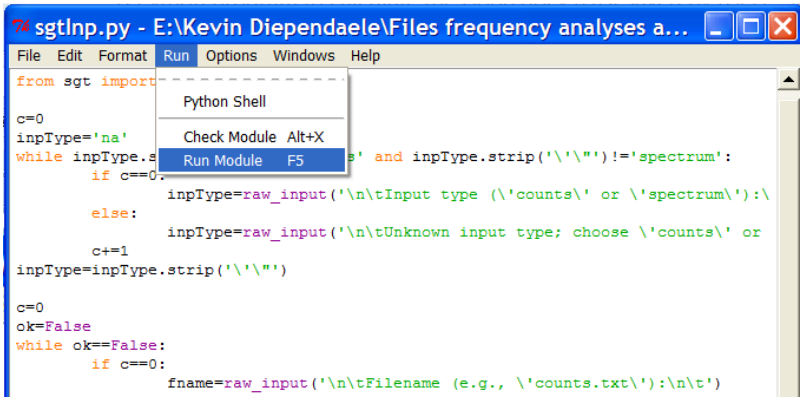


Go to your Python Shell and open the file ***sgtInp.py*** from the directory in which you saved it.

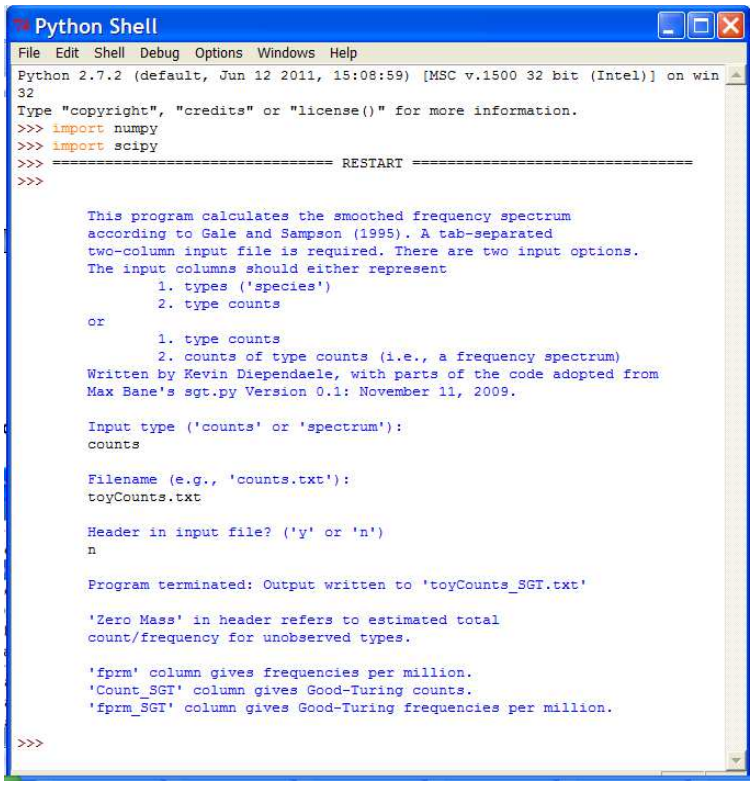




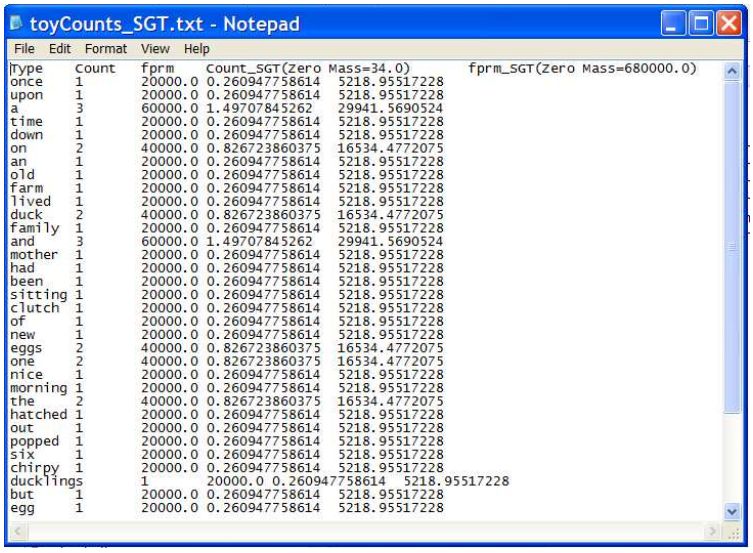
Click on Run:



Input the required data to get the outputfile:

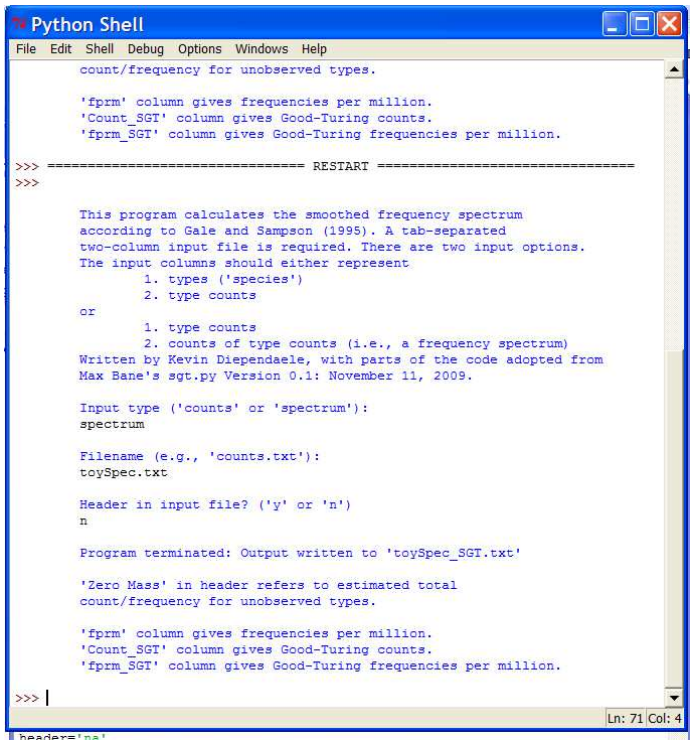


Open the file *toyCounts_SGT.txt* to see the output. In this file you get the output both as raw counts and as frequencies per million. So, the zero mass is 34/50 or 680,000 pm. The words observed once (or 20,000 pm) get a recalculated frequency of .26/50 or 5,219 pm. The words observed twice have a new frequency of .83/50 or 16,534 pm. Finally, the words observed three times have a new frequency of 1.50/50 or 29,942 pm.



Type	Count	fprm	Count_SGT(Zero Mass=34.0)	fprm_SGT(Zero Mass=680000.0)
once	1	20000.0	0.260947758614	5218.95517228
upon	1	20000.0	0.260947758614	5218.95517228
a	3	60000.0	1.49707845262	29941.5690524
time	1	20000.0	0.260947758614	5218.95517228
down	1	20000.0	0.260947758614	5218.95517228
on	2	40000.0	0.826723860375	16534.4772075
an	1	20000.0	0.260947758614	5218.95517228
old	1	20000.0	0.260947758614	5218.95517228
farm	1	20000.0	0.260947758614	5218.95517228
lived	1	20000.0	0.260947758614	5218.95517228
duck	2	40000.0	0.826723860375	16534.4772075
family	1	20000.0	0.260947758614	5218.95517228
and	3	60000.0	1.49707845262	29941.5690524
mother	1	20000.0	0.260947758614	5218.95517228
had	1	20000.0	0.260947758614	5218.95517228
been	1	20000.0	0.260947758614	5218.95517228
sitting	1	20000.0	0.260947758614	5218.95517228
clutch	1	20000.0	0.260947758614	5218.95517228
of	1	20000.0	0.260947758614	5218.95517228
new	1	20000.0	0.260947758614	5218.95517228
eggs	2	40000.0	0.826723860375	16534.4772075
one	2	40000.0	0.826723860375	16534.4772075
nice	1	20000.0	0.260947758614	5218.95517228
morning	1	20000.0	0.260947758614	5218.95517228
the	2	40000.0	0.826723860375	16534.4772075
hatched	1	20000.0	0.260947758614	5218.95517228
out	1	20000.0	0.260947758614	5218.95517228
popped	1	20000.0	0.260947758614	5218.95517228
six	1	20000.0	0.260947758614	5218.95517228
chirpy	1	20000.0	0.260947758614	5218.95517228
ducklings	1	20000.0	0.260947758614	5218.95517228
but	1	20000.0	0.260947758614	5218.95517228
egg	1	20000.0	0.260947758614	5218.95517228

You get the same information if you run the toySpec.txt file:



You find the information in the toySpec_SGT.txt file now arranged as a frequency spectrum (a distribution of the different frequency values).