Word Frequencies

Write a Python program named `freak.py` that takes one or more positional arguments of file names and counts the number of times each word occurs in the text. You will need to remove any non-letters from each word (e.g., punctuation so that "foo," becomes "foo") and count irrespective of case (so "foo," "Foo," and "FOO" are all the same word). Your program should also accept `-s|--sort` option that allows the user to choose the output to be sorted by "word" (alphabetically, this is the default) or "frequency" (numerically in ascending order) as well as a `-m|--min` option that is an integer value indicating the minimum number of times a word must occur to be included in the output (default 0).

As you read each file into words, you may choose to use a regular expression to remove anything that is not a letter or number, e.g.:

```
>>> import re
>>> re.sub('[^a-zA-Z0-9]', '', 'f)0o_,b@a,>r!')
'f0obar'
```

I used a `defaultdict(int)` structure for my word counter, but you might also want to consider a `Counter`, both from the `collections` module.

The output should be formatted with:

```
print('{:20} {}'.format(word, count))
```

To find out how to sort a dicitionary by values rather than words, cf. https://github.com/hurwitzlab/biosys-analytics/blob/master/lectures/08-python-patterns/python-common-patterns.md#sort-a-dictionary-by-values.

You will be given bad input files, but you need not create a specific error message but only throw an error on bad files. For this exercise, I chose to use a `type=argparse.FileType('r', encoding='UTF-8')` for my `file` argument which causes `argparse` to throw the error rather than me checking the input. This is because I don't need to know the *name* of the file in my output, so I just rely on `argparse` to give me a list of filehandles to read!

Expected Behavior

```
$ ./freak.py
usage: freak.py [-h] [-s str] [-m int] FILE [FILE ...]
freak.py: error: the following arguments are required: FILE
$ ./freak.py -h
usage: freak.py [-h] [-s str] [-m int] FILE [FILE ...]

Print word frequencies
```

```
positional arguments:
  FILE                File input(s)

optional arguments:
  -h, --help          show this help message and exit
  -s str, --sort str  Sort by word or frequency (default: word)
  -m int, --min int   Minimum count (default: 0)
$ ./freak.py foo
usage: freak.py [-h] [-s str] [-m int] FILE [FILE ...]
freak.py: error: argument FILE: can't open 'foo': [Errno 2] No such file or directory: 'foo
$ ./freak.py -m 50 -s frequency data/const.txt
such                 52
congress             60
as                   64
have                 64
any                  79
state                79
for                  85
united               85
a                    97
by                   101
president            109
states               129
in                   147
or                   160
be                   179
to                   202
and                  264
shall                306
of                   495
the                  727
$ ./freak.py -m 50 -s word data/usdeclar.txt
and                  57
of                   80
the                  78
to                   65
```

Test Suite

A passing test suite looks like this:

```
$ make test
python3 -m pytest -v test.py
```

```
============================== test session starts ===============================
platform darwin -- Python 3.6.8, pytest-4.2.0, py-1.7.0, pluggy-0.8.1 -- /anaconda3/bin/pyth
cachedir: .pytest_cache
rootdir: /Users/kyclark/work/worked_examples/14-word-freak, inifile:
plugins: remotedata-0.3.1, openfiles-0.3.2, doctestplus-0.2.0, arraydiff-0.3
collected 26 items

test.py::test_usage PASSED                                                   [  3%]
test.py::test_bad_file PASSED                                                [  7%]
test.py::test_01 PASSED                                                      [ 11%]
test.py::test_02 PASSED                                                      [ 15%]
test.py::test_03 PASSED                                                      [ 19%]
test.py::test_04 PASSED                                                      [ 23%]
test.py::test_05 PASSED                                                      [ 26%]
test.py::test_06 PASSED                                                      [ 30%]
test.py::test_07 PASSED                                                      [ 34%]
test.py::test_08 PASSED                                                      [ 38%]
test.py::test_09 PASSED                                                      [ 42%]
test.py::test_10 PASSED                                                      [ 46%]
test.py::test_11 PASSED                                                      [ 50%]
test.py::test_12 PASSED                                                      [ 53%]
test.py::test_13 PASSED                                                      [ 57%]
test.py::test_14 PASSED                                                      [ 61%]
test.py::test_15 PASSED                                                      [ 65%]
test.py::test_16 PASSED                                                      [ 69%]
test.py::test_17 PASSED                                                      [ 73%]
test.py::test_18 PASSED                                                      [ 76%]
test.py::test_19 PASSED                                                      [ 80%]
test.py::test_20 PASSED                                                      [ 84%]
test.py::test_21 PASSED                                                      [ 88%]
test.py::test_22 PASSED                                                      [ 92%]
test.py::test_23 PASSED                                                      [ 96%]
test.py::test_24 PASSED                                                      [100%]

=========================== 26 passed in 1.55 seconds ============================
```