Python FASTA GC Segregator

Write a Python program called "gc.py" that takes

- One or more FASTA files as positional arguments
- An option `-o|--outdir` directory name to write the output (default "out")
- An option `-p|--pct_gc` integer value between 1-100 (inclusive) for the percent GC to decide high/low content (default 50)

Generate a "usage" if given no arguments or the `-h|--help` argument.

The program should iterate through the FASTA files, calculate the GC content of each sequence, and write the sequence to a "high" file if the percent GC is greater than or equal the `--pct_gc` argument or to the "low" file if it is lower than the argument. The name of the high/low files should be the basename of the input file plus "_[low/high]"; that is, if the input file is"foo.fa" then you should create "foo_high.fa" and "foo_low.fa" in the given `--outdir`. Note that if the output directory does not exist, you will need to create it.

If a given "file" argument is not a file, print ' "XXX" is not a file' to STDERR and continue processing.

Expected behavior

```
$ ./gc.py
usage: gc.py [-h] [-o DIR] [-p int] FASTA [FASTA ...]
gc.py: error: the following arguments are required: FASTA
$ ./gc.py -h
usage: gc.py [-h] [-o DIR] [-p int] FASTA [FASTA ...]

Segregate FASTA sequences by GC content

positional arguments:
  FASTA                 Input FASTA file(s)

optional arguments:
  -h, --help            show this help message and exit
  -o DIR, --outdir DIR  Output directory (default: out)
  -p int, --pct_gc int  Dividing line for percent GC (default: 50)
$ ./gc.py foo
"foo" is not a file
Done, wrote 0 sequences to out dir "out"
$ ./gc.py fasta/CAM_SMPL_GS108.fa
  1: CAM_SMPL_GS108.fa
Done, wrote 500 sequences to out dir "out"
$ ./gc.py -p 32 -o splits fasta/*
```

```
  1: CAM_SMPL_GS108.fa
  2: CAM_SMPL_GS112.fa
Done, wrote 1000 sequences to out dir "splits"
```

Hints

You will proabably want to use the following methods:

- os.makedirs: to create directories
- os.path.basename: to get the filename from a path
- os.path.join: join (in a OS-independent way) directory names and files
- os.path.splitext: to split "foo.fa" into "foo" and ".fa"
- Use the SeqIO module from Bio to `parse` and `write` sequences

Look back at the many examples of counting DNA to pick a method you like to count the number of Gs and Cs, then divide by the length of the sequence. Remember this will give you something like "0.43219" and you are comparing to an integer like "43," so do the proper math.

Test Suite

A passing test suite looks like the following:

```
$ 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/06-fasta-gc, inifile:
plugins: remotedata-0.3.1, openfiles-0.3.2, doctestplus-0.2.0, arraydiff-0.3
collected 5 items

test.py::test_usage PASSED                                               [ 20%]
test.py::test_bad_input PASSED                                           [ 40%]
test.py::test_good_input1 PASSED                                         [ 60%]
test.py::test_good_input2 PASSED                                         [ 80%]
test.py::test_good_input3 PASSED                                         [100%]

=========================== 5 passed in 1.87 seconds ===========================
```