Calculate the N50 score of An Assembly

Tania Chakraborty, Project, Biosystems Analytics

Genome assemblies are often given scores to denote how "complete" an assembly is, otherwise, it would be hard to understand the quality of the genome assemble. N50 score is one such measure, higher the N50 score, more complete the genome assembly is. It is calculated in the following way. First, all the contigs in the assembly are lined up in order of their sequence lengths. Then the contig sequences are counted, starting from the largest to smallest until the contig length reaches half of the length of the total assembly. The length of shortest contig in this list is the N50 value. Another way of thinking about it is that at least half of all the nucleotides in the assembly are in contigs of length N50 or higher. Write a Python program **n50.py** that calculates the N50 score of input sequences into an output text file.

The inputs for this program can be generated by the moog.py program. You can run **make fasta** to create fasta files of variable sizes in this directory. You can then use these files for testing your program. The testfile.fasta is a very small file and I included that while writing the program myself, and it will help in testing the program logically at every step since there are only 10 sequences. I have also included a version of the *Arabidopsis thaliana* genome, so that the program can be run on a real genome assembly. The fasta file for the genome is zipped, so you have to **gunzip** it first.

The parameters for your program are:

1. One or more positional FILE arguments.
2. An output text file where you output the N50 score (default "n50.txt").

Here is the usage your program should create for -h or --help:

```
$ ./n50.py -h
usage: n50.py [-h] [-o FILE] FILE
Calculate N50 score of sequences
positional arguments:
FILE                    Input FASTA file
optional arguments:
-h, --help              Show this help message and exit
-o FILE, --outfile FILE  Output filename (default: n50.txt)
```

When run with **file1.fa** it should print this to the stdout:

```
$ ./n50.py file1.fa
Processed the fasta file "file1.fa". Details are in "n50.txt".
```

The output file should contain the following information:

1. The name of the input file
2. Total size (in base pairs) of the sequences in the fasta file
3. Number of contigs
4. The N50 score.

This is what it should look like :

```
Filename: file1.fa
Total Size = 56,266
Total Contigs = 456
N50 Score: 148
```

All tests should pass:

```
============================== test session starts ==========================
.....
collected 9 items

test.py::test_exists PASSED                                  [ 11%]
test.py::test_usage PASSED                                   [ 22%]
test.py::test_bad_file PASSED                                [ 33%]
test.py::test_file1 PASSED                                   [ 44%]
test.py::test_file2 PASSED                                   [ 55%]
test.py::test_file3 PASSED                                   [ 66%]
test.py::test_file4 PASSED                                   [ 77%]
test.py::test_Atgenome PASSED                                [ 88%]
test.py::test_outfile_options PASSED                         [100%]

============================== 9 passed, 1 warning in 5.67s ===================
```