**Specific comments for this assignment:**

Students should not assume that all genomes will arrive in UPPER or lower case. Also, some microbial genomes have nucleotides other than (A,C,G,T) if a certain nucleotide during sequencing could not be determined (e.g., a genome might include an 'N' to represent a certain nucleotide is "unkNown").

Students need practice with formatting output, e.g., to two places after the decimal point: just show them how to do it, including how to format numbers in scientific notation.

Assuming independence of nucleotides, finding the likelihood of a particular motif ("word") in DNA involves multiplying small numbers together. This is an excellent time to introduce the notion of underflow, the log() and exp() mathematical functions, and the technique of taking the sum of logs (see LeBlanc and Dyer, 2007, p85-86 for a discussion -- Perl for Exploring DNA, Oxford University Press, 2007).

The starter kit includes a user-defined function, getDNA(), to open a file of DNA and return the contents as one DNA sequence (string). This is a good time to introduce a "walk-through" of someone else's code. Although most students will not have seen the conditional and looping control structures used in this function, nor even user-defined functions, it provides a valuable lesson and peek at things to come.

Do not underestimate the degree of help that students need in order to appreciate what is appropriate in a scientific write-up of their results. For example, undergraduates tend toward the first person (e.g., "The Python program I used ..." rather than "The Python program used ..."), so I recommend a rich discussion on what is appropriate. This will avoid some angst while you read/grade their reports. In addition, students need to be shown by example how to correctly format all Figures and Tables, including the correct use of legends for each figure and table. I typically encourage students to format their Tables of results in Excel (and this also may need some time and examples) and then include these nicely formatted Tables into their final report. Some faculty may feel that this is not an appropriate use of class time in a course focused on programming, however, my experience and student evaluations all point toward a sharpening of student writing, a most worthy side-effect.

Spend some time in class allowing students to access the National Center for Biotechnology Information (NCBI) website. I strongly recommend a tilt toward microbial genomes as these are considerably smaller than eukaryotic (e.g., human) genomes and the microbial annotation is easier to grasp. While life science students tend to gravitate toward human disease, the human microbiome projects now provide a vital connection between understanding microbial genomes and human health. <ftp://ftp.ncbi.nlm.nih.gov/genomes/HUMAN_MICROBIOM/Bacteria/> (to download DNA in FASTA formatted files) <http://www.ncbi.nlm.nih.gov/genome/browse/> (to explore microbial genomes)

**General comments for the entire set of DNA-focused programming assignments:**

(0) Regularly allocate time in your class sessions to bring in colleagues, in particular, a biologist who can talk (briefly) of the beauty of DNA. In this case, Chargaff was a pioneer in the early days of DNA research, albeit one who was not adequately recognized at the time so a guest speaker might be able to highlight this point.

(1) Give students a sense of the real demand for future scientists who can work in multidisciplinary groups on computationally-intensive problems. Don't be shy to share that the skills to be learned in this course can "get students a job" at the start of an exciting career. Many students in the life sciences have general plans to "be a doctor"; these assignments may be an undergraduate's first exposure to the excitement in and demand for computational scientists in regards to research and medicine.

(2) Use a "flipped classroom" where students watch lectures before/after class in order to maximize the amount of hands-on Python play. For example, (a) require students to watch lectures on biological topics (cf. Udacity's 'Tales of the Genome' MOOC) outside of class; and (b) leverage programming practice sites for student practice outside of class (cf. codeAcademy's Python course).