GS 373 Homework 9

Due June 7th before 1:30 PM on Canvas

- (100 points): 4 bioinformatics questions (80 points), 1 programming assignment (20 points).
- Bonus programming question (3 points)
- Submit answers to the bioinformatics questions in a Microsoft Word document or PDF via Canvas. Your answers do not need to contain the text of the questions, but they need to be clearly labeled (e.g., 1a., 3b., etc.)

Submit the programming assignment as a separate .py file onto Canvas. The script should be able to be directly run by Python.

Bioinformatics Questions (80 points)

1. (15 points) **What HTS assay or combination of assays** could you use to determine the following pieces of information**? Briefly explain the general experimental and computational workflow** needed to obtain the information from the assay(s):
   a. (5 points) The introns and exons of an organism
   b. (5 points) The regulatory targets of a transcription factor
   c. (5 points) The number of copies of each gene in a genome

2. (20 points) Sequencing technologies
   a. (10 points) **Describe the purpose of the emulsion in emulsion PCR**. What would we observe from a given bead if we omitted the emulsion step, but completed the rest of the workflow for randomly sheared genomic DNA?
   b. (10 points) Suppose you are doing a study using **paired-end sequencing of 500bp fragments with 150bp reads**. You obtain a number of reads that align to the reference genome at or near the coordinates below.
      i. What kind of **structural variant** is captured by these reads?
      ii. What information can be inferred about the **size and location of this variant**?
   Explain your reasoning.

fwd start position	fwd end position	rev start position	rev end position
chr1:1000	chr1:1150	chr1:1800	chr1:1950

3. (25 points) Hash-based alignment
   a. (10 points) If the average human genome contains 4 million SNPs uniformly randomly distributed amongst the 6 billion bases, **compute the likelihood that a random 100 base short read is not an exact match to the reference**. Ignore sequencing error and **show your work**.
   b. (5 points) Consider a read that aligns to the reference with 3 mismatches. If the read is split into 5 spaced seeds, **what are the minimum and maximum numbers of spaced seeds that map to the correct region of the reference genome**?
   c. (10 points) Describe **one advantage and one disadvantage** of using a hashing-based alignment compared to a Smith-Waterman local alignment.

4. (20 points) Variant calling
    a. (15 points) A genomic position is covered by five observed reads that contain A, T, T, G, T at that position with Phred scores of 25, 6, 6, 20, 15, respectively. **Which single base has the highest likelihood of being correc**t? Show your work.
    b. (5 points) Cancer cells often have structural defects like large duplications. If HTS was applied to cancer tissue with a whole chromosomal duplication, **what are the possible allele ratios of the reads you would expect to see at the loci in that chromosome**? Assume the effects of sequencing error and de novo mutations are minor.

Programming (20 points)

Write a program that computes the log likelihood of a given set of observed reads given each base (A, T, C, or G) in a single genomic position, based on the Phred scores of the nucleotides from the reads.

The probabilities that each read's call reflects a sequence error are shown below:

Read	Base call	Q Phred score	P(error) = $10^{-Q/10}$	P(correct)
1	A	10	0.1	0.9
2	A	10	0.1	0.9
3	G	30	0.001	0.999

In this example, the likelihood *p(A)* for if A is the true base at this position is:

*p(A)= P(Read 1 correct) * P(Read 2 correct) * P(Read 3 error)*

p(G) could be calculated in the same way for this example set of reads. Problem 4a above is another example calculation.

- **Your program should take an input file from the command line an input file that lists nucleotide calls and corresponding Phred scores from multiple aligned reads.** The input file ("input_phred.txt") is formatted such that each line corresponds to a read. The first element in the line is the nucleotide observed in the read, and the second element is its Phred score. The elements are separated by spaces.

- **The program should output the base and the $\log_{10}$(probability) of that base (separated by spaces).**

- **Because these computations require multiplying many small probabilities together, doing them by adding the logarithms of the probabilities will be easier and more accurate**. Remember that adding the logarithms of two numbers is equivalent to multiplying them; e.g.: $log_{10}(p_1 * p_2) = log_{10}(p_1) + log_{10}(p_2)$, and therefore, $p_1 * p_2 = 10^{log_{10}(p_1)+log_{10}(p_2)}$.

- The `math.log10(x)` function computes the base 10 logarithm of x. The `pow(10,y)` function can convert a $\log_{10}$ probability y back into the original probability space. Remember to place import math at the top of your program.

Example usage of your program:

```
python homework9_skh.py input_phred.txt
```

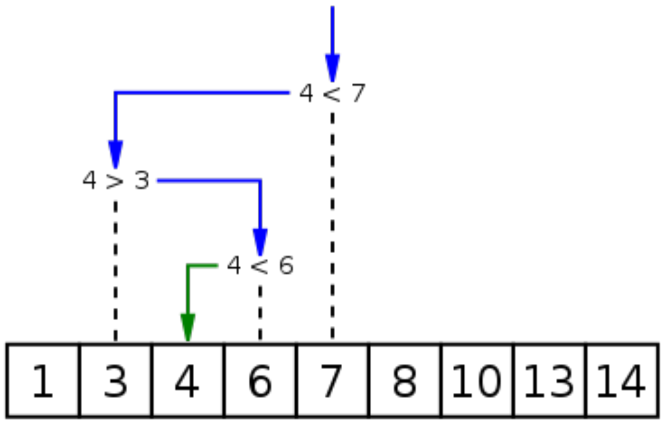
```
Nucleotide log10(probability)
A -31.7
C -24.5
T -2.8
G -7.7
```

Hints. you may want to …

- Define a function that converts Phred scores into the log probability that a base is correct.
- Read in your input file into a list of lists or dictionaries, where every element represents a read (line in the file). You can then loop through each read and use its information to calculate its contribution to the probabilities.
- Keep track of log probabilities in a dictionary, where the keys are bases and the value is the log probability calculated so far.

Optional programming question (3 bonus points):

Implement the recursive binary search algorithm. The binary search algorithm is for finding a value in a sorted list. If you were looking for a specific name in a phonebook, you might first open the book to the middle, then decide whether you are before or after your query, and then repeat your "splitting" on either the first or second half of the book depending on your result. See the figure below for an example.



- Your implementation should work on a large list of non-repeating, sorted integers (like binsearch_example.txt).
- The input will be this large list and a query number
- The output will be either "Not in the list" or the index where the query number exists in the list.

Example:

```
python homework9_skh_binary.py binsearch_example.txt 2033

Found on line 11


python homework9_skh_binary.py binsearch_example.txt 20

Not in the list
```