Bioinformatics computing
CSE40532/60532
Homework #1

Reading:

Read Chapter 1 of the text (first week)
Read sections 3.1-3.4; 3.6 of the text (8/24; week of 8/29)

Problems: (due 8/31/2017)

1. Read the CSE honor code statement and state in an accompanying writeup to this assignment you have read and agree to the terms (2 points)

2. Download the complete genomic sequence of bacteriophage lambda (accession NC_001416.1) and place it in your dropbox named "lambda.fasta" (1 points)

3. Write a small program to reverse complement the lambda genome. Save the new sequence with a FASTA header of ">reversed" and name the resulting file as "lambda.rev.fasta". Place this file and the source code in your dropbox, and instruct your instructor how to compile and run it in an accompanying write up called "report.txt" (5 points)

4. Write a small program that reports nucleotide frequencies and dinucleotide frequencies of lambda. Add instructions on how to run it and include your actual output in your write up (report.txt), and also place your code in the drop box (5 points)

5. Download the human mitochondrial genome (NC_012920). Place it in your dropbox as "human_mito.fasta." Download the Neanderthal hypervariable region (AF254446). Save it in your dropbox as "neander_sample.fasta." (1 points)

6. Write small programs to compute the log of the probability of the Neanderthal sequence under a multinomial model and a markov model of order 3, both of which are trained from the human mitochondrial sequence. Place the code in your dropbox and report the results in your write up. (8 points)

7. Generate a random sequence of 20,000 characters using a Markov model of order 3 trained on the human mitochondrial genome (you can start with your code from part #5). Place the code for this in your dropbox and tell your instructor how to run the program in your write up. (8 points)

   HINT: If you use C/C++, DNA sequences can be represented in binary (and therefore as integers) by mapping A to 00, C to 01, G to 10 and T to 11. As such AAAC would be 1 (00000001). Bitwise operators can then be used for computational efficiency.