Homework #2

Due March 3rd, 11:59pm

Each homework submission must include:

- An archive (.zip or .gz) file of the source code containing:
  - The makefile used to compile the code on Monsoon **(5pts)**
  - All .cpp and .h files **(5pts)**
- A full write-up (.pdf of .doc) file containing answers to homework's questions **(5pts)**, including the exact command line needed to execute every subproblem of the homework

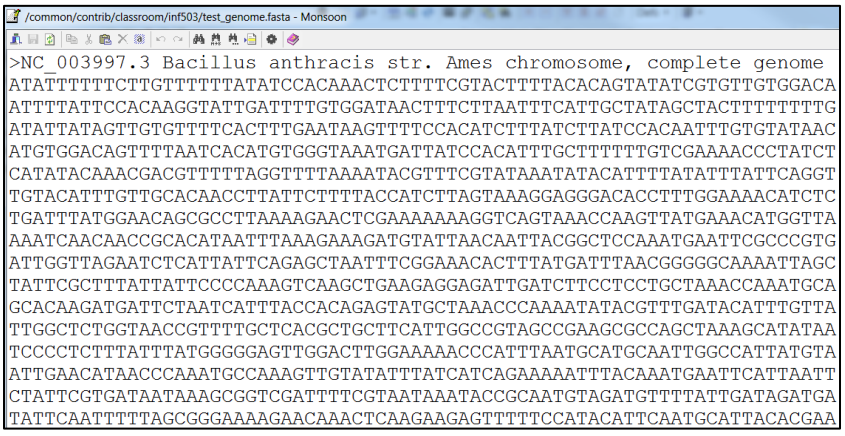
The source code must follow the following guidelines:

- No external libraries that implement data structures discussed in class are allowed, unless specifically stated as part of the problem definition. Standard input/output and utilities libraries (e.g. math.h) are ok.
- All external data sources (e.g. input data) must be passed in as a command line argument (no hardcoded paths within the source code **(5pts)**.
- Solutions to sub-problems must be executable separately from each other. For example, via a special flag passed as command line argument **(5pts)**

For this homework, you will continue to use the High Throughput Sequence reads dataset located on Monsoon: **/common/contrib/classroom/inf503/hw_dataset.fa**. Refer to Homework #1 assignment for description of the dataset.

You will also need to use the genome sequence for Bacillus anthracis bacterium located at: **/common/contrib/classroom/inf503/test_genome.fasta**

- This genome file contains a header (denoted by '>') followed by ~5.2 million characters of its genomic code (alphabet A, C, G, T)
- Please be aware that the genome is spread across multiple lines of the file (see insert)



Problem #1: Fun with linked lists

Create a class called ***FASTAreadset_LL***. The purpose of the class will be to contain a FASTA read set (similar to homework #1) and all of the functions needed to operate on this set. Use the **linked list** data-structure to store the genomic sequences of the read dataset. Use character arrays (char[ ]) to store the actual sequence fragment within each node of the linked list – there is no need to have a linked list of a linked list that stores one character at a time. For this assignment, you can completely disregard the headers of the sequence fragments (i.e. R0_0_1…) – the entire contents of the file go into the linked list. At minimum, the class must contain **(15pts):**

- A default constructor
- At least one custom constructor (e.g. one taking a file path or ifstream as input)
- A function to read the genome file
- A destructor
- A copy constructor

A. **(30 pts)** Read in the entire ~36 million read set (*query read set*) and store it in the FASTAreadset_LL class. Implement a search function which would take a sequence fragment (OK to assume that it will be exactly 50 characters long) and search for this fragment within the FASTAreadset_LL object. The search function should return the pointer to the node containing the match OR the NULL pointer value if a 'hit' was not found.

- Which of the following sequences were found in the read set?

  i. CTAGGTACATCCACACACAGCAGCGCATTATGTATTTATTGGATTTATTT
  ii. GCGCGATCAGCTTCGCGCGCACCGCGAGCGCCGATTGCACGAAATGGCGC
  iii. CGATGATCAGGGGCGTTGCGTAATAGAAACTGCGAAGCCGCTCTATCGCC
  iv. CGTTGGGAGTGCTTGGTTTAGCGCAAATGAGTTTTCGAGGCTATCAAAAA
  v. ACTGTAGAAGAAAAAAGTGAGGCTGCTCTTTTACAAGAAAAAGTNNNNNN

- Would sorting the linked list help speed up the search (on average and in the worst case)? Explain why or why not?

B. **(30 pts)** Read in the *Bacillus anthracis* genome into a character array (you will need to determine the exact size of the sequence). Iterate through all possible 50-character long fragments within the genome by shifting fragment start location by one character each time. Use these fragments to search within the FASTAreadset_LL object.

- How many 50 character fragments can you make from the *B. anthracis* genome?

- What is the overlap between the genome's 50-mers and the ~36 million fragments you've stored in the FASTAreadset_LL object? Please note that depending on the efficiency of your algorithm, this step may take a long time. First estimate the total time using 1,000, 10,000, and 100,000 queries – if total time estimate is greater than 24 CPU hours, provide estimate rather than exact number.

- You've iterated through all 50-mers found in the genome and used them to search within the *query read set*. Would it have been faster to flip the problem – i.e. store the genome's fragments in a data structure and iterate through the query read set? Explain why or why not.