

Homework #2

Due October 2nd, 11:59pm

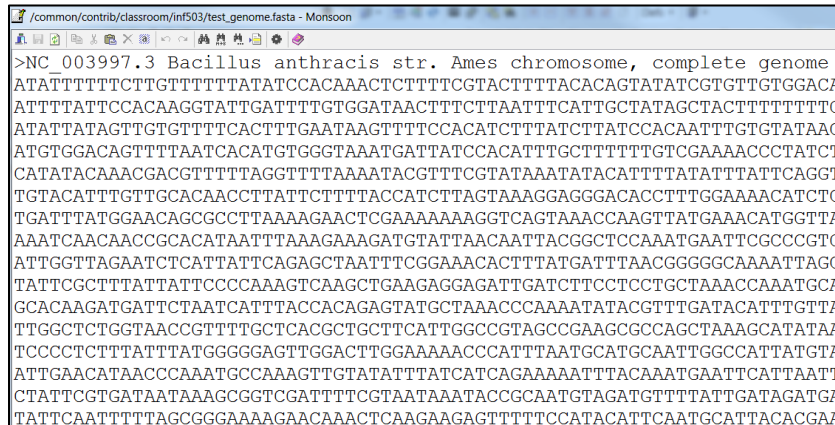
Each homework submission must include:

- 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 readme.txt file outlining all modules (if any) needed for the execution of the code and the exact command lines needed to answer homework's questions (**5pts**)
- A full write-up (.pdf or .doc) file containing answers to homework's questions (**5pts**) – screenshots of code output are ok.

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).
- 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)



```
/common/contrib/classroom/inf503/test_genome.fasta - Monsoon
>NC_003997.3 Bacillus anthracis str. Ames chromosome, complete genome
ATATTTTCTTGTGTTTTATATCCACAACTCTTTTCGTACTTTTACACAGTATATCGTGTGTGGACA
ATTTTATCCACAAGGTATTGATTTTGTGGATACTTTCTTAATTTTCATTGCTATAGCTACTTTTTTTG
ATATTATAGTTGTGTTTCACCTTTGAATAAGTTTCCACATCTTTATCTTATCCACAATTTGTGTATAAC
ATGTGGACAGTTTAAATCACATGTGGGTAAATGATTATCCACATTTGCTTTTTTGTGAAAACCCATCTC
CATATACAAACGACGTTTTAGGTTTTAAATACGTTTCGTATAAATATACATTTTATATTTATTCAGGT
TGTACATTTGTTGCACAACCTTATCTTTTACCATCTTAGTAAAGGAGGGACACCTTTGGAAAACATCTC
TGATTTATGGAACAGCGCTTAAAGAACTCGAAAAAAGGTCAGTAAACCAAGTTATGAAACATGGTTA
AAATCAACAACCGCACATAATTTAAAGAAAGATGTATTAACAATTACGGCTCCAAATGAATTCGCCCGTG
ATTGGTTAGAACTCTATTATTCAGAGCTAATTTCCGAAACACTTTATGATTTAACGGGGGCAAAATAGC
TATTCGCTTTATTATTCCCCAAAGTCAAGCTGAAGAGGAGATTGATCTTCCTCCTGCTAAACCAAATGCA
GCACAAGATGATTCTAATCATTTACCACAGATGCTAAACCCAAAATATACGTTTGATACATTTGTTA
TTGGCTCTGGTAACCGTTTTGCTCACGCTGCTTCATTGGCCGTAGCCGAAGCGGCAGCTAAAGCATATAA
TCCCTCTTTATTATGGGGGAGTTGGACTTGGAAAAACCATTTAATGCATGCAATTGGCCATTATGTA
ATTGAACATAACCCAAATGCCAAGTTGTATATTTATCATCAGAAAAATTACAAATGAATTCATTAATT
CTATTCGTGATAATAAAGCGGTCGATTTTCGTAATAAATACCGCAATGTAGATGTTTTATTGATAGATGA
TATTCATTTTGTAGCGGAAAAGAACAACTCAAGAAGAGTTTTTCCATACATTCATGCATTACACGAA
```

Problem #1 (of 2): 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 given 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. `RO_0_1...`). At minimum, the class must contain:

- A default constructor
 - At least one custom constructor (e.g. one taking a file path or ifstream as input)
 - A function to read the FASTA fasta file
 - A destructor
 - A copy constructor
- A. Read in the entire 36 million read set and report RAM and CPU time used to load the data into memory.
- B. Implement a destructor for your class to delete / deallocate your array data structure. How long did it take? Does this make sense to you? Explain why.
- C. Implement a copy constructor and perform a deep copy of the entire FASTAreadset_LL object. How long did it take? Does it make sense? Explain why.
- D. 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 a match OR the NULL pointer value if a 'hit' was not found. Which of the following sequences were found in the read set:
- CTAGGTACATCCACACACAGCAGCGCATTATGTATTTATTGGATTATTT
 - GCGCGATCAGCTTCGCGCGCACCGCGAGCGCCGATTGCACGAAATGGCGC
 - CGATGATCAGGGGCGTTGCGTAATAGAACTGCGAAGCCGCTCTATCGCC
 - CGTTGGGAGTGCTTGGTTTAGCGCAAATGAGTTTTCGAGGCTATCAAAAA
 - ACTGTAGAAGAAAAAAGTGAGGCTGCTCTTTTACAAGAAAAAAGTNNNNNN

Problem #2 (of 2): Basic search

Read in the *Bacillus anthracis* genome (/common/contrib/classroom/inf503/test_genome.fasta). Read in the sequence read fragments used in problem #1 – you must use the FASTAreadset_LL created in the previous problem to store your sequence reads.

- A. Break down the genome sequence into all 50-character long fragments contained within. Store these fragments in an array or linked list data structure (your choice). How many 50-character fragments did you observe in the genome?
- B. Iterate through all 50-mers found in the genome, using the search function you developed in 1D to query the read set. How many genome 50-mer fragments were found in your read set? How long it take to complete the entire search process (all 50-mers)?

Note that the problem 1B may take a LONG time to complete. If you feel that you will not be able to complete the execution of the program in time to submit the assignment, estimate the total amount of time and the number of found 50-mers by running 1000, 10000, 100000 searches and estimating (extrapolating) the outcome from those.