CSC 448: Bioinformatics Algorithms **FALL 2015**

Lab -1

Due: Tuesday, September 29

Now, let us do it automatically…

In this lab, you are asked to create software that performs the task of determining the word, encoded in a DNA message (the exercise you went through during the first day of classes with your BIO 441 partners) automatically.

This is a pair programming lab. You get to pick a partner.

Specifications.

In the game you played, on each round, you were provided with four DNA strings in nucleotide alphabet (A,T,C,G). Your collective goal was to determine a message (usually a single word) encoded in the DNA. The message was encoded on a pair of DNA fragments out of the four you had (the other two were fakes). The two DNA fragments that form the message must have an overlap of at least four base pairs as shown below:

A T G T T G C T T G A T G A T C
A T G A T C T T G C C T G T A A

In the game, you performed a role of three functions that manipulate DNA strings, while your BIO 441 partners performed a role of the software that searched for the correct word.

The three DNA manipulation functions are:

Reverse Complement. This function takes a DNA string (e.g., ATGCTA) and returns its ***reverse complement*** (TAGCAT).

Translation (in a specific reading frame). The function takes a DNA string in a nucleotide alphabet (e.g., ATGCTA) and a reading frame (from a list 1, 2, 3: for example 1), and returns back a translation of the input string into the aminoacid alphabet in a given reading frame. For example, the translation of ATGCTA in reading frame 1 shall return ML.

Attempted Merge. The function takes two DNA strings (order is important), for example ATGCTA and GCTAGT, and attempt to find a merge point for the strings - i.e., align them so that the tail end of the *first string* matches the beginning of the *second string.* For example for the the strings above, the following match will be discovered:

A T G C T A
G C T A G T

The function returns the merged string: in the example above the string is ATGCTAGT.

Note: Unlike reverse complement and translation, *merge* is a difficult operation, and we will spend some time in the course talking about implementing it efficiently. For the purposes of this lab, please implement the following *naive algorithm* for this function:

Starting with a possible overlap of 4 characters and until all characters are exhausted, check whether the two strings align properly.

To illustrate, here is how your implementation would work on the pair of strings AAAATTTT and AATTTTTGG:

Step 1: try to align: AAAATTTT
AATTTTGG
result: fail
Step 2: try to align AAAATTTT
AATTTTGG
result: fail
Step 3: try to align AAAATTTT
AATTTTGG
SUCCESS: return AAAATTTTGG

Assignment.

Develop a program that plays the Lab 1 game automatically and decodes the message hidden in a pair of DNA strings.

The program takes as input a text file with four DNA strings in it (one string per line), and outputs a word encoded in the message.

The encoded message is a single English word (we will limit ourselves to single words). You can use `/usr/share/dict/linux.words` to determine if what you got is a proper English word.

Submission. Submit your code together with instructions for compiling and running. I will provide test data.
Use
handin dekhtyar 448-lab01 <files>