**HW3: Finding hairpin structure sequences using LCS**

**Algorithms, Spring 2017, SNU**

**Due**: June 5th 11:59 P.M.

**Submission to** Hongryul Ahn (hrahn@snu.ac.kr)

(mail title: [Algorithm HW3] StudentID Name)

**What to submit:** a zip file that includes README and your program. The file should be named with a string “hairpin” then your university id

ex) 2011\_11111\_hairpin.zip (that include README and source code)

README file should contain all command lines that should run by cut-and-paste. If you used C, then

$ gcc –o hairpin \_2011\_11111 hairpin\_2011\_11111.c

$ hairpin\_2011\_11111 DNAseq.fasta

**Input:** a long DNA sequence can be downloaded from

http://epigenomics.snu.ac.kr/teaching/2017/algorithm/HW3/DNAseq.fasta

(This is a sequence in FASTA format that begins with a single-line sequence id (starts with “>”), followed by lines of sequence data)

**Output:** Print out all the hairpin structure sequences with the LCS sequence and the loop sequence. Suppose that you find two hairpin structure sequences: “AACTGCTTCAA” with “AATC” LCS sequence and “GCT” loop sequence and “CTTAGATGGATC” with “CTAG” LCS sequence and “ATG” loop sequence. Then, the output example is

/\*-------------------------------------Output example-----------------------------------------

AACTGCTTCAA

AACT

GCT

CTTAGATGGATC

CTAG

ATG

-------------------------------------Output example-----------------------------------------\*/

**Programming language:** any of C, C++, Java, Perl, Python

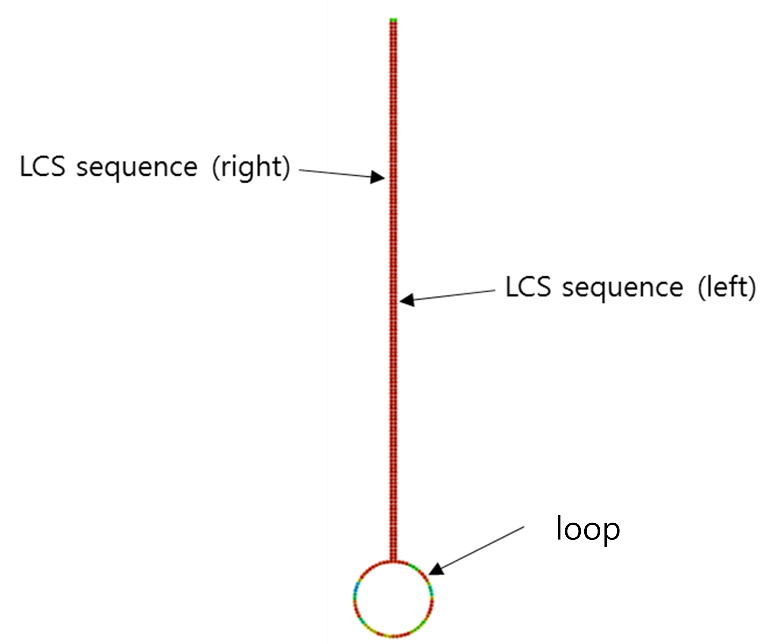
**Description**

In HW3, you will implement a program to detect the hairpin structure sequence using longest common subsequence (LCS). The human genome is 3.3M characters long, which embeds specific structure sequences, called “hairpin structure sequences”. A hairpin structure sequence is defined as a partially palindromic sequence where the left and right sides of the sequence show palindromic relationship (i.e. the left and the right are in reverse order), but the middle part, called loop, is not palindromic. Table 1 shows an example of "hairpin structure sequence” where the RED left and the BLUE right sequences are the same in reverse order and separated by BLACK loop sequences. So it can be folded in shape of hairpin (Figure 1). In the given biological sequence, a few hairpin structure sequences are embedded. By using LCS, we are able to detect hairpin structure sequences. The implementation of LCS must be based on dynamic programming (Wikipedia [link](https://en.wikipedia.org/wiki/Longest_common_subsequence_problem)). Your program should output the hairpin structure sequences.

**Example**

Table . an example of "hairpin structure sequence” (It is just an example). the RED left and the BLUE right sequences have palindromic relationship and the BLACK middle sequence, called loop, is not palindromic.

|  |
| --- |
| > A long DNA sequence  …TGACATCACCTCTTTCTTTCTCAACACACCCACAAAGGCACACACTGCTGCATAATTTTGCTTTTGTCTGAGGAAGAAAAATTTAATAACGATACCAATTTTTATTTTTTAATTTTATGTATAAATTAGAAACTACATATGAGGAGAATACCAGACGTTATTTTTTTGAACGACCACATACATAGCATACACATAATAAATTTAAAAGAGGAGTATACATCAAAGATTAAATATGTATTTTAATTTTTTATTTTTAACCATAGCAATAATTTAAAAAGAAGGAGTCTGTTTTCGTTTTAATACGTCGTCACACACGGAAACACCCACACAACTCTTTCTTTCTCCACTACAGT… |



**Figure 1.** The secondary (folded) structure of the hairpin structure sequence

**What to implement and report**

Since the human genome (3.3 million nt) is very long, we cannot scan the whole genomic sequence. The input sequence in HW3 is much shorter, 0.1M nucleotides (characters) long. In HW3, you are asked to use **k-mers** (k size words) to find anchor points. The anchor points are used to first find candidate regions to perform dynamic programming. We recommend to build a k-mer dictionary first using the whole biological sequence. Then find k-mer pairs that are located within 300 nt. Using the anchor points, choose a region of length X containing the anchor points and perform dynamic programming. Notice that the two left/right palindromic sequences are placed in **reverse** order. The palindromic sequences are separated by a **<50 nt loop sequence**. The palindromic sequences can contain insertions and deletions, thus they may not be perfectly the same.

1. Perform a k-mer search on the input sequence and select candidate anchor points. Anchor points are a pair of k-mers within 300 nt distance. Since we are searching for palindromic sequences, the k-mers should also reflect this.
2. Implement dynamic programming to find a local alignment of two palindromic sequences based on the chosen anchor points. You should try different sequence lengths and select the best, by incrementing the sequence length. Notice that the two palindromic sequence are separated by a <50 nt loop sequences. Thus, the loop sequence should not be considered during the local alignment.
3. Termination condition: Only three or less insertion and deletion characters can appear continuously in a hairpin structure sequence in HW3. In other words, if LCS does not appear continuously in four or more characters, the hairpin structure sequence should stop extending and be finalized
4. If you find a hairpin structure sequence, print out the hairpin sequence, LCS and loop sequence
5. Repeat this process until you reach the end.

**Hint**

The length of loop sequence < 50

The length of hairpin structure sequence including loop x is 200 ≤ x ≤ 400

**Notice**

Your program should end within 10 min because TA should test about 100 programs.