Assignment 6

Problem 1: Bioinformatics

Samantha, a diligent researcher in a Computational Biology Lab, has made a groundbreaking discovery—a new species with a unique genetic signature. She reached out to her renowned friend, Beyonce, who graciously sequenced the organism (again, let's all take a moment to thank her for her contribution), and provided Samantha with a text file containing a lengthy sequence of nucleotides (A, T, G, C). However, the file data is riddled with unexpected spaces and gaps, likely due to formatting issues during transfer (we'll excuse Beyonce this time for the sake of science).

Note: The input file is provided as ***"dna.txt"***.

Samantha now needs to analyze this data to reveal all potential proteins that the organism can produce. To assist her, follow these steps:

1. Clean the file, remove all invalid symbols (anything other than A,T,G,C). Obtain the complement of the DNA strand. In this context, complement means: **A binds with T, G binds with C, C binds with G, and T binds with A .**

2. Using the complement DNA strand, generate the corresponding RNA strand. In this transcription process, **replace all Thymine (T) nucleotides with Uracil (U)** to convert the DNA strand into an RNA sequence.

3. Finally, interpret the RNA strand to identify all possible proteins. For this, use the provided codon-to-amino-acid mapping file, ***'codon.txt'***. Process the RNA sequence in overlapping triplet groups (i.e., frameshifts) and identify all potential proteins and print them onto the terminal. Start from the 'AUG' sequence and continue the encoding until you encounter a STOP codon.

File Format:

- File contains numerous lines of characters.
- take in 2 input files: "dna.txt" and "codon.txt"

- File "dna.txt" contains a sequence of nucleotides (A, T, G, C) (case sensitive - do not consider "a", "t", "g", "c" as nucleotides) with unexpected spaces and gaps. (never empty)
- File "codon.txt" contains a codon-to-amino-acid mapping in the following format:

  ```
  UUU F
  UUC F
  UGA _
  UUG L
  ```

Output Format:

- Output the list of proteins as a sequence of amino acids onto the terminal

Constraints:

- The number of characters in the file do not exceed 200000.

Example File:

```
TGCATGCTTAGTGCACTCACGCAGTATAATTAATAAC
```

Example Output 1:

```
NA
```

Example File:

```
ATTAAAGGTTTATACCTT
```

Example Output 2:

```
ME
```

Explaination:

```
DNA: ATTAAAGGTTTATACCTT
Complement: TAATTTCCAAATATGGAA
RNA: UAAUUUCCAAAUAUGGAA
```

```
Protein sequences:
    ME
```

While encoding the amino acids, look for the codon/sequence "AUG" and sequence until the next "STOP" codon. Number of proteins would be equal to the number of "AUG" codons.

Important Notes:

- Do not use any external libraries.(can use string.h not any inbuilt bio-libraries)
- The file format is given. Do not change the format.
- The output format is also given. Do not change the format.

Problem 2: We're Expecting!

Domingo and Kelsey are preparing for the arrival of their first child and have compiled two separate, sorted lists of potential baby names, each stored in an input file. To make it easier for them to choose, they want a program that will merge these lists, count the occurrences of each name (to indicate popularity), and display a list of the top 10 unique names by their frequency.

The task is to write a C program that:

1. Takes an input file containing two sorted lists of baby names, separated by a special identifier &
2. Merges the two sorted lists into one unified list and counts the occurrences of each name.
3. Displays the frequencies of the first 10 names onto the terminal.

Input Format:

Two alphabetically sorted lists of baby names, separated by a special identifier "&". The filename is ***"names.txt"***.

```
Angel
Baker
Baker
Chase
....
Zepplin
&
Aspen
```

```
Chase
Domino
....
Zudio
```

Output Format:

Output the space-seperated frequencies of the first 10 names onto the terminal.

Constraints:

- Each name is at max 16 characters long.
- There are atmost 10000 names in each list.

Input 1:

```
Angel
Baker
Baker
Chase
Domingo
Espen
Fonika
Gru
Howie
Sven
&
Aspen
Chase
Domino
Fonika
Fonika
Gru
Gru
Gru
Sawyer
```

Output 2:

```
1 1 2 2 1 1 1 3 4 1
```

Explaination:

The merged names list is as follows:

```
Angel 1
Aspen 1
Baker 2
Chase 2
Domingo 1
Domino 1
Espen 1
Fonika 3
Gru 4
Howie 1
Sawyer 1
Sven 1
```

Problem 3: Processing Huge Files in Batches

You are given two very large files (**input1.txt** and **input2.txt**) containing strings, with each string on a new line. Due to the size of the files, it is not feasible to load their contents into memory all at once. Your task is to process these files in batches to perform the following operations:

1. **Reverse** the contents of **input1.txt** (reverse each string line by line) and write the reversed lines to a new file named **output.txt**.
2. After processing **input1.txt**, reverse the contents of **input2.txt** (line by line) and append the reversed lines to the same **output.txt** file.

Input Format:

Two input files:

- **input1.txt**: Contains a very large list of strings, one string per line.
- **input2.txt**: Contains another very large list of strings, one string per line.

Output Format:

A single output file named **output.txt** where:

- All reversed lines from **input1.txt** appear first.
- All reversed lines from **input2.txt** appear next.

The processing must be performed in batches to handle large file sizes efficiently.

Constraints:

- You cannot load the entire contents of the files into memory. The files could be upto 50GB in memory.
- Process the files in manageable batches.

Example:

Input Files:

input1.txt:

```
Hello
World
This
Is
InputOne
```

input2.txt:

```
Another
Batch
Of
Strings
```

Output File:

output.txt:

```
olleH
dlroW
sihT
sI
enOtuptnI
rehtonA
hctaB
```

```
fO
sgnirtS
```

Submission Guidelines

Do not rename any files given in the handout. Only write the code in the specified C files in the respective directories.