ECE368 Programming Assignment #5

Due Friday, April 14, 2017, 11:59pm

Description

This project is to be completed on your own. The problem is made available by Dr. Ken Sung from the School of Computing, National University of Singapore. The problem is in the area of comparative genomic, where the goal is to find a longest gene sequence that is conserved among a set of species.

Each integer from a set of n integers $\{1,2,\ldots,n\}$ represents a gene. Assume that we are given m species S_1, S_2, ..., S_m, with each species S_i identified by a permutation of $\{1,2,\ldots,n\}$, which represents the ordering of genes in S_i.

Given a permutation, a subsequence of the permutation is obtained by omitting some integers from the permutation. For the permutation $\langle 5,3,4,1,2\rangle$, both $\langle 5,3,4,1,2\rangle$ and $\langle 3,4,2\rangle$ are valid subsequences. However, $\langle 4,3\rangle$ is not because the ordering in the permutation is not preserved.

If a subsequence of length k $\langle x_1,x_2,...,x_k\rangle$ appears in all m species, we call that a *conserved gene sequence* among the m species. Consider the following 3 species:

- $S_1 = \langle 5,3,4,1,2\rangle$
- $S_2 = \langle 2,5,4,3,1\rangle$
- $S_3 = \langle 5,2,3,1,4\rangle$

The subsequences $\langle 5,4\rangle$, $\langle 5,3\rangle$, $\langle 5,1\rangle$, $\langle 3,1\rangle$ are all conserved gene sequences, each of length 2. The subsequence $\langle 5,3,1\rangle$ is the longest conserved gene sequence, which is of length 3.

The goal of this project is for you to write an efficient program to compute the length of a longest conserved gene sequence for m species.

Functions you have to write:

You have to write your own include file `genome.h` and source file `genome.c`. The `genome.h` file should declare the following function:

```
int *Longest_conserved_gene_sequence(char *filename, int *size_of_seq);
```

Given the name (`filename`) of the input file, the function returns the address of an array containing a longest conserved gene sequence. The length of that longest conserved gene sequence is stored in `*size_of_seq` at the end of the function. `This should be the only function declared in genome.h. You may define structures that you need in genome.h.`

The function `Longest_conserved_gene_sequence` should be defined in `genome.c`.

The input file is in **binary** format. Assuming that a gene sequence contains n integers and there are m sequences altogether. The input file contains $(2+m\times n)$ integers (in binary). The first two integers in the file store n and m, respectively. The next n integers correspond to the first gene sequence. The second n integers correspond to the second gene sequence. The last n integers

correspond to the m-th gene sequence. **You may assume that $n \leq 100,000$ and $m \leq 10,000$. You may also assume that if an input file can be opened, it is of the correct format.**

There may be several longest conserved gene sequences. You are allowed to store any one of them in the array, whose address is returned. You should allocate the space for the array in this function. **If the input file cannot be opened or read properly, or the function cannot not proceed because of lack of memory, you should return** `NULL` **and set** `*size_of_seq` **to 0.**

It is likely that you will have to write other helper functions in the file `genome.c`. **You should declare and define these helper functions as** `static`**. As they are static, it is not necessary (and you should not) declare them in** `genome.h`**.**

You should also provide the `main` function in `genome_main.c`. **In** `main`**, you may call many functions. However, the only function in** `genome.c` **that you can call in** `main` **is** `Longest_conserved_gene_sequence`**.**

Deliverables:

You are to develop your own include file `genome.h` and source files `genome.c` and `genome_main.c`, which can be compiled with the following command:

```
gcc -Werror -Wall -Wshadow -O3 genome.c genome_main.c -o proj5
```

The executable `proj5` would be invoked as follows:

```
./proj5 input_file
```

The executable loads the gene sequences (permutations) from binary `input_file`, calculates the length of the longest conserved gene sequence, and print the length to the screen.

For the example discussed earlier, the screen dump should look like this:

```
Length: AAAA
```

where `AAAA` is an integer that corresponds to the length of the longest conserved gene sequence. In this example, `AAAA` is 3.

Grading:

The project requires the submission (through Blackboard) of a zip file containing the source and include files and a report detailing the results. The report should contain a description of the approach you have used to solve the problem. You should also comment on the run-time complexity of your algorithm. Your report should not be longer than 1 page and should be in plain text format or pdf.

You should create a zip file called proj5.zip that contains all of the above and submit the zip file. If the zip file contains a makefile, we will use that file to make your executable.

Grading:

The grade depends on the correctness of your program, the efficiency of your program, the clarity of your program documentation and report.

The report will account for 10% of the entire grade. The `main` function will account for 10% of the entire grade. The `Longest_conserved_gene_sequence` accounts for 80% of the entire grade.

It is important all the files that have been opened are closed and all the memory that have been allocated are freed before the program exits. Memory errors will result in at least 50% penalty.

Getting started:

We provide a sample input files (in `samples.zip`) for you to evaluate your program. The input files are `genome.in[0-6]`, and files storing the corresponding screen output are `genome.log[0-6]`.

For your convenience, the zip file also contains `genome_a.in[0-6]`, which are the text format of the binary input files. In the text format, the first line stores the length of a gene and the number of species. Subsequent lines store the genes. Each of the `genome.out[0-6]` files stores the length of the longest conserved gene sequences and also a longest conserved gene sequence. However, you should not write your program to accept input files of the text format. Also, you should not try to match the longest conserved gene sequences in these files.

The example provided in this description can be found in `genome.in0`.

Copy over the files from the Blackboard website. Check out the Blackboard website for any updates to these instructions.