Design and Analysis of Algorithms
CS375 Spring 2017

Programming Assignment 1

Release Date: 3/1/2017
Due: 3/15/2017 (Wednesday) at start of class

Tasks

1. Write a program that solves the Longest Common Subsequence (LCS) problem by the bottom-up dynamic programming approach. The inputs to the program are two files **filex.txt** and **filey.txt**. The output is written into **output1.txt**. [40%]

   The input files:

   Each **filex.txt** contains one line with a sequence of characters (without space). For example: abacabbcdeaa

   Similarly each **filey.txt** contains a sequence of characters (without space). For example: aaabbccddadeee

   The output file:

   For input strings of size less or equal to 10:
   Each line *i*, for $i = 0$ to m (m = length of the string in **filex.txt**) of the output file will contain a row of the matrix lenLCS (as shown in the examples in Lecture 15). It will contain the lenLCS[i, j] for columns $j = 0, 1, \ldots, n$ (n = length of the string in **filey.txt**).
   Line m+1 will contain a longest common subsequence.
   Line m+2 will contain the running time of the algorithm.

   For inputs of size greater than 10 the output file will contain:
   Line 1: The length of the LCS
   Line 2: The running time of the algorithm

2. Write a program that solves the LCS problem by recursively computing the length of a longest common subsequence (without memoization). The input as before are the files **filex** and **filey**. The output is written into **output2.txt** containing: [30%]

   Line 1: The length of the LCS
   Line 2: The running time of the algorithm

   Note: this program does not compute and store matrices and you are not required to find a longest common sequence.

3. Write a program that solves the LCS problem by the top-down dynamic programming approach - recursively computing the length of a longest common subsequence with memoization. The input as before are the files **filex** and **filey**. The output is written into **output3.txt** containing: [30%]

   Line 1: The length of the LCS
   Line 2: The running time of the algorithm

4. Verify the correctness of your programs based on the example in Lecture 15 and other test data of your own containing strings of different lengths. Run the three programs with the same test data and pay attention to their running time. **Note:** this task is not going to be graded. Your programs from Tasks 1 to 3 will be graded based on test data containing strings of length up to 100.

Directions and Requirements:

1. All three programs above should be run like this.
Prompt> program1 <filex.txt> <filey.txt> <output1.txt>
Prompt> program2 <filex.txt> <filey.txt> <output2.txt>
Prompt> program3 <filex.txt> <filey.txt> <output3.txt>

2. Submit a *.zip* (or .tar) file through the submission link at Blackboard.
The zip file should be named (lower case) as follows:
<last name>_<first name>
When the file is unzipped it should contain a directory with the same name as the zip file. The directory should contain the following files:

> File(s) for the source code of the three programs.
>
> README file named for each of the 3 programs *readme*1.*txt, readme2.txt, and readme3.txt* which provides details on how to compile the source code and additional documentation.
>
> Declaration of Academic Integrity file named declaration.txt which **include the following statements and your full name**:
>
> "I, _____, have done this assignment completely on my own. I have not copied it, nor have I given my solution to anyone else. I understand that if I am involved in plagiarism or cheating I will have to sign an official form that I have cheated and that this form will be stored in my official university record. I also understand that I will receive a grade of **0** for the involved assignment for my first offense and that I will receive a grade of **"F" for the course** for any additional offense."

3. Additional general requirements for programming assignments from the course syllabus (**see item 5**) also must be followed.