Bonus Assignment: Longest Consecutive Words

ECE 345 Algorithms and Data Structures
University of Toronto
Dept. of Electrical and Computer Engineering
Fall Semester, 2014

Due date: Dec 12, 2014 by email to `briank@eecg.toronto.edu`

1 Introduction

This bonus assignment is intended to allow you to learn about some of the practical aspects of algorithms and data structures by solving a programming problem in C++. It is worth a bonus of up to **4%** of your final grade and is *completely optional*. This is an individual assignment (no groups).

2 Problem

You are given an input file with multiple lines where each line contains a list of words separated by a space. All words contain only lowercase letters (`a-z`). Write an efficient C++ program to find and output the *longest common consecutive sequence of words* belonging to *every* line in the file.

Here's an example input file with three lines:

```
here lies a foot at the bottom of my bed
the bottom of my bed contains more than i like to know
at the bottom of the sea you might find mighty creatures such as these
```

Your algorithm should return:

```
the bottom of
```

You may assume that there exists a longest common consecutive sequence of words and that it is unique.

Sample input files (`input1.txt`, `input2.txt`, `input3.txt`) are available for testing.

2.1 Details

Your project should contain the following:

- Be written in standard C++ (no external libraries except ones included in C++, specifically `gcc-4.6.3`).
- A `Makefile` (using `GNU Make`) that calls `gcc` to compile your source files to generate an executable named `lcs`
- Tar and gzip your source files and Makefile for submission into a filename named `<student number>.tar.gz`
- Your program should accept a single command line argument for the input file and output the longest common consecutive sequence of words across all lines.

An example of how the tester program will run your code:

```
> tar -xvzf 123456789.tar.gz
> make
> lcs /path/to/input0.txt
the bottom of
```

Please ensure that your submission can be extracted, compiled and executed as shown above as there will be minimal (if any) effort done to fix your submission by the marker if something fails.

2.2 Testing

Your final program will be tested against the above input files as well as several other test files that will only be available during marking. The correctness of your program will be determined by comparing your program output to the correct answer. The efficiency will be judged by the program run-time with respect to all other submissions as well as a reference implementation. If your program fails to finish running within 600 seconds or uses more than 4 GB of RAM for a given execution, it will be considered as a failure to produce the correct output.

3 Deliverables

There are two files to be emailed to `briank@eecg.toronto.edu`

1. A PDF report (4 pages max) named `<studentid>.pdf` containing the following:
   (a) Your name, student number, date, and assignment name.
   (b) A brief introduction to the problem.
   (c) High-level description and intuition of your algorithm. Be sure to include descriptions of what algorithms you investigated and why you chose your particular algorithm.
   (d) Description of implementation of your algorithm which should describe the important code, data structures, and optimizations used. Be sure to include descriptions of the different approaches you tried *why* you made a particular choice.
   (e) Big-$\mathcal{O}$ asymptotic analysis of the run-time and memory consumption of your algorithm.
   (f) The output of your program for the three sample input files (`input1.txt`, `input2.txt`, `input3.txt`).
   (g) A brief conclusion of what you learned.
2. A gzipped tarball named by your student id (`<studentid>.tar.gz`) containing your C++ source code and a Makefile.
   - The tar file should only contain `*.cpp/*.h` files in addition to a single `Makefile`.
   - Your C++ code must be able to compile on `gcc-4.6.3/GNU Make` and run on Linux.

4 Marking Scheme

The report will be marked out of 100 points with the following breakdown:

- 50 points for the report.
- 25 points for program generating correct answer (and finishing within a reasonable amount of time/memory).
- 25 points for program run-time efficiency (relative to reference program and other students).

5 Useful Resources

- C++ Reference: `http://www.cppreference.com`
- Longest Common Substring problem:
  `http://en.wikipedia.org/wiki/Longest_common_substring_problem`