CSCI-1200 Data Structures — Spring 2018
Lab 9 — Maps I

This lab gives you practice initial practice in working with the STL associative container, maps. No downloads are needed until Checkpoint 3. If you have checkpoints from Lab 8 that need to be checked off, you can do so only at the *beginning* of Lab 9. Do not wait until you have Lab 9 Checkpoint 1 done to get your Lab 8 checked off!

Checkpoint 1

Write a program from scratch that uses a `map` to find **all** the modes in an input sequence of integers. Remember, a mode is an integer that occurs at least as many times in the sequence as any other integer. Thus, in the sequence

```
 19  83  -12  83  65  19  45  -12  45  19  45
```

the two modes are 19 and 45. Include one command-line argument to provide an input file. Use `operator[]` for maps when inserting values.

To complete this checkpoint: show a TA your debugged implementation and how it runs correctly on several interesting test cases.

Checkpoint 2

Rewrite your program from checkpoint 1 to use `find` or `insert` or both instead of `operator[]`.

To complete this checkpoint: show a TA your revised and tested program.

Checkpoint 3

Please download the following file needed for the final checkpoint and turn off your internet connection:

`http://www.cs.rpi.edu/academics/courses/spring18/csci1200/labs/09_maps/phonebook.cpp`

This code implements a simple caller ID program. This program could be used by a university or company to perform a "reverse lookup" — given the 4 digit phone number extension, return the name of the caller. A vector of strings stores the complete database of names assigned to phone number extensions. Compile and run this program. Add your own test cases to the main function.

Part 1: Analyze the computational cost of this program using the order notation in terms of n the number of assigned phone numbers in the phonebook, and N the largest possible phone number. What is the running time of constructing the phonebook and of the add and identify functions? How much memory is used by this program? Express this using order notation. What would happen if you extended this to a 7- or a 10-digit number? Would it work on a cell phone?

Part 2: Rewrite this program to use maps, storing only the numbers that are assigned. Analyze the cost of creating the map, and of the add and the identify functions. Test it on 7 digit numbers. In what ways is the vector version better and in what ways is the map version better?

To complete this checkpoint: Present your analysis from Part 1 to a TA, demonstrate the new version of your program, and be prepared to discuss your analysis of Part 2.

Extra Credit Checkpoint 4

Stay through the end of lab and work on Homework 6. Present your progress to a TA and discuss plans for testing, debugging, improving the clarity of the code, and/or improve the performance of your algorithm.