

Lab 7: Practice with classes and vectors

CPSC 1021 – Summer 2019

**Due: Thursday, June 6, 1:59 PM**

# Lab Objective

* Practice writing classes
* Working with vectors
* Reading data from a file
* File input/output
* Practice with composition “has-a” relationship between classes
* Practice with iterators

# Introduction

You will be given several .h files, each containing a class declaration or one or more functions. It is your job to implement the functions.

The files you will be given are as follows:

Person.h

Address.h

Date.h

Functions.h

Data.txt

You are to create a .cpp file for each of the above .h files. You are to implement the functions in the .cpp files. Functions.h has one helper function that uses an instance of person. The prototype for this function is as follows:

vector<Person> readData(string);

The parameter of type string represents the name of the input file which will be defined on the command line (command line argument). This function also returns a vector of type Person.

I have given you Data.txt which is the file I used to test my program. This is the format of the data we will use when testing your program.

Once you have successfully read in and stored **ALL** of the data you are to simply print the information to an output file. Notice each class has a printInfo function. **Before you print information for each person, you have to sort them by their last name.** After you have read the data from the file, stored it in the vector of type Person, you will sort the vector by a person’s last name and loop through the vector calling printInfo. **You can make use of the sort utility function provided by algorithm library and write your own comparison function. To give you practice using iterators, you are required to use an iterator to iterate through the vector when calling printInfo.** (See the class notes for an example.) Since, the vector is of type Person, Person’s printInfo should print the information specific to Person. Using the instance of Address and Date that person has, you should call Address and Date’s printInfo, each printing the information specific to their class. In addition to a sample input file, I will give you a sample output file.

Each of your files must have a header similar to the header listed below

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*Your name

\*CPSC 1020 Sm19 lab 6

\*Your email

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

1. Your lab should be tested on the SOC servers prior to submitting the files.
2. Tar zip your files naming the tarred file <username>\_lab<#>.tar.gz. Your tar file should **NOT** have nested files. Change directories to the folder that contains your file, using a terminal type

tar –cvzf **<yourUserName>\_lab<#>.tar.gz** \* The (\*) is a wildcard and tells the tar utility to tar everything in the folder.

1. There should be no lines of code or comments over 80 characters
2. Your code should have neat and consistent indention.