Virginia Tech
Bradley Department of Electrical and Computer Engineering
ECE-3574: Applied Software Design Spring 2016

Homework 1

Submission Details

You must submit the solution for this homework in electronic form through Scholar (under ECE3574 → Assignments → Homework 1). The submission must be a gzipped tar file (.tar.gz) with your source code. Include all necessary project files, but no binary or compiled files. Your program will be run to evaluate its correctness, and the source code will be reviewed for adherence to the Qt programming style. Your program must run on Ubuntu 15.10.0 and compile/build using the GNU C/C++ compiler and the qmake/make tools. The following information must be included at the top of each of your source files as comments: your full name, your student ID number, your email address, class (ECE 3574), and the title of the assignment (Homework 1). The submitted file must be given a name in the following form: ***LAST_FIRST_hw1.tar.gz*** where LAST is your last or family name and FIRST is your first or given name. You are only allowed to make one submission. Paper, email or Drop Box submissions will not be accepted. All work must be submitted by the announced due date/time. **Late submissions will not be accepted!** (Don't do it! You have been warned!)

Questions

Use the Homework 1 forum in the Discussion Board area of the class web site to ask questions about this assignment. Do not post questions that contain specific information about the solution.

Honor Code

As stated in the syllabus, discussion and cooperative learning are allowed when working on homework and projects. However, copying or otherwise using another person's detailed solutions to assigned problems is an honor code violation. See syllabus for details.

Learning Objectives

The primary learning objectives of this homework include learning how to write code using a library such as Qt and reinforcing the concepts of containers and the iterator pattern. Additional objectives include getting exposure to the Qt development environment and to console-based Qt application development.

Exercise

Write a birthday reminder application called *birthdays*. The classes you **must** use are QDate, QFile, QString, QStringList, QTextStream, and std::exception. You may find some other Qt classes useful too.

- Store name/birthday pairs in any format you like, in a file called birthdays.dat.
- *./birthdays* without any command line arguments lists all birthdays coming up in the next 30 days.
- *./birthdays -a yyyy-MM-dd "John Smith"* adds or updates the birthday for "*John Smith*".
- *./birthdays -n 40* shows birthdays coming up in the next 40 days.
- *./birthdays <NAMESPEC>* shows all birthdays that contain the substring <NAMESPEC>
- *./birthdays -d "John Smith"* should delete this entry from the file.
- *./birthdays -d yyyy-MM-dd* should delete all entries with this birthday from the file.
- *./birthdays -m "John Smith" 40* shows birthdays coming up in the next 40 days from John Smith's birthday. This includes John Smith's birthday.
- *./birthdays -u* updates records as needed such that all birthday reminders correspond to the person's next birthday (i.e. changes dates in the past and dates more than 1 year in the future to dates within the following year – exception birthdays on February 29th, which are allowed to be up to four years in the future). If today is 2016 August 21st a birthday for today stays the same where a birthday on 2017 August 21st becomes today.
- EXTRA CREDIT: make the previous command, *./birthdays -u* also list what records it updates. This time, a third column should be displayed, showing the time left until the birthday, in this format: 8 months, 3 weeks, 2 days. If any of these numbers is zero, do not display it (eg. 8 months, 2 days). When displaying this duration you may assume that the unit "months" is 30 days and "weeks" is 7 days.

Notes

- All output should be in a format similar to a table (in two columns) like this:

```
   Name          Birthday
   ====          ========
John Smith        2016-11-04
 Jenny Doe        2016-12-29
```

- For any illegal input, you have to throw exceptions and have a top-level exception handler deal with them. The handler should print error messages to *stderr* on a new line with the text stated below. The following situations must be taken into consideration:
  1. The person name should consist of only printable characters from the standard ASCII character set (including spaces but excluding tabs and newlines). When invalid, print "Invalid name".
  2. Check for invalid dates and months; for example, 2010-11-31 or 2010-14-23. Adding

past dates is also invalid. Print “Invalid date” for these cases.

3. The <NAMESPEC> could be a full name, partial name or even a letter; character restrictions are the same as (1). The search must be case sensitive.
4. For *./birthdays –m "John Smith" 40*, if the exact name is not found, an exception should be raised. Print “Name not found”.

- A birthday, B, is “coming up in the next X days” from date Y if Y ≤ B ≤ Y+X. So *./birthdays –n 0* displays all and only birthdays today.
- If there are no birthdays in the specified time frame or matching the given <NAMESPEC>, the program should print “No birthdays found” to *stdout (not stderr as in previous cases)*. This is not considered an error, so an exception should not be raised.
- When birthdays are selected by date (in the without arguments, -n and -m cases and additionally the -u case if going for extra credit) they should be printed in chronological order. When they are selected by <NAMESPEC>, they should be printed in alphabetic order.
- There are never duplicate names. If the -a option is used twice for the same name the program should update the corresponding birthday instead of adding a new entry.

Grading Grid

- Points awarded for each feature are further divided as follows:
  1. The code compiles and runs according to specs – 50%
  2. Correctness of the code – 50%
- Extra credit points will only be awarded for code that runs according to specs.

Feature	**Points**
QT style programming and indentation	10
./birthdays without arguments	10
./birthdays -a <DATE> <NAME>	10
./birthdays -n <N>	8
./birthdays <NAMESPEC>	8
./birthdays -d <NAME>	8
./birthdays -d <DATE>	8
./birthdays -m <NAME> <N>	8
./birthdays –u	8
Correct output format	6
Correct output order	6
Errors treated correctly	10
Extra credit (no partial credit)	10
Total	**110**