Virginia Tech
Bradley Department of Electrical and Computer Engineering
ECE 3574: Applied Software Engineering ◆ Spring 2013

Homework 3

Submission Details

The solutions for this homework must be submitted as an electronic submission. Your submission will be a tarred and gzipped file containing the source files for the programming problem. These files must be in a directory entitled HW3_*LAST_FIRST* where LAST is your last or family name and FIRST is your first or given name. Your program will be run to evaluate its correctness, and the source code will be reviewed for adherence to Qt programming style. Your program must run on the latest version of Ubuntu and be compiled/built using the GNU C/C++ compiler and qmake. The following information must be included in your source files: your full name, your student ID number, your email address, class (ECE 3574), title of the assignment (Homework 3), and date of submission. You must submit the solutions using the "Submission" button on HW 3 assignment page on the Scholar class web site. The file must be given a name of the following form: HW3_*LAST_FIRST*.tar.gz where LAST and FIRST are as specified above. You can make multiple submissions. Paper and email submissions will not be accepted. All work must be submitted by the announced due date/time.

Honor Code

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

Homework 3:

Begin with the Dictionary class that you developed for HW 2. For this homework you are to develop a simple GUI that is an interface to this class and its ability to spell check and do a word frequency analysis of text files. You should start up the GUI with the command:

```
./dictionaryGUI
```

This GUI should have the following features.

1. From the menu bar you should be able to click on the "File" and be able to select a file to analyze from a selection window (we will go over this in class).
2. As with HW 2, your GUI should include a sub-window with the text in the file displayed. Again, words that are misspelled should be displayed as red in a scrollable window. The text in the window should be editable.
3. The menu bar should include a way to "Save" the text on in the window. That is, if you edit the text in the window, you should be able to save the changed text to a file of your choice.
4. The GUI should also be able to display a list of the most frequent words in another window (or sub-window). Note that the word frequencies should "reset" when load in a new text file.
5. There should be a way to interactively change the number of the words that are listed in the window described in requirement 3. The interactive way to do this is up to you, e.g., a slider, a text window you type in, etc. The window described in (3) should update as the number of words is changed.

6. There should be a way to interactively select the dictionary to be used, e.g., from American-English to British-English, or visa versa. The windows described in (2) and (3) should update if the dictionary used is changed. Again, the word counts should "reset" when you change dictionaries.
7. There should be a way to exit the application, e.g., a quit button, or exit from a menu on the main window, etc.

The overall design of this GUI is completely up to you. You may either implement your QUI completely in code, or you can use Qt Designer to design the layout.

Grading Policy

- QT style programming and indentation – 20%.
- The code complies and runs – 0%.
- The code satisfies the specifications – 60%.
- The design of your GUI interface – 20%.