Assignment 1: Design

January 26, 2018

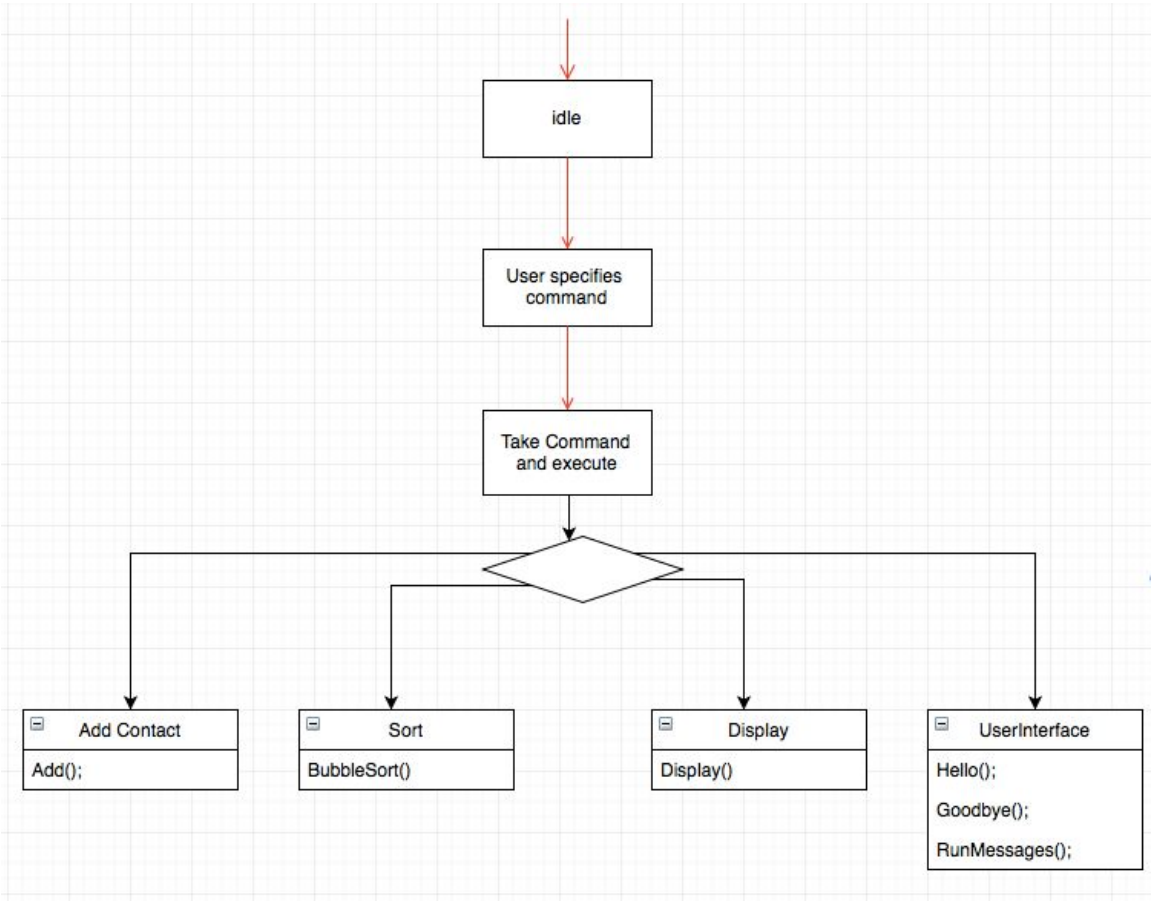
Winter 2018

Kristin Hsu, Vidhi Jain

Introduction

We are writing a command shell called rshell in C++ which will print a command prompt and read in a command on one line. The rshell consists of executables in the PATH and a list of zero or more arguments separated by spaces and possibly connected with a connector (; or II or &&). If a command is followed by a ";" connector, the following command always executes. However, if a command is followed by a "&&" connector, the following command only executes if the previous command was successful. Lastly, if a command is followed by a "||" connector, the following command only executes if the previous command fails.

Diagram



Classes

- User Interface
    - This class allows the user to type commands and begin or exit their contact list on rshell.
    - Hello(): This function displays an opening message (e.g. “Hello, welcome to your contact list”).
    - Goodbye(): This function displays a goodbye message to let the user know they have exited their contact list (e.g. “Goodbye”).
    - Exit(): This function allows the user to exit their contact list.
- Display
    - This class displays the user’s entire contact list in alphabetical order by first name.
    - Display(): This function displays the contact list, whether alphabetized or not.
- Sort
    - This class sorts the user’s contact list.
    - BubbleSort(): This function uses bubble sort to alphabetize the pre existing contact list.
- Add Contact
    - This class adds a contact to the contact list.
    - Add(): This function takes the user’s parameter and adds it to the contact list.

Coding strategy

We will split the classes between us. Kristin will implement the Add function and Sort classes and Vidhi will implement the Display and UserInterface classes. We will not pair up for any classes since we feel we are capable of completing each class on our own.

Roadblocks

- Lack of experience with shell and git
    - Due to our unfamiliarity with shell and git, a significant amount of time will be spent on learning and understanding the different commands. Since a proper and

thorough understanding of git and shell is needed to be successful with this project, we will spend time reading and watching tutorials

- Multiple classes
    - With multiple classes interacting with each other, one or multiple classes may depend on one another. This means that testing classes that depend on one another may be difficult and done near the end, since we would have to have all these classes finished in order to start testing. In order to prevent having to test our code at the last minute, we have designed a schedule as to when each set of classes needs to be done in order for testing to start.
- Scheduling Conflicts
    - Due to our personal schedules, whether it be classes or other activities, scheduling time to work together on this project may be a possible challenge. In hopes to try and resolve this possible roadblock, we have exchanged our schedules with one another and have set plans on certain meeting times to discuss our progress on our assigned parts.