Program 2: Mobile Phone

Michael Lawrence

CSC 331 - 002

September 27, 2019

1. Problem Statement

The purpose of this program is to create a class that inherits from the Mobile class given in this assignment and make it more robust. It should be able to interpret phone numbers in different formats and it should sort the contact list alphabetically. The program should also have a front-end that displays all relevant information and allows user to create and enter phone information.

1. Input and Output

The first input that will be required for this program will be the part of initializing an object of the Mobile class. The class will require the model (String), phone number (String) and serial number (String). If this information is not entered, it will use default values. The user can then give the program contact information including phone number (String) and name (string), and the user can also give the program a sim card number. The program will collect and output this information in proper format and will dial (print into console) the contact entered.

1. Usage

The program should be compiled using Visual Studio 2017 by opening PhoneForm.sln solution file. To run the program, you can use either Visual Studio’s run system or you can open the executable located at “Program2\_MichaelL\Phone Form\bin\Debug\PhoneForm.exe.”

1. Theory
   1. Information about the concepts being attempted in the lab

The goal of this lab is to experiment using classes and methods to cut down on the use of repeated code. This is especially important in programming due to the hard-to-read nature of programs. Why write repeated code you used before in the program or when someone else has written the code? This lab will practice the writer of using other’s code and manipulating it rather than starting from scratch using inheritance.

* 1. Algorithm description

MyMobile class will inherit the features from the given Mobile class and implement more robust inputs for the Mobile class and a alphabetical sort for the contacts list. It will do this with four additional methods: CheckFormat(), addContact(), sortContactList() and viewINFO() with appropriate get and set methods.

CheckFormat(String) : String “This will convert phone number string into XXX-XXX-XXXX format or throw an error.”

addContact() : void “calls checkFormat(), catches exceptions and adds Contact into list.”

sortContactList() : void “sorts ContactList[] in alphabetical order by name.”

viewINFO() : void “Displays object information into console.”

* 1. Error indications

The program’s goal is to be as robust as possible, with as many errors dealt with by the program as possible, but some errors cannot be avoided. Without considering errors such as entering the wrong parameter data type in a method and other compile errors, the list is short.

C1: Phone number is entered into CheckFormat() without containing 10 digits. The program will throw the MobilePhoneExceptionMessage exception in that case.

1. Correctness
   1. Correctness Argument

My algorithm provides a clean expansion to the Mobile class by overriding the addContact() method with a more robust input and providing a method to allow other programmers to check the correct format without adding contact. In terms of expanding the Mobile class correctly, MyMobile throws less errors and adds further functionality.

* 1. Correctness Testing

The following tests were used to evaluate and fix any issues with MyMobile class.

1. Sending random character strings with 10 digits engrained within to the addContact and checkFormat methods. This should be sufficient in removing other conventional phone number formats.
2. Sorting a list of 0 contacts, 1 contact, and 5 contacts with the 5 contact list including 2 same name contacts and 2 slightly different names.
3. Self-Evaluation
   1. Known Bugs

The program currently does not check the phone number entered during the creation of a My MyMobile object, adding a contact with the phone number entered might throw an error.

* 1. Possible Improvements

The program includes some algorithm inefficiencies. Once the proper format is found through the MyMobile class, it is sent to be rechecked by the Mobile class, cutting the recheck process will speed it up slightly. Another is adding more Sim Card functionality.