|  |  |  |
| --- | --- | --- |
| American University of Sharjah  College of Engineering  Department of Computer Science & Engineering  P. O. Box 26666, Sharjah, UAE |  | **Instructors:** Dr. Michel Pasquier  **Lab Instructor:** Praveena Kolli  **Office:** EB2-126  **Phone**: 971-6-5152352  **e-mail**: pkolli@aus.edu  **Semester**: Spring 2020 |

**CMP 305 L – Data Structures and Algorithms**

**Lab. Assignment 2 - STL**

***Objectives:***

* To practice using C++ STL containers and algorithms
* To make use of STL vector and list classes
* To write generic C++ code using iterators

**Note:**

***Lab:*** Exercises 1 (10 marks)

***Bonus and optional*:** Exercise 2 (2 marks)

***Exercise 1:***

The file ‘persons.txt’ (provided) contains the details of several persons who are part of a team. Each line contains the information about one person and consists of space-separated fields i.e., two strings that are the first name and the last name, followed by the telephone number.

For example, the first two lines might contain:

Ahmed Badawi 508998024

Syes AlGhabra 6480235

You are to develop a program that allows users to add, search, delete and view persons’ details.

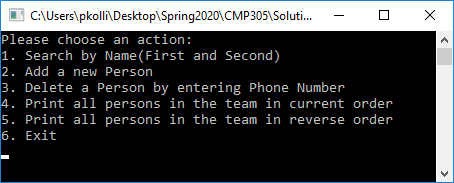
The Person class is provided to you with the following data and function members:

1. Three data members: first name, last name, phone number – as described above.
2. A constructor with arguments (first name, last name and phone number).
3. Getter functions for all data members.
4. An overloaded output stream operator << that prints all three data members, separated by a space character.

Your main() function should create a *team* which is a container (either a **vector** or a **list** object) that stores several *persons* (cf. the provided driver code). Note that you should be able to use either a vector or list interchangeably; all your code should be exactly the same regardless.

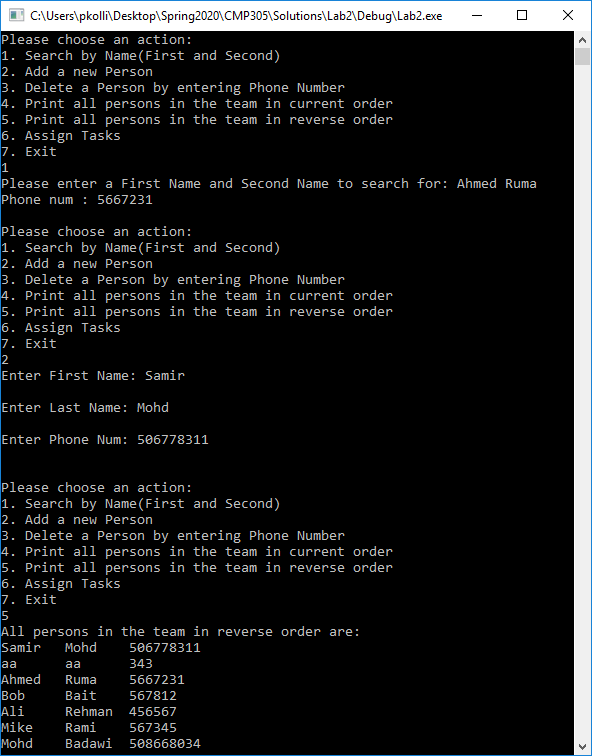
Recall that using C++ STL containers, iterators, and algorithms allows writing code that works in all cases i.e., regardless of which container is used or what kind of data it contains.

Your program should read all persons’ details from the input file and present a text menu, as illustrated hereafter, that allows users to search, add, or delete a person to/from the team. Note that your program should repeatedly present the menu until the user chooses to exit.



* For options #1, your program should ask the user to type in a name first and last name, then search for a matching person. It will print the phone number, if found, or an error message if not.
* Option #2 will prompt the user for the three fields and add a new person to the team.
* Option #3 will ask the user to provide a phone number then search for a match. It will delete the person from the team, if found, or print an error message if not.
* Option #4 will print all persons in the team in the current order.
* Option #5 will print all persons in the team in **reverse**.
* Option #6 will cause the program to terminate.

*Sample Output:*



\* You may use the provided main function in driver.cpp file to complete.

**Bonus and optional**

**Exercise 2:**

You are assigning tasks to your team. There are four tasks to be **rotated** among team members. Tasks are: fetching water, collecting firewood, cooking, and cleaning up. You are to add an option (#6) to the program in part (a) that prompts the user for the desired number of working days and prints a schedule of daily duties for the team.

The “Task” class is provided to you with the following data and function members:

1. Three data members: id, name, duration, – as described above.
2. A constructor with arguments (id, name and duration)
3. Getter functions for all data members.
4. An operator overload function for extraction << that prints three data members, separated by a space character.

Create a *tasks* variable which is a container (either a **vector** or a **list** object). Just like in part (a), all your code should work the same regardless.

The program must read tasks from an input file ‘tasks.txt’ (provided), which may look like the example hereafter. Each line represents a task, with ID, name, and duration (in hours).

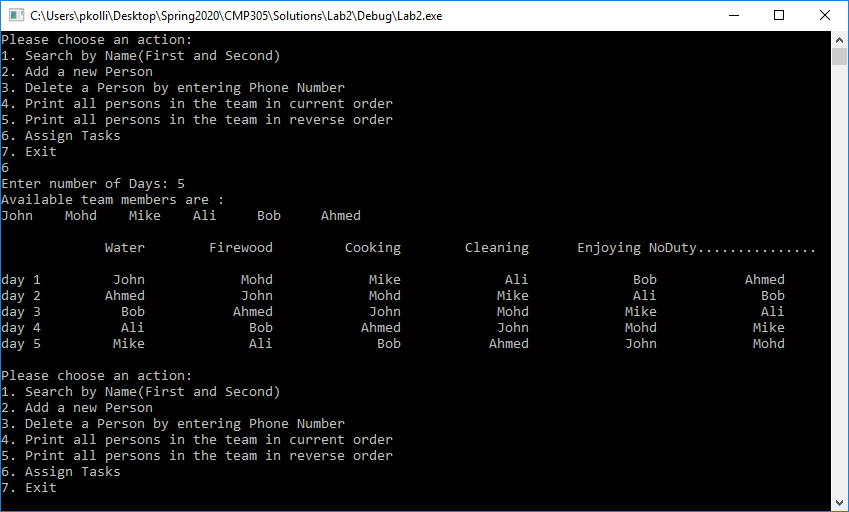
100 Water 1

200 Firewood 2

300 Cooking 3

400 Cleaning 2

*Sample Output:*



***Notes:***

* Accessing vector/list entries should be done using **iterators**, in a generic way, i.e.:

for ( auto it = team.begin(); it != team.end(); ++it ) { /\* your code \*/ }

* You should use generic functions from the <algorithm> library e.g., rotate (in part b), etc.

***References to C++* libraries*:***

String: <http://www.cplusplus.com/reference/string/>

Vector: <http://www.cplusplus.com/reference/vector/vector/>

List: <http://www.cplusplus.com/reference/list/list/>

Algorithm: <http://www.cplusplus.com/reference/algorithm/>