2019

Data Structure and Algorithm- DS232L

(Semester Project)

This document contains general information related to the semester project given to session 2017 in semester spring 2019 EE UET Lahore.

8/4/2019

Data Structure and Algorithms

Semester Spring 2019

Department of Electrical Engineering

University of Engineering and Technology Lahore

Instructors: Sana Noor & Maham Fatima

1. Problem Statement

In this project, you are asked to practise the usage of data structure to handle the contact book problem. You are supposed to provide a efficient solution to handle this problem by implementing Create, Retrieve, Update, Delete and Sorting operations with least running time.

2. Problem Detail

A contact book contains the data you need when contacting your friends. The data may consist of various attributes such as first name, last name, home number age, phone number, email, company name and birthday. See Table I, for an example. The data should be stored in some data structure instead of storing as plain text to allow efficient operation. To simplify the task, assume that each element in some of the attribute is unique. For instance in our example in Table I, we have assumed that each phone number is unique.

First Name	Last Name	Phone Number
Harry	Potter	0995435355
Hermione	Granger	0958416991
Ron	Weasley	0901772105
Ginny	Weasley	0926925019
Draco	Malfoy	0990224661
Albus	Dumbledore	0911276562
Tom	Riddle	0975867725

TABLE.I An example of contact book

3. Functions of contact book

Your program should implement all function on at least five attributes **(First name, Last name, Phone number, Home number and Company name)** where home number and company name is optional. Some operations associated to this problem:

a. Create operation shall create and initialize the contact book (Cbook).

b. Retrieve operation shall search for a given attribute of friend inside the Cbook, find another (or all) attribute of that friend and print attribute (or all) of that friend.

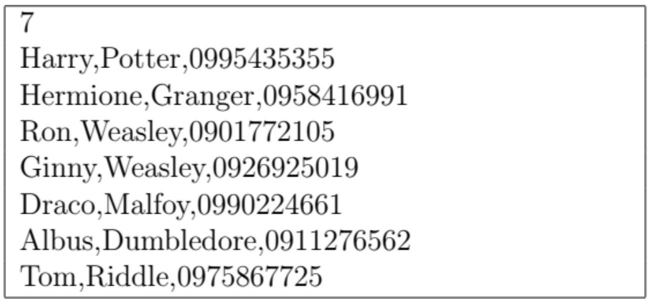
c. Insertion operation shall add new contact in Cbook.

d. Deletion operation shall delete an existing friend from your contact book by given a specific attribute of that friend.

e. Edit operation shall update an attribute of a friend by given a specific attribute of a friend.

f. Sorting: List the friends sorted by a specific attribute. This function should work on all attributes, which means that we can list the friends sorted by any attribute we want.

g. Load Read data from a file which contains data of multiple friends. See Figure 1 as a file example. This function may be used at the time the program starts, and the function can be thought as a sequence of multiple insertions. However, we encourage you to think about if there is some way to load data faster. Note that you can define your own file format, which means that your program does not need to be stored according to the format of Figure 1.



```
7
Harry,Potter,0995435355
Hermione,Granger,0958416991
Ron,Weasley,0901772105
Ginny,Weasley,0926925019
Draco,Malfoy,0990224661
Albus,Dumbledore,0911276562
Tom,Riddle,0975867725
```

Figure: 1 Example of file content of table I

4. Report Requirement

Following are the essential requirements of this project, on which you shall be graded;

- ✓ **Page 01:** Up to how much extent were you able to understand the problem statement? **(*Simply write a few lines, describing the problem statement in your words)***
- ✓ **Page 02:** Propose your solution for the given problem statement by choosing an appropriate data structure type. ***(Feel free to use any data structure's type (ADT) but give your reason to choose)***
- ✓ **Page 03:** Design your algorithm for that particular problem. ***(Show your design flow diagram)***
- ✓ **Page 04 & 05:** Time complexity of your proposed algorithm. ***(Show your hand written work and your time complexity shall be in asymptotic notation)***
  - o *Calculate the time using clock_t*
  - o *Compute the time complexity by hand in asymptotic notation*
- ✓ **Page 06:** Results of your algorithm under the subject of various types of inputs.

General Instruction:

- ❖ Your implementation should not be that much hard to understand, try to use all the information that has been covered in your theory and in your lab.
- ❖ Be unique and efficient while coding.
- ❖ Highest score shall be awarded on how much your proposed solution is meeting the entire requirement and is efficient in its time complexity.
- ❖ Please don't try to copy someone's code as plagiarism these days is not that hard to find.
- ❖ Grading policy is given below:

Functions	**Functions Grading**
Retrieve Insertion Deletion Edit	4%
Create Sorting Load	6%
Other	
Proposal O/P Efficiency Report	5%
Total	**15%**