Phonebook:

Write a program which maintains a phonebook with following specifications:

A- After running the program, it must enter an infinity loop which provides the below menu:
   a. Create a new entry.
   b. Delete an entry.
   c. Find an entry and show its information.
   d. Save the whole phonebook in a file.
   e. Exit.

B- If the user chooses (a), he must be asked to enter the related information (phone number, contact name and group). Refer to below description for **contact_info** structure. Appropriate actions must be undertaken to modify the linked list and the lookup table. Everything, even if entered in small letters, must be stored in CAPITAL letters.

C- If the user chooses (b), he must be asked for the name of the person whose entry is to be deleted. After receiving the name if it is in the list, the information related to the entry must be printed out and the user must be asked to confirm entry removal. If he confirms, the entry must be removed and appropriate actions must be under taken to modify the linked list and the lookup table.

D- If the user chooses (c), he must be asked for the name of the person. Then the lookup table must be consulted. The portion of the linked list starting from the address received from the lookup table must be traversed to find the entry. A check must be provided for the case the entry does not exist and the search must be terminated upon reaching to an entry starting with the next letter.

E- If the user chooses (d), the whole phone book must be saved in a file called "**phone_book.txt"**. On the first line the number of entries must be saved. Entries must be separated from each other and from the first line by a line of twenty '*'s (**********). For each entry, the name must be stored on the first line, the phone number on the second line and the group on the third line.

F- If the user chooses (e), the phone book must be stored in a file using the instruction given in (E) and the program must be terminated.

To carry out the tasks defined above, following requirements MUST be followed:

1- Every entry in the phonebook has following information:
   a. Phone number.
   b. Contact name (**CAPITAL** letters).
   c. Contact Group (*one of*: **Family**, **Friends**, **Colleagues**, **VIP** and **Others**).

   which are to be taken care of using a structure called **contact_info**.

2- The phonebook is to be maintained as a linked list. A linked list is a chain of items which have links to each other. Figure (1) shows a typical linked list. Note that there is a beginning to the list (to which no arrow points) and an end to the list (from which no arrow is emanating).

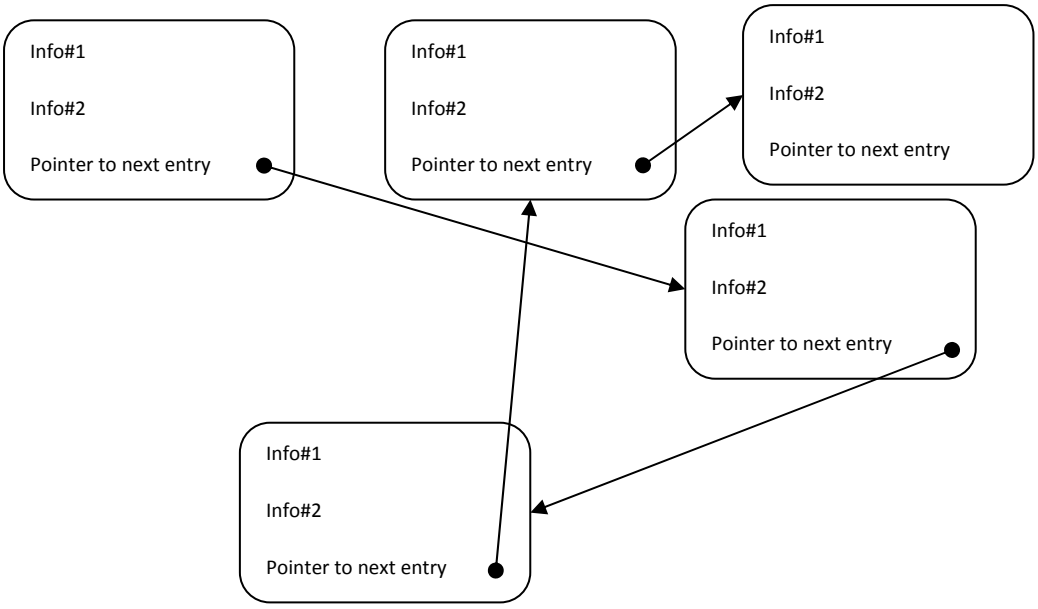

Figure 1 – A simple linked list

Therefore, added to information listed in item#1, the **contact_info** structure should be equipped with a pointer to **contact_info** which holds the address of the next element in the list. Last element has a pointer with the value of zero (pointing nowhere).

3- There must be variable which holds the number of entries in the phonebook called **N_entries**. It must be updated whenever the number of entries available in the list changes.

4- The entries in the list must be maintained in the alphabetic ascending order (A to Z) which means if there are three entries with contact names BAHMANI, RAHMATI and ALIZADEH the list must look like Figure (2).

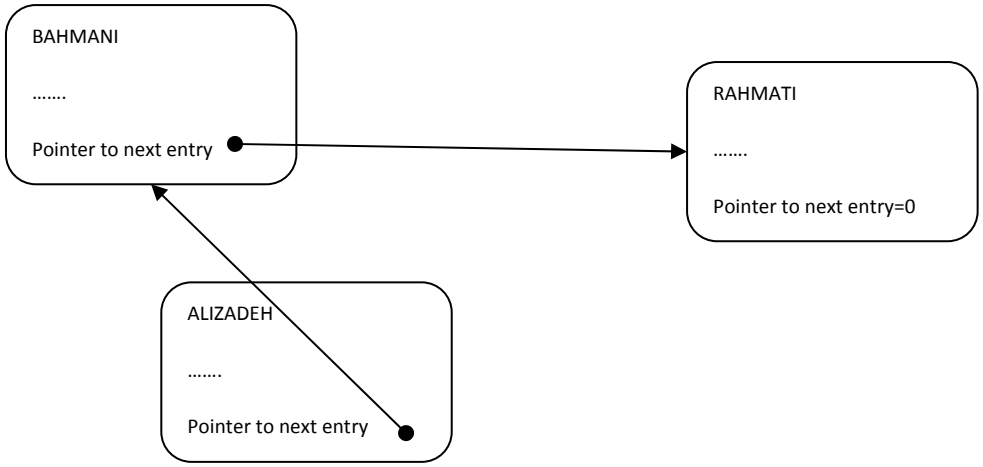

Figure 2 – Sample linked list.

5- There must be a one-dimensional 26 element array, called **look_up**, whose elements are pointers to **contact_info** structure. In ***i***th element of the **look_up ,** pointer to the first phone book entry beginning with ***i***th letter must be stored (A is the zeroth letter, B is the first letter, C is the second letter, …, Z is the 25^{th} letter). If no entry with that letter is recorded, zero must be

used. For example, if there are just two entries beginning with B (BAHRAMI and BEIGI), **look_up[1]** must hold the pointer to the **contact_info** structure corresponding to the entry BAHRAMI (since "BAHRAMI"< "BEIGI"). Refer to Figure (3).

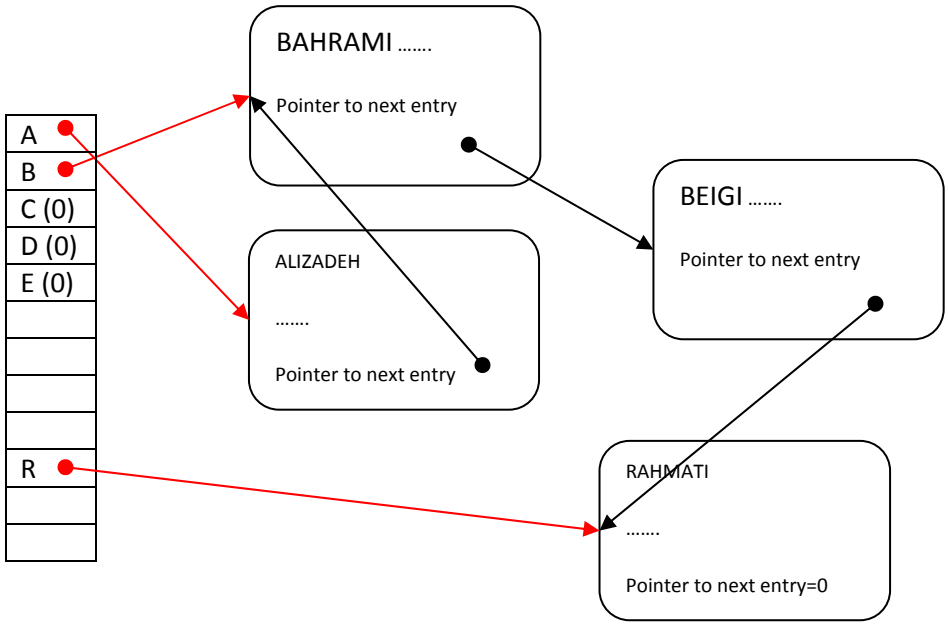

Figure 3 – Lookup table for the phonebook linked list (red arrows).

6- Every time a new entry is introduced, the **look_up** array must be consulted to find out where, in the linked list, is the suitable place to put the entry. If an entry for BASTANI is to be created, it must be inserted so that, there is a pointer from BAHMANI to BASTANI and a pointer from BASTANI to BEIGI (Figure 4). Note that if the name was BAGHERI the **look_up[1]** needed modification, too; because, "BAGHERI" < "BAHMANI" and **look_up[1]** needed to be modified to hold the pointer to the entry corresponding to BAGHERI.

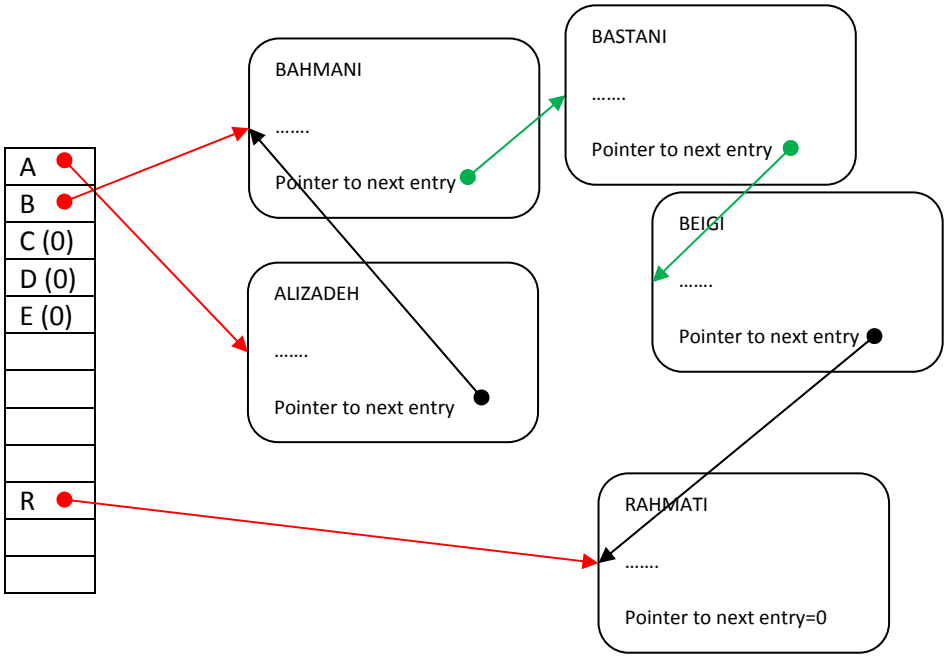

Figure 4 – Adding an element to the phonebook (BASTANI is added with the corresponding pointer modification shown in green).

NOTEs:

1- **This is a design project, therefore, you should decide about anything not specified explicitly herein and your work will be evaluated upon the choice you make.**
2- **The specifications provided herein must be followed as minimum design requirements.**
3- **You must be able to defend your choices and whatever you submit. Therefore, you had better design and write the code by yourself!**
4- **The penalty paid for dishonesty was crystal clear from the beginning of the semester. Accepting the risk is a matter of your choice.**