**Address Book**

**Sayedshahram Pezeshkitoosi**

**13.12.2013**

**Table of Content:**

1- General Description

2- Problem analysis

3- The structure of solution

4- Detailed description of subprograms

4.1 – sliceList (List) :

4.2 – joinList () :

4.3 – sortFile ():

4.4 – listTofile():

4.5 – Delete()

4.6– Search()

4.7 – showList()

4.8 – getFullName()

4.9 – getPhone()

4.10 – getAddress()

4.11- getMail()

4.12- Add()

4.13 – Edit()

4.14- Body of the program

**1- General Description:**

a simple database program should have ability to store data in the disk and have some features like inserting new items to this database, deleting Item(s) from database, editing an Item, search the database and browsing all the existing data which are available in that database. In order to create this database and provide these features we decided to write a simple phone address book. In this case we have a file which is our database and stores phone addresses (name phone numbers, addresses and e-mail addresses) and have the feature that I mentioned. In this program we can ask user to add a new contact, delete an existing contact, search and find a special contact, edit a contact and finally see all the contacts on output.

**2- Problem Analysis:**

we should consider that we can not read just a specification part of the file and we could not edit this file directly. We need to bring this file to a list and do what ever that we want (search, edit, add and delete). Then we must rewrite this list to the file.

The other problem is tat when we want to add a new record to our address book. This contact for example, may has not any email address or may has some email address. There is no limitation for the emails, addresses or phone number for a person.

The other problem is about search. We should search not only the names but also we should search all other records for a contacts. Then we have list of search and user can select which field he wants to select.

The other challenges was about the format that data should be saved in the disk. We must save data and record them in a format to read easily and separate each line to the various records. Also, in order to save data on disk we need to join different records and add needed separators between them.

Error handling in this program is one of the important issue. User could not have permit to input invalid data in the database and the program should answer the proper response when user tries to input invalid data.

**3- The Structure of The Solution:**

in order to read from file and write on it, we bring all the file into a list and do what ever that we want. For example, in list we delete, search, add new account and then we sort this list and finally take it back to the disk and save it on it. In fact, this process is implemented for each order. For example if we want to add a new contact or search a record this process will do for each of them. We defined a procedures to write to database (Address book.txt): ' listTofile' which read from list and overwrite it to the file.

W considered that each line in our database is the information and record about a contact. Each contact have four fields: 1- full name 2-telephone numbers 3- Addresses 4- e-mail address. Full name must be one and must be imputed, otherwise the contact is not created. However the other fields can have no record or as many as user wants.

These fields are separated by “%” sign. If we have more than on telephone number, these numbers are separated by “?” sign. If we have more than one address for each contact, these addresses are separated by “|”sign. If we have more than one email addresses, these email addresses are separated by “;” sing. When user wants to add a new contact to this database (address book.txt) by 'add' function, these separators are added to the file automatically. Each line of database can be something like this:

Sara Kamali%0449248765?0558467345%Emmauksenkatu 9A 82, Turku%sara@utu.fi\n

On the other hand, we use a function (sliceList) to slice our database by these separators and put it into a list of lists. In fact this function receive a list which each item of it is a line of our database, and the slice each item in point of separators. For example this function converts list1 to list2.

List1=['Sara Kamali%0449248765?0558467345%Emmauksenkatu 9A 82, Turku%sara@utu.fi\n']

list2=[[“Sara Kamali”,['0449248765','0558467345'],['Emmauksenkatu 9A 82, Turku'],[['sara@utu.fi](mailto:'sara@utu.fi)']]

Also, we have another function (joinList) which convert and join list2 to list one. This function is used whenever that we want o write from list to file.

For handling error, we limit user for inputting data. For example name when ever that we ask for a number, the program check if the input data is integer and if it is in range. Also we have some limitation for inputting some data. For instance email must be in this format: [xxxxx@xxxx.xxx](mailto:xxxxx@xxxx.xxx) .

In order to solve the problems we defined various function. These functions have their own characteristics which I explain them completely and explain about their tasks.

**4**. **Detailed description of subprograms**

**4.1- sliceList(List):**

This function is used to slice each line of database. In fact this function get a list which each record of this list is a line of database. Then this function separates this list to a list of lists and separates each line by defined separator.

This subprogram use one parameter. The name of this parameter is 'List'. This parameter is a list which each record of them is a line of database.

The return value of this subprogram is a list which is separated by some signs ('%','?','|'and ,).

**4.2 – joinList(List):**

This function is used to mixes sliced data of each contact. We want to mix theses data to be in a format that we want to store it in the database in disk.

This function receive a parameter called 'List', which is a sliced list and return a list which each record of that can be a line of the database.

**4.3 – listTofile(List,addressBookFile):**

This procedure is used to read all the Item in the list and overwrite it on the database. We need this procedure whenever that we made some changes about our database in the list and we want to overwrite it on the database in disk.

This procedure have two parameters: 1- 'List' which is a list that have modified record and 2- 'addressBookFile' which is the file that we want to overwrite it.

**4.4 – sortFile():**

this procedure is used to sort the lines in database based on the name of the contact. In fact this function does not have any parameter. It just read the database (address book.txt) and brings it to a list and then sorted this list by the name of contact and then rewrite this sorted list to the file.

**4.5 – search (searchStr):**

this function is used to search and find a string in our file. The function use a parameter which name is 'searchStr. searchStr is a string that the function wants to search it through the file 'address book.txt'. Each line that have this string in our database is added to a list called 'searchList'.if search function find something , it use sliceList(List) function to slice data and put it in the searchList list. Then search list is returned as a value of this function.

**4.6 – delete():**

this subprogram is a procedure which does not have any argument. This procedure ask user a string (delStr) and uses 'search' function to find the contacts. Then ask user to select which contact does he want to delete and then t deletes this contact from the database.

In this procedure variable 'delItem' is a list that we want to delete. Variable 'delSearchList' is a list that holds the search function results. The variable delStr is a string that user input and procedure pass it to the search() function.

**4.7 – showList(List):**

this procedure is used to show a sliced list in output in a format that is more user friendly. For contact name we considered 32 character; contact name is showed in the middle of this 32 characters. For phone number 30 character, for address 50 character and for email address 30 character. The records are put in a table.

The argument for this procedure is 'List' which is a list that we want to show it in output. The variable are: 1-nameCounterwhich is a counter to count the number of names in the address book. 2-PhoneCounter which is a counter to count the number of phones in a contact. 3-addressCounter which is a counter to count the number of addresses in a contact. 4- emailCounter which is a counter to count the number of emails in a contact.

**4.8. - getFullName():**

this subprogram is a function that is used in add() and edit() subprogram and is used to get user a name for new contact. When user input a string as a name he is not allowed to use '%','?','|' and ';' characters.

Values in this subprogram are: 1- especial\_chars which is a list of mentioned characters. These characters are used for separating different field of a contacts in file. So they are reserved for that purpose. 2- while\_flag which is a flag for while loop. This flag use in the following while loop to keep user in this function in order to enter a proper name or type cancel to exit. 3- full\_name which is a string that user input as the contact's name. 4- validity\_flag which check if the input data is valid and permitted.

this function returns full\_name as a string as the contact's name.

**4.9- getPhone():**

this subprogram is a function that is used in add() and edit() subprogram and is used to get user phone numbers. The user can input "1","2","3","4","5","6","7","8","9","0","#","\*","p","P" characters as an input for telephone number.

Values in this subprogram are: 1- valid\_phone\_char is a list that contains all possible characters in a phone numbers. 2- while\_flag which is a flag for while loop. This flag use in the following while loop to keep user in this function in order to enter a proper name or type cancel to exit. 3- phone\_input which is a string that holds inputting user phone number. 4- validity\_flag which check if the input data is valid and permitted.

this function returns phone\_input as a string which is phone number of a contact.

**4.10 – getAddress():**

this subprogram is a function that is used in add() and edit() subprogram and is used to get user an address for a new contact. When user input a string as a name he is not allowed to use '%','?','|' and ';' characters.

values in this subprogram are: 1- especial\_chars which is a list of mentioned characters. These characters are used for separating different field of a contacts in file. So they are reserved for that purpose. 2- while\_flag which is a flag for while loop. This flag use in the following while loop to keep user in this function in order to enter a proper name or type cancel to exit. 3- full\_name which is a string that user input as the contact's name. 4- validity\_flag which check if the input data is valid and permitted.

this function returns address\_input as a string as the contact's address.

**4.11 – getEmail():**

this subprogram is a function that is used in add() and edit() subprogram and is used to get user an email address for a new contact. User must put a email in this format : [xxxxx@xxxx.xxx](mailto:xxxxx@xxxx.xxx) . We used a regular expression (“r"^[A-Za-z0-9\.\+\_-]+@[A-Za-z0-9\.\_-]+\.[a-zA-Z]\*$"”) to check email address.

values in this subprogram are: 1- while\_flag which is a flag for while loop. This flag use in the following while loop to keep user in this function in order to enter a proper address or type cancel to exit. 2- email\_input which is a string that user input as the contact's email address. 3- validity\_flag which check if the input data is valid and permitted.

this function returns email\_input which is string and is used as the contact's email address.

**4.12- add():**

This procedure add a new contact to "address book.txt" file by using getFullname(), getPhone(), getAddress() and getEmail() functions. User can input one name and input s many number, address and email for each contact.

values in this subprogram are: 1- record:record is a string which contains one contact as a strirng when a new contact add to addressbook such that name, phones, addresses and Emails separate by % phones separate by ? addresses separate by | and emails separate by ';' . 2- phone :phone is an string which contains phones of one contact when a new contact add and each phone number separate by '?' . 3- address: address is an string which contains addresses of one contact when a new contact add and each address separate by '|' . 4- email:email is an string which contains emails of one contact when a new contact add and each email separate by ';' .

this procedure add a new line when we put a new Name and asks user for the phone number, address and email for that name.

**4.13 – edit():**

this procedure is to edit a contact. In fact this procedure is a mix of all the above subprogram and use all of them. The edit procedure algorithm is in this form: at first it asks user to say which contact he wants to edit. User input a string. After that edit procedure use search function to find related contacts. Then user must select the exact contacts that he wants to edit from search list. Edit procedure consider the selected contact and makes a copy of it and holds the copy in a list.

Then it delete the selected contact from database by delete procedure. User can edit each field of selected contact in the list that stored in memory. After implementing all the changes in the contact, the edit function add the edited contact to the database by add procedure. Add procedure will sort the database by sort function.

values in this subprogram are: 1- editItem:editItem is a list that contains the contact which user wants to edit and all changes effect on this list. 2- tempEditItem: tempEditItem is a list that contains the contact which is joined by joinList() that user wants to edit but no changes are done in it.(the copy of the contact) 3- editSearchList: editSearchList is a list that includes all contacts that found in "address book.txt" and contains the string which user input. 4- editStr: editStr is a string that user want to search in "address book.txt" to edit it.. 5- editSelection: a string that user input which select the exact contact that he wants to edit from search list.

**4.14- Body of Program (Main):**

the body of the program shows user an output and ask them to select which features he wants to do. 1- adding a new contact. 2- delete. 3- edit. 4- search, 5- show all the contact 6- quit. Based on the user selection the related sub program is called.