MIDDLE EAST TECHNICAL UNIVERSITY, NORTHERN CYPRUS CAMPUS
CNG213 Data Structures – Programming Assignment 1

Date handed out: **Monday 30 October 2017**
Date submission due: **Monday 13 November 2017 23:55**

FriendBook

This assignment aims to help you practice linked list data structure and basic linked list operations. Your main task in this assignment is to create a C program that will allow you manage your friend list which we call FriendBook. Your FriendBook will include the list of your friends with their details (first name, last name, gender and their birthday) and will help you manage your list of friends, where you can add a friend, find the details of a friend, and block a friend from your friend list. The idea of FriendBook is similar to how Facebook manages your list of friends.

When you run the C program, your FriendBook will first create a linked list of your friends which is initialised by reading the details of your friends from an external text file and then some operations will be provided for you to manage your FriendBook. Therefore, your program needs to support the following operations.

- **Reading your friends from an external file and initialising your friend list**: The program needs to be invoked with a command line argument which is the name of the file that includes your friend details. If this command line argument is not provided then your program needs to ask the user to enter the name of the text file. The file should include the first names, last names, genders and date of birth (dd/mm/yyyy) of your friends, separated by a semicolon (;). We assume that there is no duplicates in the first names and surnames. An example file is shown below that contains there friends:

  ```
  Tiffany;Evans Smith;F;22/01/1989;
  Alex;Williams;M;23/06/1988;
  Clay;Bristol;F;30/12/1989;
  ```

  The program will read your friends from the file and use a linked-list to store your friends' details. Please note that you cannot make any assumptions about the number of your friends and there is no upper limit for the number of friends.

  Your friends may not be sorted in an external file, however your FriendBook which will be created by the program should be sorted by the **first names in ascending orde**r. Therefore, when you insert a new friend to the list, you need find its position and then insert it. The program should not add a new friend at the end of the list and then sort the list.

- **Inserting a new friend:** After the initialisation of your FriendBook with an external file, you may want to insert a new friend to your friend list. Therefore, the program should ask you the required details (first name, last name, gender, and date of birth) for your new friend and add it in the

appropriate position in your friend list. Please note that your friend list should be sorted by the first names in ascending order.

- **Printing your friend list**: The program should traverse the list of your friends and print your friends on the screen with their details.

- **Searching for a friend in your friend list**: The program should allow you to search for a friend with his/her first name. Please note your search algorithm should have the O(logn) complexity.

- **Blocking a friend:** The program should allow you to block your friends. First of all, you need to enter the first name of your friend to block him/her, then the program will find the friend, then delete it from the list of your friends, and then add your blocked friend to the list of your blocked friends. Therefore, you need to have another linked list to keep the list of your blocked friends.

- **Printing your blocked friends:** The program should traverse the list of your blocked friends and print them on the screen with their details.

Programming Requirements

When you write this program, you need to consider the data structure that will be necessary. Clearly, two linked-lists will be required: one for the list of your current friends and another one for the list of your blocked friends. Therefore, it is recommended that you implement your basic linked list operations first (add/delete/search, etc). By using these basic operations on linked lists, you can implement other required operations.

In order to achieve the steps you need to have the following functions. Please strictly follow the requirements of the functions given below! In addition to these functions, you can write some helper/auxiliary functions.

Function	Explanation	Input	Output
`initialiseFriendBook`	To create and initialise a list of friends by reading the friends from an external file	File name	A list of friends
`insertFriend`	To insert a new friend in an appropriate position in the list of friends by asking the required details	A list of friends	-
`printFriends`	To display the list of the friends with their details	A list of friends	-
`searchFriend`	To search for a specific friend with given first name, and then print his/her details – O(logn)	A list of friends First name of the friend	-

`blockFriend`	To delete a friend from the the list of friends and add it to the blocked friend list	A list of friends A list of blocked friends	
`printBlockedFriends`	To display the list of blocked friends with their details	A list of blocked friends	-

Grading:

Your program will be graded as follows:

Grading Point	Mark (out of 100)
Structures to represent a list of friends	5
Main function to control and coordinate the commands	10
`initialiseFriendBook`	20
`insertFriend`	20
`printFriends`	5
`searchFriend`	15
`blockFriend`	20
`printBlockedFriends`	5

Note: Remember to have good programming style (Appropriate comments, variable names, formulation of selection statements and loops, reusability, extensibility etc.). Each of the items above will include10% for good programming style.

Sample Run:

```
C:>FriendBook friends.txt
Your FriendBook has been created.

(1) Insert a new friend
(2) Print your friends
(3) Search for your friend
(4) Block your friend
(5) Print your blocked friend
(6) Exit
Enter your option: 1

Enter the details of your friend to insert.
First name: Olivia
Second name: Carter
Gender: F
Birth Year: 22/11/1990
Your friend has been added to your friend list.

(1) Insert a new friend
(2) Print your friends
(3) Search for your friend
(4) Block your friend
(5) Print your blocked friend
(6) Exit
Enter your option: 2
```

```
Your friends are listed below.
Name         Surname            Gender       Birth Year
Alex         Williams           M            23/06/1988
Clay         Bristol            F            30/12/1989
Olivia       Carter             F            22/11/1990
Tiffany      Evans Smith        F            22/01/1989

(1) Insert a new friend
(2) Print your friends
(3) Search for your friend
(4) Block your friend
(5) Print your blocked friend
(6) Exit
Enter your option: 3

Enter the first name of your friend to search.
First name: Alex
Your friend is in your friend list.
Name         Surname            Gender       Birth Year
Alex         Williams           M            23/06/1988

(1) Insert a new friend
(2) Print your friends
(3) Search for your friend
(4) Block your friend
(5) Print your blocked friend
(6) Exit
Enter your option: 4

Enter the first name of your friend to block.
First name: Clay
Your friend Clay has been blocked.

(1) Insert a new friend
(2) Print your friends
(3) Search for your friend
(4) Block your friend
(5) Print your blocked friend
(6) Exit
Enter your option: 5

Your blocked friends are listed below.
Name         Surname            Gender       Birth Year
Clay         Bristol            F            30/12/1989

(1) Insert a new friend
(2) Print your friends
(3) Search for your friend
(4) Block your friend
(5) Print your blocked friend
(6) Exit
```

```
Enter your option: 2

Your friends are listed below.
Name         Surname            Gender       Birth Year
Alex         Williams           M            23/06/1988
Olivia       Carter             F            22/11/1990
Tiffany      Evans Smith        F            22/01/1989

(1) Insert a new friend
(2) Print your friends
(3) Search for your friend
(4) Block your friend
(5) Print your blocked friend
(6) Exit
Enter your option: 6

Goodbye!
```