EGRE 246 Advanced Engineering Programming Using C++
Homework #3 – Linked Lists

This homework must be your own (individual) work as defined in the course syllabus and discussed in class.

1) Write a C program to read in and manipulate a list of account holders and their account balances. The program should start by reading the list of current accounts from a file. The program should prompt the user for the name of the input file. Each line in the file is an entry for one account holder. It consists of the account holder's last name, first name, account number, and their account balance. An example input file is shown below:

```
Smith       Joe         1274        4.50
Thomas      Fred        4578        123.67
Gates       Bill        9782        0.43
Hilton      Paris       2934        1432345.90
```

The program should store the list of accounts in a ***linked list of structures***. The base structure should have two string members to hold the last name and first name of the account holder, an integer to hold the account number, and a double member to hold the account balance.

Once the program reads in the list from the input file, it should give the user the option to perform 4 different options; print the list on the screen, add a new account to the head of the list, calculate the average balance of all of the accounts, or exit the program and write the list to an output file. The program should use a "menu" format to allow the user to select the operation and should continue to ask the user for an operation, perform it, and ask again, until the user chooses to write the output file and exit the program. An example of the "menu" is shown below:

```
C:\Temp\egre246\code\hw_solutions>hw2
Program to read and sort a list of numbers using dynamic memory
allocation
Input filename:hw2_test1.txt

Enter option:
        (0) Write output file and exit
        (1) Print the list
        (2) Add new account at head of the list
        (3) Calculate and display average balance
                Choice ?
```

For the input file shown above, selecting option 1 would result in the following display:

```
Member List
Last name  first name   account no.       balance
-----------------------------------------------------
Hilton      Paris       2934              1432345.90
Gates       Bill        9782              0.43
Thomas      Fred        4578              123.67
Smith       Joe         1274              4.50
```

Selecting option 2 to add a new account and then selecting option 1 to print the list would result in the following display:

```
Enter new member's last name ?Enstein
Enter new member's first name ?Frank
Enter new member’s account number ?5801
Enter new member's balance ?0.05

Enter option:
        (0) Write output file and exit
        (1) Print the list
        (2) Add new account at head of the list
        (3) Calculate and display average balance
                Choice ?1

Member List
Last name  first name   account no.       balance
-----------------------------------------------------
Enstein     Frank       5801              0.05
Hilton      Paris       2934              1432345.90
Gates       Bill        9782              0.43
Thomas      Fred        4578              123.67
Smith       Joe         1274              4.50
```

Selecting option 3 with the above list would result in the following display:

```
There are 5 members with an average balance of 286494.91
```

Finally, selecting option 0 would result in the list being written to an output file in the same format as the input file. The name of the output file should be the name of the input file with the “.txt” extension replaced with “_out.txt”. For example, an input file with the name “hw2_test.txt” would result in an output file named “hw2_test_out.txt”.

For this assignment, you must turn in a zip file that contains a .h file that defines the structure and includes all global `#define` statements, a .c file that includes the implementation of any functions you construct for your solution, a .c file that includes the implementation of your `main()` program, a .txt file that includes a test input for your program, and a Makefile that compiles your complete solution.

Turn in your assignment by attaching the zip file to the assignment submission page.

Remember the class policy on late submissions – no late submissions are allowed unless prior arrangement is made with the instructor.