It is extremely important that you understand the programming assignments; please participate in the classroom discussion. You are expected to solve the programming assignments individually. You are encouraged to discuss with your friends. However, **do not copy**. Follow the submission and grading policies for full points.

Doubly Linked List **in C**

The primary focus is to implementation *doubly linked list* in C. Students are also challenged to implement/use *Stack ADT* to solve singly linked list related problem.

Important Notes: There are two (2) questions. The first one is the mandatory question and the second one is the optional bonus question. Please make your file names as Prog03q1.c (MAIN for mandatory), Prog03q2.c (MAIN for bonus), etc. as discussed in class.

Q1 (Mandatory): Please write code for *doubly linked list* in C for the following function prototypes using the given *typedef* and *struct* information.

```
typedef int ElementType;
struct Node;
typedef struct Node *PtrToNode;
typedef PtrToNode List;
typedef PtrToNode Position;
struct Node
{
  ElementType Element;
  Position    Prev;
  Position    Next;
};
```

Required functions:

```
char UserMenu( char *opt ); /* display menu and receive option */
void MakeEmptyList( List L ); /* create an empty list L */
void BuildList( List L, FILE *fpi ); /* build list using values from file "hw3in.dat" */
void DisplayList( List L ); /* display all items in list L */
void InsertItem( ElementType X, List L, Position P ); /* insert item X at position P   in list L */
void DeleteItem( ElementType X, List L ); /* delete item X from list L */
void SaveList( List L, FILE *fpo ); /* save the list L in a file "hw3out.dat" */
void DestroyList( List L ); /* delete all items in list L and free L */
void ExitProg( void ); /* normal exit */
```

Additional functions:

```
int IsEmpty( List L ); /* check if list L is empty */
int IsFirst( Position P, List L ); /* check if position P holds the first item in list L */
int IsLast( Position P, List L ); /* check if position P holds the list item in list L */
Position FindItemPos( ElementType X, List L ); /* find the position of item X in list L */
void DeleteList( List L ); /* delete all items in list L (don't free L) */
```

Sample (from input file):

10
25
15
30
20

NOTE: "prog04in.dat" is in /usr/users/User11/drzaman/CS460/Samplecodes/

Insert 5
Insert 35
Insert 45
Delete 35
Delete 15

Sample (from output file):

Item 1: 10
Item 2: 25
Item 3: 30
Item 4: 20
Item 5: 5
Item 6: 45

Function "main"
Driver program

Function "UserMenu"
Display user friendly menu

Sample (menu):

Welcome to CS460 (Fall 2013) Prog04 Solution!

Please see your options:
1) Exit
2) Make an empty list
3) Build list using file "prog04in.dat"
4) Display the list
5) Find an item
6) Insert an item
7) Delete an item
8) Save the list in file "proh04out.dat"
9) Destroy the list

Please enter your option >

Sample Codes:

```
drzaman@kirk:~/CS460/Samplecodes$ pwd
/usr/users/User11/drzaman/CS460/Samplecodes
drzaman@kirk:~/CS460/Samplecodes$ ls -ltr
-rw-r--r-- 1 drzaman drzaman   16 Sep 11 01:25 prog04in.dat
```

Q2 (Optional 25% Bonus Question): Given a singly linked list of characters, write a program in C using Stack that returns true if the given list is palindrome, else false. For example,

SLL "L->b->a->c->a->b->a->NULL" is Not Palindrome; but
SLL "L->a->b->a->c->a->b->a->NULL" is Palindrome.

METHOD 1 (using a Stack) involves three steps:

(i) Traverse the given list from head to tail and push every visited node to stack.
(ii) Traverse the list again. For every visited node, pop a node from stack and compare data of popped node with currently visited node.
(iii) If all nodes matched, then return true, else false.