Sheet # 2 (Linked Lists)

Important notes:

- This sheet is a group work (two students in each group)
- You have to submit *only a hard copy* of exercises **2 and 4**
- Due date: Saturday (12 March 2011) before 8:00 AM, in box#16
- You should bring a soft-copy of your code (ex#4) in the lab time, and you will be given an additional function to solve it in lab time.
- Linked list ADT is attached with this file, you will need it to solve Ex#4

> ***For exercises 1 to 3:*** *Make sure that your algorithms works for empty lists, lists containing only one node, and lists containing many nodes.*

Exercise #1:

Write the algorithm of a *SortedInsert()* function which given a list that is sorted in increasing order, and a single node, inserts the node into the correct sorted position in the list.

```
void SortedInsert(struct node** headRef, struct node* newNode) {
// Your code...
}
```

Exercise #2:

Write an algorithm of a function to print out the information stored in a circular list.

Exercise #3:

Given two circular lists A and B, write the algorithm of a function to join them into one circular list A, leaving the other list B empty.

Exercise #4 (Coding):

Write a program to manage an address book using a linked list. Each entry in the linked list contains: *name, phone, and city*; where the name is unique. The program allows the user to do one of the following:

a. Insert new entry: by choosing this option, the program will take the data (name, phone, city) from the user.

b. Delete an entry: by choosing this option, user will enter person's name, and the program will delete his data. You should print a message to tell the user either if you delete the data, or its not found.

c. Print the address book: this will print all of the addresses in the list, in the form:

```
Mohammed, 1234567, Riyadh
Sara, 9876543, Makkah
.....
```

If the list is empty, you must print an appropriate message.