Tutorial – Linked Lists

Q1 You are given the following structure definitions and variable declarations,

```
struct person{
  char firstName[15];
  char lastName[15];
  struct{
     int age;
     float height;
     float weight;
     char firstName[15];
     }Info,*InfoPtr;
  struct person personP;
  }student1;
typedef struct person person_t;
person_t* studentPtr = &student1;
person_t** studentPtrPtr = &studentPtr;
```

a Is there any syntax error?

b Write an expression that can be used to access *age* from *studentPtr*.

c Write an expression that can be used to access *age* from *studentPtrPtr*.

Q2. (moveEvenItemsToBackLL) Write a C function `moveEvenItemsToBackLL()` that moves all the even integers to the back of the linked list.

The function prototype is given as follows:

```
void moveEvenItemsToBackLL(LinkedList *ll);
```

Some sample inputs and outputs sessions are given below:

If the linked list is **2, 3, 4, 7, 15, 18**:

```
The resulting Linked List after moving even integers to the
back of the Linked List is: 3 7 15 2 4 18
```

If the linked list is **2, 7, 18, 3, 4, 15**:

```
The resulting Linked List after moving even integers to the
back of the Linked List is: 7 3 15 2 18 4
```

If the current linked list is **1, 3, 5**:

```
The resulting Linked List after moving even integers to the
back of the Linked List is: 1 3 5
```

```
If the current linked list is 2 4 6:
The resulting Linked List after moving even integers to the
back of the Linked List is: 2 4 6
```

Q3. (moveMaxToFront) Write a C function `moveMaxToFront()` that traverses a linked list of integers at most once, then moves the node with the largest stored value to the front of the list.

The function prototype is given as follows:
```
int moveMaxToFront(ListNode **ptrHead);
```

For example, if the linked list is (**30, 20, 40, 70, 50**), the resulting linked list will be (**70, 30, 20, 40, 50**).

```
1: Insert an integer to the linked list:
2: Move the node with the largest stored value to the front of the list:
0: Quit:

Please input your choice(1/2/0): 1
Input an integer that you want to add to the linked list: 30
The Linked List is: 30
Please input your choice(1/2/0): 1
Input an integer that you want to add to the linked list: 20
The Linked List is: 30 20
Please input your choice(1/2/0): 1
Input an integer that you want to add to the linked list: 40
The Linked List is: 30 20 40
Please input your choice(1/2/0): 1
Input an integer that you want to add to the linked list: 70
The Linked List is: 30 20 40 70
Please input your choice(1/2/0): 1
Input an integer that you want to add to the linked list: 50
The Linked List is: 30 20 40 70 50

Please input your choice(1/2/0): 2
The resulting Linked List is: 70 30 20 40 50

Please input your choice(1/2/0): 0
```

Q4. (removeDuplicatesSortedLL) Write a C function `removeDuplicatesSortedLL()` that removes all duplicate values from a sorted linked list. ***You may assume that the list is already in ascending sorted order.***

The function prototype is given below:
```
void removeDuplicatesSortedLL(LinkedList *ll);
```

For example:
If the linked list is (**1, 2, 2, 4, 4, 5, 5**), the resulting linked list will be (**1, 2, 4, 5**).
If the linked list is (**1, 2, 3, 4, 5**), the resulting linked list will be (**1, 2, 3, 4, 5**)

Sample test cases are given below:
```
1: Insert an integer to the linked list:
2: Remove duplicates from a sorted linked list:
0: Quit:

Please input your choice(1/2/0): 1
Input an integer that you want to add to the linked list: 1
```

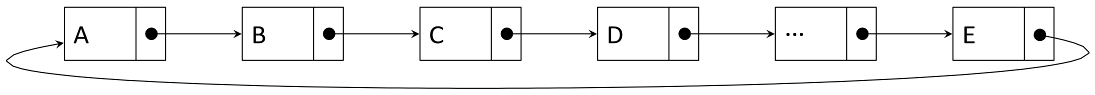
```
The resulting linked list is: 1
Please input your choice(1/2/0): 1
Input an integer that you want to add to the linked list: 2
The resulting linked list is: 1 2
Please input your choice(1/2/0): 1
Input an integer that you want to add to the linked list: 2
The resulting linked list is: 1 2 2
Please input your choice(1/2/0): 1
Input an integer that you want to add to the linked list: 4
The resulting linked list is: 1 2 2 4
Please input your choice(1/2/0): 1
Input an integer that you want to add to the linked list: 4
The resulting linked list is: 1 2 2 4 4
Please input your choice(1/2/0): 1
Input an integer that you want to add to the linked list: 5
The resulting linked list is: 1 2 2 4 4 5
Please input your choice(1/2/0): 1
Input an integer that you want to add to the linked list: 5
The resulting linked list is: 1 2 2 4 4 5 5

Please input your choice(1/2/0): 2
The resulting linked list after removing duplicate values from the sorted
linked list is: 1 2 4 5

Please input your choice(1/2/0): 0
```

Information: Program templates for questions (**Q2**, **Q3**, and **Q4**) are available in NTULearn. You must use them to implement your functions.

Additional Question

Q1. We assign the link of the last node to the first node instead of assigning it to a null value. This turns the linked list into a circular linked list. Let Aptr and Bptr point to any two nodes in the linked list. What is the outcome of the following functions?

Figure 1.1: A Circular Linked List



```
typedef struct node{
      int item;
      struct node next;
      }ListNode;

void Q3F1(ListNode *Aptr, ListNode *Bptr)
{
   Q3F2(Aptr, Bptr);
   Q3F2(Bptr, Aptr);
}
void  Q3F2(ListNode *s, ListNode *q)
{
    ListNode *temp = s;

    while(temp->next != q) temp = temp->next;
    temp->next = s;
}
```