NANYANG TECHNOLOGICAL UNIVERSITY
SCHOOL OF COMPUTER ENGINEERING

CZ1007 DATA STRUCTURES

Title:	Report for Assignment 1
Student's (official) name:	Huang Jian Wei
Matric No:	U1521567A
Tutorial group:	FSP5

Question 1 InsertSortedLinkedList()

```
int insertSortedLinkedList(LinkedList *ll, int item)
{

	ListNode *temp, *prev, *next; //stores the nodes "temp","previous" and
	"next"
	temp = (ListNode*)malloc(sizeof(ListNode)); //allocate a piece of memory
	for temp
	temp->item = item;
	temp->next = NULL;//initialize temp
	int index = 0;//index of new position of item to be returned

	if (!ll->head) { //if linkedlist is empty, put first user input into head
	node
		ll->head = temp;
	}

	else {	//if not empty, transverse through linkedlist and find the
		right position to slot it into the linked list

		prev = NULL;
		next = ll->head;//contains a value of the current node

		while (next && next->item <= item) { //loop through linkedlist; if
		item in linkedlist is smaller than the user input, jump to the next
		node
			prev = next;
			next = next->next;
			index++; //increase index count for each jump
		}
		//break out of loop if item in linkedlist bigger than user input
		if (prev) {//if previous is not NULL(meaning item in list smaller
		than user input at least once AND prev is initialized to "next" node

			temp->next = prev->next;
			prev->next = temp; //prev->next pointing to new value
		}
		//continue on next page
```

```
            else { //else if the user input does not go thru the loop e.g
            smallest case,   head node will be user input

                  temp->next = ll->head;
                  ll->head = temp;
            }

      }
      return index; //returns index of position inserted

}
```

Question 1 Test Case 1

1.) Inserting value ‘6’ into LinkedList of 2,3,5,7,9

```
C:\WINDOWS\system32\cmd.exe
Please input your choice(1/2/3/0):1
Input an integer that you want to add to the linked list:
5
The resulting linked list is:
2 3 5

Please input your choice(1/2/3/0):1
Input an integer that you want to add to the linked list:
7
The resulting linked list is:
2 3 5 7

Please input your choice(1/2/3/0):1
Input an integer that you want to add to the linked list:
9
The resulting linked list is:
2 3 5 7 9

Please input your choice(1/2/3/0):1
Input an integer that you want to add to the linked list:
6
The resulting linked list is:
2 3 5 6 7 9

Please input your choice(1/2/3/0):
```

2. Inserting a negative value and ‘03’

```
C:\WINDOWS\system32\cmd.exe
Please input your choice(1/2/3/0):1
Input an integer that you want to add to the linked list:
9
The resulting linked list is:
2 3 5 7 9

Please input your choice(1/2/3/0):1
Input an integer that you want to add to the linked list:
6
The resulting linked list is:
2 3 5 6 7 9

Please input your choice(1/2/3/0):1
Input an integer that you want to add to the linked list:
-5
The resulting linked list is:
-5 2 3 5 6 7 9

Please input your choice(1/2/3/0):1
Input an integer that you want to add to the linked list:
03
The resulting linked list is:
-5 2 3 3 5 6 7 9

Please input your choice(1/2/3/0):
```

Question 2 Test Case 2

To get index of new nodes inserted

Test 1

```
C:\WINDOWS\system32\cmd.exe
9
The resulting linked list is:
2 3 5 7 9

Please input your choice(1/2/3/0):1
Input an integer that you want to add to the linked list:
6
The resulting linked list is:
2 3 5 6 7 9

Please input your choice(1/2/3/0):2

The value 6 was added at index 3

Please input your choice(1/2/3/0):1
Input an integer that you want to add to the linked list:
10
The resulting linked list is:
2 3 5 6 7 9 10

Please input your choice(1/2/3/0):2

The value 10 was added at index 6

Please input your choice(1/2/3/0):
```

Test 2

```
C:\WINDOWS\system32\cmd.exe
2 3 5 7 9

Please input your choice(1/2/3/0):1
Input an integer that you want to add to the linked list:
6
The resulting linked list is:
2 3 5 6 7 9

Please input your choice(1/2/3/0):1
Input an integer that you want to add to the linked list:
-5
The resulting linked list is:
-5 2 3 5 6 7 9

Please input your choice(1/2/3/0):1
Input an integer that you want to add to the linked list:
03
The resulting linked list is:
-5 2 3 3 5 6 7 9

Please input your choice(1/2/3/0):2

The value 3 was added at index 3

Please input your choice(1/2/3/0):
```

Conclusion: Program running as intended. insertSortedLinkedList() simply returns an index which is incremented every time there is a jump to the next node.

Discussion & Analysis

Write a C function insertSortedLinkedList() that adds a new node to an existing linked list, maintains the list values in ascending sorted order. Note that the existing list is already in ascending sorted order.

In order to implement such a function as mentioned above, the first thing that comes to mind is that we definitely need a condition that checks user against the elements in the list; if user input is bigger than element in the list, we transverse until it reaches its correct position and we add the new user input into the new position.

The function can work independently and does not require given functions like insertNode() or findNode(). A counter declared as “Index” is used to count the new position of the new user input in the linked list.

We also must take note of special case where the linked list is empty, where the user Input will become the head of the linkedlist.

Question 2 sortMergedLinkedList()

```
void sortedMergeLinkedList(LinkedList *ll1, LinkedList *ll2, LinkedList
*resultMergedList)
{
	ListNode *temp1, *temp2; //Declare two temp Nodes
	temp1 = (ListNode*)malloc(sizeof(ListNode)); //Allocating memory to
temp 1
	temp2 = (ListNode*)malloc(sizeof(ListNode)); //Allocating memory to
temp 2

	//initializing temp1 and temp2
	temp1 = ll1->head;
	temp2 = ll2->head;

	//initialize result,list 3 or mergedList
	ListNode* mergedList;
	ListNode** tail = &mergedList; //address of pointer variable "tail"
	link to address of mergedList
	mergedList = (ListNode*)malloc(sizeof(ListNode)); //allocate memory
	for mergedList Node
	if (resultMergedList->head) { //empty list if it is not empty
		resultMergedList->head = NULL;
	}
	if (ll2->head == NULL) { //if ll2 does not exist, mergedlist will be
	linkedlist1 and prints list 1 (Catch statement)
		resultMergedList->head = ll1->head;
		return;
	}
	if (ll1->head == NULL) { //if ll1 does not exist, mergedlist will be
	linkedlist1 and prints list 2( Catch statement)
		resultMergedList->head = ll2->head;
		return;
	}


//continue on next page
```

```
            while (temp1 != NULL && temp2 != NULL) { //if there exist
            element to transverse
            if (temp1->item < temp2->item) { //temp1's integer less than
            temp2's Integer

                  *tail = temp1; //tail node will store node of temp1
                  temp1 = temp1->next; //pointer will point to next node of
                  temp1/linklist1
            }
            else {
                  *tail = temp2;//if temp1's integer more than temp2, will
                  insert temp2 integer first
                  temp2 = temp2->next;// pointer will point to next node of
                  temp2/linklist2
            }
            tail = &((*tail)->next);
      }
            *tail = temp1 ? temp1: temp2; //if temp1 is true, value of tail
            = address of temp1 else, value of temp 2
      resultMergedList->head = mergedList; //value of merged list

      ll1 ->head = NULL; //reset linklist1
      ll2 ->head = NULL; //reset linklist2

}

int insertSortedLinkedList(LinkedList *ll, int item) //code as per p1.c
solution
{
      /* add your code here */
      ListNode *temp, *prev, *next;
      temp = (ListNode*)malloc(sizeof(ListNode));
      temp->item = item;
      temp->next = NULL;
      int index = 0;

      if (!ll->head) {
            ll->head = temp;
      }
```

```
else {
            prev = NULL;
            next = ll->head;
            while (next && next->item <= item) {
                  prev = next;
                  next = next->next;
                  index++;

            }


            if (prev) {
                  temp->next = prev->next;
                  prev->next = temp;
            }

            else {
                  temp->next = ll->head;
                  ll->head = temp;
            }

      }
      return index;
}
```

Question 2 Test Case 1

Input as per assignment requirement

```
C:\WINDOWS\system32\cmd.exe

Please input your choice(1/2/3/0): 1
Input an integer that you want to add to the linked list 1: 4
Linked list 1: 2 4

Please input your choice(1/2/3/0): 1
Input an integer that you want to add to the linked list 1: 6
Linked list 1: 2 4 6

Please input your choice(1/2/3/0): 2
Input an integer that you want to add to the linked list 2: 1
Linked list 2: 1

Please input your choice(1/2/3/0): 2
Input an integer that you want to add to the linked list 2: 2
Linked list 2: 1 2

Please input your choice(1/2/3/0): 2
Input an integer that you want to add to the linked list 2: 3
Linked list 2: 1 2 3

Please input your choice(1/2/3/0): 3
The resulting linked list is: 1 2 2 3 4 6

Please input your choice(1/2/3/0):
```

Merge successful.

Test Case 1

Linklist 1 have data and LinkedList 2 is empty

```
C:\WINDOWS\system32\cmd.exe

1: Insert an integer to the sorted linked list 1:
2: Insert an integer to the sorted linked list 2:
3: Print sorted merged linked list:
0: Quit:
Please input your choice(1/2/3/0): 1
Input an integer that you want to add to the linked list 1: 2
Linked list 1: 2

Please input your choice(1/2/3/0): 3
The resulting linked list is: 2

Please input your choice(1/2/3/0): _
```

Result: Merge successful

LinkedList 2 have data and LinkedList 1 is empty

```
C:\WINDOWS\system32\cmd.exe
1: Insert an integer to the sorted linked list 1:
2: Insert an integer to the sorted linked list 2:
3: Print sorted merged linked list:
0: Quit:
Please input your choice(1/2/3/0): 2
Input an integer that you want to add to the linked list 2: 5
Linked list 2: 5

Please input your choice(1/2/3/0): 3
The resulting linked list is: 5

Please input your choice(1/2/3/0):
```

Result: Merge successful

Test Case 3

User is able to input linkedlist1 and linkedlist2 and merge again repeatedly. LinkedList will be reset every time it merge

```
C:\WINDOWS\system32\cmd.exe
3: Print sorted merged linked list:
0: Quit:
Please input your choice(1/2/3/0): 1
Input an integer that you want to add to the linked list 1: 2
Linked list 1: 2

Please input your choice(1/2/3/0): 2
Input an integer that you want to add to the linked list 2: 3
Linked list 2: 3

Please input your choice(1/2/3/0): 3
The resulting linked list is: 2 3

Please input your choice(1/2/3/0): 2
Input an integer that you want to add to the linked list 2: 3
Linked list 2: 3

Please input your choice(1/2/3/0): 1
Input an integer that you want to add to the linked list 1: 5
Linked list 1: 5

Please input your choice(1/2/3/0): 3
The resulting linked list is: 3 5

Please input your choice(1/2/3/0):
```

Conclusion : Program able to run successfully in every cases.

Discussion & Analysis

Write a function sortedMergeLinkedList() that takes two lists, each of which is sorted in ascending order, and merges two lists together into one list that is also in ascending order. sortedMergeLinkedList() should return the new list.

Write an additional C function insertSortedLinkedList() in order to create two sorted linked lists in ascending order which are used for merging

To begin working on this problem, we must understand that the first step has to be creating two linked list using insertSortedLinkedList(), which we implemented in Q1. After having 2 linkedlist, the next step has to be merging the list together.

We start with special cases where List 1 is NULL. The merge result(NULL, LL2) will be LL2, hence we assign merged result to LL2 and return then handler to the main function. This also apply to the case where List 2 is NULL.

We can then began the merge by going through the both list to and compare the elements in the list. E.g head of linkedlist 1 > head of linkedlist 2, head of linkedlist 1 will then be the head of mergedlist. The program will then compare next element of linkedlist 1 against head of linkedlist 2. If it is smaller, the element in linkedlist 2 will be place inside the mergedlist. The traversing of the linkedlist will be carried out using a while loop where condition is that the next element of Linkedlist 1 and Linkedlist 2 is not equal to NULL.

Question 3 sortMergedLinkedList()

```
void createQueueFromLinkedList(LinkedList *ll, Queue * q) //creating a queue
from existing linkedlist
{
	/* add your code here */
	if (isEmptyQueue) {
		removeAllItemsFromQueue(q);
	}// empty the queue if it is not empty
	while (ll->size)
	{
		enqueue(q, ll->head->item); //queue item from Linkedlist
		removeNode(ll, 0); //remove node from Linkedlist once it has been
		placed into queue

	}

}

void removeEvenValues(Queue *q) //function to remove even values from queue
{
	//allocate memory for queue which contains all the odd element
	Queue *odd = (Queue*)malloc(sizeof(Queue));
	odd->ll.size = 0; //initialize odd queue
	odd->ll.head = NULL;

	while (isEmptyQueue(q)) { //while there exist an element in q
	//check whether if the integer is odd, if true, enqueue integer  into
	odd queue
		if ((q->ll.head->item) % 2 != 0) {
			enqueue(odd, q->ll.head->item);
		}
		//else even will remove integer from queue
		dequeue(q);
	}

	//while element exist in odd, adds integer resides in odd into q
	while (isEmptyQueue(odd)) {
		enqueue(q, odd->ll.head->item);
		dequeue(odd); //remove odd every time it is place into queue q
	}
	free(odd);

}
```

Question 3 Test Case 1

```
C:\WINDOWS\system32\cmd.exe
Input an integer that you want to add to the linked list: 2
The resulting linked list is: 1 2

Please input your choice(1/2/3/0): 1
Input an integer that you want to add to the linked list: 3
The resulting linked list is: 1 2 3

Please input your choice(1/2/3/0): 1
Input an integer that you want to add to the linked list: 4
The resulting linked list is: 1 2 3 4

Please input your choice(1/2/3/0): 1
Input an integer that you want to add to the linked list: 5
The resulting linked list is: 1 2 3 4 5

Please input your choice(1/2/3/0): 2
The resulting queue is: 1 2 3 4 5

Please input your choice(1/2/3/0): 3
The resulting queue after removing even integers is: 1 3 5

Please input your choice(1/2/3/0): 2
The resulting queue is: Empty

Please input your choice(1/2/3/0):
```

Result as per sample output, Linkedlist will be empty as it will be transferred into queue.

Question 3Test Case 2

```
C:\WINDOWS\system32\cmd.exe
The resulting linked list is: 5 7

Please input your choice(1/2/3/0): 1
Input an integer that you want to add to the linked list: 2
The resulting linked list is: 5 7 2

Please input your choice(1/2/3/0): 1
Input an integer that you want to add to the linked list: 8
The resulting linked list is: 5 7 2 8

Please input your choice(1/2/3/0): 1
Input an integer that you want to add to the linked list: 12
The resulting linked list is: 5 7 2 8 12

Please input your choice(1/2/3/0): 1
Input an integer that you want to add to the linked list: 19
The resulting linked list is: 5 7 2 8 12 19

Please input your choice(1/2/3/0): 2
The resulting queue is: 5 7 2 8 12 19

Please input your choice(1/2/3/0): 3
The resulting queue after removing even integers is: 5 7 19

Please input your choice(1/2/3/0):
```

Test case with different input
Conclusion: Program working as intended without any error

Discussion & Analysis

Write a C function createQueueFromLinkedList() to create a queue (linked-list-based) by enqueuing all integers storing in the linked list. The first node of the linked list is enqueued first, and then the 2nd node, and so on. Remember to empty the queue in the beginning if the queue is not empty. After the queue is built, write another C function removeEvenValues() to remove all even integers in the queue. Note that you should only use enqueue() or dequeue() when you add or remove integers from queues. You are not allowed to directly access the underlined linked list of the queue. (Hint: you may define an auxiliary queue inside the function to help for the task.)

First, we check if the queue if empty in the beginning using the given function isEmptyQueue(). If it returns a true, we clear everything in the queue using removeAllItemsFromQueue(). We can then begin using the given function, enqueue() to queue all our items in the exisiting linkedlist. The next step will be to remove all even values from the queue. We create an temporary queue where only odd elements exist. We then transverse the current queue to and check whether the current item is odd or even using mod2. If it is odd, we place it in the new queue, else we will remove the even element using dequeue().

We then loop again to place all the odd elements back to the original queue.

Another possible way to do this is to simply dequeuer even element from the origina queue.

Question 4 frontBackSplitLinkedList()

```
void frontBackSplitLinkedList(LinkedList* ll, LinkedList * resultFrontList,
LinkedList *resultBackList)
{

      if (!ll->head)
      return; //handle the case where list is empty
      ListNode *endOfFrontList = NULL;
      ListNode *middle = ll->head;
      int i = 0;//initialize i

      if (ll->size % 2 ==0)//make sure that it will loop the correct number
      of times
            i = 1;
      else i = 0;

      for (i; i <= ll->size/2; i++) {

      endOfFrontList = middle;//tranverse until the split of the linkedlist,
      pointing to the middle element
      middle = middle->next;

            }
      endOfFrontList->next = NULL;  // ends the 2nd half of list

      resultFrontList->head = ll->head;//initialize front list, starting from
      the start of the linked list

      resultBackList->head = middle;//initialize backlist, starting from
      middle of the list
}
```

Test Case 1

Test: Scenario as per pdf
Linked List: 2 3 5 6 7 (odd)

```
C:\WINDOWS\system32\cmd.exe
The resulting linked list is: 2

Please input your choice(1/2/3/0): 1
Input an integer that you want to add to the linked list: 3
The resulting linked list is: 2 3

Please input your choice(1/2/3/0): 1
Input an integer that you want to add to the linked list: 5
The resulting linked list is: 2 3 5

Please input your choice(1/2/3/0): 1
Input an integer that you want to add to the linked list: 6
The resulting linked list is: 2 3 5 6

Please input your choice(1/2/3/0): 1
Input an integer that you want to add to the linked list: 7
The resulting linked list is: 2 3 5 6 7

Please input your choice(1/2/3/0): 3
The resulting linked lists after splitting the given linked list are:
Front linked list: 2 3 5
Back linked list: 6 7


Please input your choice(1/2/3/0):
```

Result: Able to split Linkedlist successfully. 2 3 5 front and 6 7 back.

Test Case 2
Now, we test if the program work on even number

```
C:\WINDOWS\system32\cmd.exe
Input an integer that you want to add to the linked list: 2
The resulting linked list is: 2

Please input your choice(1/2/3/0): 1
Input an integer that you want to add to the linked list: 4
The resulting linked list is: 2 4

Please input your choice(1/2/3/0): 1
Input an integer that you want to add to the linked list: 4
The resulting linked list is: 2 4 4

Please input your choice(1/2/3/0): 1
Input an integer that you want to add to the linked list: 7
The resulting linked list is: 2 4 4 7

Please input your choice(1/2/3/0): 2
The resulting linked list is: 2 4 4 7

Please input your choice(1/2/3/0): 3
The resulting linked lists after splitting the given linked list are:
Front linked list: 2 4
Back linked list: 4 7


Please input your choice(1/2/3/0):
```

Result: Program running as intended.

Discussion & Analysis

Write a C function frontBackSplitLinkedList() that splits singly linked list into two sublists – one for the front half, and one for the back half. If the number of elements is odd, the extra element should go in to the front list. The frontBackSplitLinkedList() prints two lists which contains frontList and backList.

To get the expected output, there are many ways to do this. But the simplest and most straightforward way has to be declaring a pointer that will always point to the middle of the list ; e.g 1[0],2[1],3[2],4[3] where [x] is the index. The start of the 2nd half of the list will start from that pointer. The 1st half will start from the head of the linkedList.

To get this pointer, we check whether the number of element in the linkedlist is even or odd. Then by using the (size of the linkedlist/2) we can immediately find split up the list into two.

.

----------End of Report----------