**Worksheet 3(Linked List)**

1. We are interested to maintain a list of floating numbers in **ascending** order in an array implementation of a linked list. Assuming that the linked list is initialized as shown in the table below, what will the table look like after the operations given below are performed?

Table at Initialization

|  |  |  |
| --- | --- | --- |
| 0 |  | 1 |
| 1 |  | 2 |
| 2 |  | 3 |
| 3 |  | 4 |
| 4 |  | 5 |
| 5 |  | 6 |
| 6 |  | 7 |
| 7 |  | 8 |
| 8 |  | 9 |
| 9 |  | -1 |

Insert and Delete Operations

Insert 8

Insert 21

Insert 4

Insert 15

Insert 2

Delete 8

Insert 6

Insert 33

Delete 4

Delete 2

Insert 13

Insert 7

1. Discuss the requirements and results of the following Operations, and provide their implementation with c++. Assume that the linked list contains floating number.
2. isEmpty( )
3. isFull( )
4. sizeOf( )
5. empty( )
6. removeRange( )
7. average( )
8. copy( )
9. removeFirstAndAddAtLast( )
10. replaceItem( )
11. What do the following functions do for a given Linked List?

|  |  |
| --- | --- |
| 1. void fun1(struct node\* head)   {     if(head == NULL)       return;     fun1(head->next);     cout<<(head->data);  } | 1. void fun2(struct node\* head)   {     if(head== NULL)      return;     cout<<(head->data);     if(head->next != NULL )       fun2(head->next->next);    cout<<(head->data);  } |

1. Implement a function named **reverseList()** as a new function for the linked list toolkit. The function should reverse the list.
2. With the definition of a Doubly Linked List provided, try to give the implementation code for some of the operations you can perform with a Doubly Linked List

struct node  
{  
 int value; //value stored in the node   
 node \*next; //pointer to next node   
 node \*prev; //pointer to previous node   
};

struct dlist  
{  
 node \*front; //pointer to front of list   
 node \*back; //pointer to back of list   
}

|  |  |
| --- | --- |
| 1. create(dlist \*listA) 2. void insertFront(int value); 3. void insertBack(int value); 4. void removeFront(dlist \*listA) 5. void removeBack(dlist \*listA); 6. void insertBefore(dlist \*listA ,int value, node \*nodeB); 7. void insertAfter(dlist \*listA , int value, node \*nodeA); | 1. void removeBefore(dlist \*listA, node \*nodeB); 2. void removeAfter(dlist \*listA node \*nodeA); 3. void removeNode(dlist \*listA, node \*newNode); 4. void printDListFront(dlist \*listA); 5. void printDListBack(dlist \*listA); 6. void deleteDList(dlist \*listA); |

1. Implement a function which accepts digits from the keyboard and uses singly linked lists to implement integers of unlimited size. Each node of the list should store one digit of the integer.
2. For problem 6 implement the function add(), which adds two integers implemented with linked list.
3. Give the implementation of the following functions for a circular linked list. A circular linked list is one in which the next field for the last link node of the list points to the first link node of the list.
4. insertNode( )
5. deleteNode( )
6. Search( )
7. Give the implementation of the following functions for a linked list whose data values are float.
8. Merge two linked list assuming the lists are sorted. The resulting list should be also sorted.
9. Split a sorted linked list into two sorted linked list in which one of the list contains all the elements less than or equal to a certain key value.
10. Repeat problem(c) if the original list and the spitted list are unsorted.