**LAB REPORT ON**

**Data Structures**



**Lab No. 5**

**Date 2020/10/09**

**Topic: Doubly Linked List**

**Submitted By**  **Submitted To**

Tapendra Pandey

BCT-D Department of Electronics

075BCT093 and Computer Engineering

**INTRODUCTION**

**Doubly Linked List**

Doubly linked list is a complex type of linked list in which a node contains a pointer to the previous as well as the next node in the sequence. Every nodes in the doubly linked list has three fields: left pointer, right pointer and Data.

Left pointer will point to the node in the left side (or previous node) that is left pointer will hold the address of the previous node. Right pointer will point to the node in the right side (or next node) that is right pointer will hold the address of the next node. Data will store the information of the node.

A stack, hash table, and binary tree can be implemented using a doubly linked list. Also undo functionality in Word or Photoshop is its application.

**ALGORITHMS**

Algorithm for following cases in **doubly linked list**:

1. **INSERTION OPERATIONS**
   1. INSERT NODE AT BEGINNING OF THE LIST
   2. INSERT NODE AT END OF THE LIST
   3. INSERT NODE AFTER SPECIFIC NODE
   4. INSERT NODE BEFORE SPECIFIC NODE
2. **DELETION OPERATIONS**
   1. DELETE NODE FROM BEGINNING OF THE LIST
   2. DELETE NODE FROM END OF THE LIST
   3. DELETE NODE AFTER SPECIFIC NODE
   4. DELETE NODE BEFORE SPECIFIC NODE

**a) Insert node at the beginning of the linked list**

Step 1: SET NEW\_NODE=TEMP

Step 2: SET START=NULL

Step 3: SET TEMP→ DATA = NUM

Step 4: SET TEMP→LEFT= NULL

Step 5: SET TEMP→RIGHT= START

IF START == NULL

START=TEMP

END OF IF

Step 6: SET START→LEFT= TEMP

Step 7: START=TEMP

Step 8: EXIT

**b) Insert node at end of the linked list**

Step 1: SET NEW\_NODE=TEMP, P

Step 2: SET P=START

Step 4: SET TEMP →DATA = NUM

Step 5: SET TEMP→RIGHT = NULL

Step 6: Repeat Step 7 while P→ RIGHT! = NULL

Step 7: SET P= P→RIGHT

[END OF LOOP]

Step 8: SET P→RIGHT =TEMP

Step 9: SET TEMP→LEFT= P

Step 10: EXIT

**c) Insert node after specific node**

Step 1: SET NEW\_NODE=TEMP, P

Step 2: SET P=START

Step 3: IF START = NULL

Write UNDERFLOW

Go to Step 11

Step 3: SET TEMP →DATA = NUM

Step 4: TEMP->RIGHT=p->RIGHT;

Step 5: TEMP->LEFT=p;

Step 6: P->RIGHT->LEFT=TEMP;

Step 8: P->RIGHT=TEMP;

Step 9: Repeat Step 10 while P→DATA! = NUM

Step 10: SET P= P→ RIGHT

[END OF LOOP]

Step 11: EXIT

**d) Insert node before specific node**

Step 1: SET NEW\_NODE=TEMP

Step 2: SET P= START

Step 3: IF START = NULL

Write UNDERFLOW

Go to Step 6

[END OF IF]

Step 4: SET TEMP →DATA = NUM

Step 5: IF POS=1

SET START=NULL

SET TEMP→ DATA = NUM

SET TEMP→LEFT= NULL

SET TEMP→RIGHT= START

IF START == NULL

START=TEMP

END OF IF

SET START→LEFT= TEMP

START=TEMP

END OF IF

Step 6: ELSE

SET P=START

IF START = NULL

Write UNDERFLOW

Go to Step 11

SET TEMP →DATA = NUM

TEMP->RIGHT=P->RIGHT;

TEMP->LEFT=P;

P->RIGHT->LEFT=TEMP;

P->RIGHT=TEMP;

Repeat Step 10 while P→DATA! = NUM

Step 10: SET P= P→ RIGHT

[END OF LOOP]

Step 11: EXIT

2.

**a) Delete node from beginning of the list**

Step 1: IF START = NULL

Write UNDERFLOW

Go to Step 7

[END OF IF]

Step 2: SET NEW\_NODE=TEMP

Step 3: SET TEMP= START

Step 4: START = START->RIGHT

Step 5: START->LEFT = NULL

Step 6: DELETE TEMP

Step 7: EXIT

**b) Delete at the end**

Step 1: IF START = NULL

Write UNDERFLOW

Go to Step 7

[END OF IF]

Step 2: SET P= START

Step 3: Repeat Step 4 while P→RIGHT! = NULL

Step 4: SET P= P→RIGHT

[END OF LOOP]

Step 5: SET P→ LEFT→ RIGHT = NULL

Step 6: DELETE P

Step 7: EXIT

**c) Delete after specific node**

Step 1: IF START= NULL

Write UNDERFLOW

Go to Step 9

[END OF IF]

Step 2: SET P=START

Step 3: Repeat Step 4 while P→DATA! = NUM

Step 4: SET P= P→RIGHT

[END OF LOOP]

Step 5: SET TEMP = P→ RIGHT

Step 6: SET P→ RIGHT = TEMP →RIGHT

Step 7: SET TEMP →RIGHT→ LEFT= P

Step 8: DELETE TEMP

Step 9: EXIT

**d) Delete before specific node**

Step 1: IF POS=1

Write UNDERFLOW

Step 2: IF POS=2

DO ALL STEPS OF DELETE FROM BEGINNING

END OF IF

Step 3: ELSE

POS=POS-2

DO ALL STEPS OF DELETE AFTER SPECIFIC NODE

Step 4: EXIT

**DISCUSSION AND CONCLUSION**

In this lab, we learnt about the doubly linked list, its operations and implementation. We did insertion and deletion operations of linked list using doubly linked list. These implementations helped us to realize advantage of doubly linked list over singly linked list. The traversing process was made easier with the help of extra previous pointer. Also deletion process was more efficient in the doubly linked list implementation

Hence, in the lab, we understood and implemented algorithms of doubly linked list.