Assignment No. 5

Linked Lists

Aim
1. Write a C++ program to maintain employee's information using singly linked list. Store Employee ID and Employee Name. Write functions for a) Computing total no. of employees in the industry b) Displaying all employee's details (Employee ID and Employee Name) c) Inserting new employee d) Deleting existing employee e) Displaying list in reverse order using recursion f) If there are two linked lists for two departments, then concatenate two lists. 2. Write a C++ program to sort binary numbers with the help of doubly linked list. Write functions for a) Addition of two binary numbers b) Calculation of 1's complement of a given binary number c) Calculation of 2's complement of a given binary number

Objective(s)	
1	To study basics of linked list
2	To learn the features of linked list over array
3	To understand the concept of linked list and its representation
4	To study types of linked lists: Singly linked list, Circular linked list, Doubly linked list, Doubly circular linked list

Theory
1. Define: Linked list. Comment on: The features of linked list over array. 2. How many pointers are required for implementing a singly linked list? 3. State and explain different types of linked list. How a linked list node can be represented? 4. Compare: Singly linked list and Doubly linked list. 5. Give applications of linked list. 6. Write down the detailed algorithms for the given problem statement.

Algorithms:

a) Singly Linked List Operations:

1) Algorithm for Traversing a Linked List

```
ALGORITHM FOR TRAVERSING A LINKED LIST

Step 1: [INITIALIZE] SET PTR = START
Step 2: Repeat Steps 3 and 4 while PTR != NULL
Step 3:              Apply Process to PTR->DATA
Step 4:              SET PTR = PTR->NEXT
        [END OF LOOP]
Step 5: EXIT
```

2) Algorithm to count a no. of nodes in a liked list

```
ALGORITHM TO COUNT A NO. OF NODES IN A LINKED LIST

Step 1: [INITIALIZE] SET COUNT = 0
Step 2: [INITIALIZE] SET PTR = START
Step 3: Repeat Steps 4 and 5 while PTR != NULL
Step 4:              SET COUNT = COUNT +1
Step 5:              SET PTR = PTR->NEXT
        [END OF LOOP]
Step 5: Write COUNT
Step 6: EXIT
```

3) Algorithm for Searching a Linked List

```
ALGORITHM TO SEARCH A LINKED LIST

Step 1: [INITIALIZE] SET PTR = START
Step 2: Repeat Step 3 while PTR != NULL
Step 3:              IF VAL = PTR->DATA
                              SET POS = PTR
                              Go To Step 5
                     ELSE
                              SET PTR = PTR->NEXT
                     [END OF IF]
        [END OF LOOP]
Step 4: SET POS = NULL
Step 5: EXIT
```

4) Inserting a Node at the Beginning

```
ALGORITHM TO INSERT A NEW NODE IN THE BEGINNING OF THE LINKED LIST

Step 1: IF AVAIL = NULL, then
                    Write OVERFLOW
                    Go to Step 7
        [END OF IF]
Step 2: SET New_Node = AVAIL
Step 3: SET AVAIL = AVAIL->NEXT
Step 4: SET New_Node->DATA = VAL
Step 5: SET New_Node->Next = START
Step 6: SET START = New_Node
Step 7: EXIT
```

5) Inserting a Node at the End

```
ALGORITHM TO INSERT A NEW NODE AT THE END OF THE LINKED LIST

Step 1: IF AVAIL = NULL, then
                    Write OVERFLOW
                    Go to Step 10
          [END OF IF]
Step 2: SET New_Node = AVAIL
Step 3: SET AVAIL = AVAIL->NEXT
Step 4: SET New_Node->DATA = VAL
Step 5: SET New_Node->Next = NULL
Step 6: SET PTR = START
Step 7: Repeat Step 8 while PTR->NEXT != NULL
Step 8:                 SET PTR = PTR ->NEXT
        [END OF LOOP]
Step 9: SET PTR->NEXT = New_Node
Step 10: EXIT
```

6) Inserting a Node after Node that has Value NUM

```
ALGORITHM TO INSERT A NEW NODE AFTER A NODE THAT HAS VALUE NUM

Step 1: IF AVAIL = NULL, then
                    Write OVERFLOW
                    Go to Step 12
        [END OF IF]
Step 2: SET New_Node = AVAIL
Step 3: SET AVAIL = AVAIL->NEXT
Step 4: SET New_Node->DATA = VAL
Step 5: SET PTR = START
Step 6: SET PREPTR = PTR
Step 7: Repeat Steps 8 and 9 while PREPTR->DATA != NUM
Step 8:                 SET PREPTR = PTR
Step 9:                 SET PTR = PTR->NEXT
        [END OF LOOP]
Step 10: PREPTR->NEXT = New_Node
Step 11: SET New_Node->NEXT = PTR
Step 12: EXIT
```

7) Deleting the First Node

```
Algorithm to delete the first node from the linked list

Step 1: IF START = NULL, then
                    Write UNDERFLOW
                    Go to Step 5
        [END OF IF]
Step 2: SET PTR = START
Step 3: SET START = START->NEXT
Step 4: FREE PTR
Step 5: EXIT
```

8) Deleting the Last Node

```
ALGORITHM TO DELETE THE LAST NODE OF THE LINKED LIST

Step 1: IF START = NULL, then
                   Write UNDERFLOW
                   Go to Step 8
        [END OF IF]
Step 2: SET PTR = START
Step 3: Repeat Steps 4 and 5 while PTR->NEXT != NULL
Step 4:             SET PREPTR = PTR
Step 5:             SET PTR = PTR->NEXT
          [END OF LOOP]
Step 6: SET PREPTR->NEXT = NULL
Step 7: FREE PTR
Step 8: EXIT
```

9) Deleting the Node After a Given Node

```
ALGORITHM TO DELETE THE NODE AFTER A GIVEN NODE FROM THE LINKED LIST

Step 1: IF START = NULL, then
                    Write UNDERFLOW
                    Go to Step 10
          [END OF IF]
Step 2: SET PTR = START
Step 3: SET PREPTR = PTR
Step 4: Repeat Step 5 and 6 while PRETR->DATA != NUM
Step 5:             SET PREPTR = PTR
Step 6:             SET PTR = PTR->NEXT
        [END OF LOOP]
Step 7: SET TEMP = PTR->NEXT
Step 8: SET PREPTR->NEXT = TEMP->NEXT
Step 9: FREE TEMP
Step 10: EXIT
```

b) Circular Singly Linked List Operations:

1) Inserting a Node at the Beginning

```
Algorithm to insert a new node in the beginning of circular  linked list

Step 1: IF AVAIL = NULL, then
                  Write OVERFLOW
                  Go to Step 7
          [END OF IF]
Step 2: SET New_Node = AVAIL
Step 3: SET AVAIL = AVAIL->NEXT
Step 4: SET New_Node->DATA = VAL
Step 5: SET PTR = START
Step 6:   Repeat Step 7 while PTR->NEXT != START
Step 7:             PTR = PTR->NEXT
Step 8: SET New_Node->Next = START
Step 8: SET PTR->NEXT = New_Node
Step 6: SET START = New_Node
Step 7: EXIT
```

2) Inserting a Node at the End

```
Algorithm to insert a new node at the end of the circular linked list

Step 1: IF AVAIL = NULL, then
                   Write OVERFLOW
                   Go to Step 7
          [END OF IF]
Step 2: SET New_Node = AVAIL
Step 3: SET AVAIL = AVAIL->NEXT
Step 4: SET New_Node->DATA = VAL
Step 5: SET New_Node->Next = START
Step 6: SET PTR = START
Step 7: Repeat Step 8 while PTR->NEXT != START
Step 8:              SET PTR = PTR ->NEXT
          [END OF LOOP]
Step 9: SET PTR ->NEXT = New_Node
Step 10: EXIT
```

3) Inserting a Node after Node that has Value NUM

```
Algorithm to insert a new node after a node that has value NUM

Step 1: IF AVAIL = NULL, then
                    Write OVERFLOW
                    Go to Step 12
          [END OF IF]
Step 2: SET New_Node = AVAIL
Step 3: SET AVAIL = AVAIL->NEXT
Step 4: SET New_Node->DATA = VAL
Step 5: SET PTR = START
Step 6: SET PREPTR = PTR
Step 7: Repeat Step 8 and 9 while PTR->DATA != NUM
Step 8:             SET PREPTR = PTR
Step 9:             SET PTR = PTR->NEXT
          [END OF LOOP]
Step 10: PREPTR->NEXT = New_Node
Step 11: SET New_Node->NEXT = PTR
Step 12: EXIT
```

4) Deleting the First Node

```
Algorithm to delete the first node from the circular linked list

Step 1: IF START = NULL, then
                    Write UNDERFLOW
                    Go to Step 8
          [END OF IF]
Step 2: SET PTR = START
Step 3: Repeat Step 4 while PTR->NEXT != START
Step 4:             SET PTR = PTR->NEXT
          [END OF IF]
Step 5: SET PTR->NEXT = START->NEXT
Step 6: FREE START
Step 7: SET START = PTR->NEXT
Step 8: EXIT
```

5) Deleting the Last Node

```
Algorithm to delete the last node of the circular linked list

Step 1: IF START = NULL, then
                    Write UNDERFLOW
                    Go to Step 8
          [END OF IF]
Step 2: SET PTR = START
Step 3: Repeat Step 4  while PTR->NEXT != START
Step 4:             SET PREPTR = PTR
Step 5:             SET PTR = PTR->NEXT
          [END OF LOOP]
Step 6: SET PREPTR->NEXT = START
Step 7: FREE PTR
Step 8: EXIT
```

6) Deleting the Node After a Given Node

```
Algorithm to delete the node after a given node from the circular linked list

Step 1: IF START = NULL, then
                    Write UNDERFLOW
                    Go to Step 9
          [END OF IF]
Step 2: SET PTR = START
Step 3: SET PREPTR = PTR
Step 4: Repeat Step 5 and 6 while PREPTR->DATA != NUM
Step 5:             SET PREPTR = PTR
Step 6:             SET PTR = PTR->NEXT
          [END OF LOOP]
Step 7: SET PREPTR->NEXT = PTR->NEXT
Step 8: FREE PTR
Step 9: EXIT
```

c) Doubly Linked List Operations:

1) Inserting a Node at the Beginning

```
Algorithm to insert a new node in the beginning of the doubly linked list

Step 1: IF AVAIL = NULL, then
                    Write OVERFLOW
                    Go to Step 8
          [END OF IF]
Step 2: SET New_Node = AVAIL
Step 3: SET AVAIL = AVAIL->NEXT
Step 4: SET New_Node->DATA = VAL
Step 5: SET New_Node->PREV = NULL
Step 6: SET New_Node->Next = START
Step 7: SET START = New_Node
Step 8: EXIT
```

2) Inserting a Node at the End

```
Algorithm to insert a new node at the end of the doubly linked list

Step 1: IF AVAIL = NULL, then
                    Write OVERFLOW
                    Go to Step 11
          [END OF IF]
Step 2: SET New_Node = AVAIL
Step 3: SET AVAIL = AVAIL->NEXT
Step 4: SET New_Node->DATA = VAL
Step 5: SET New_Node->Next = NULL
Step 6: SET PTR = START
Step 7: Repeat Step 8 while PTR->NEXT != NULL
Step 8:             SET PTR = PTR->NEXT
          [END OF LOOP]
Step 9: Step 11: EXITSET PTR->NEXT = New_Node
Step 10: New_Node->PREV = PTR
```

3) Inserting a Node after Node that has Value NUM

```
Algorithm to insert a new node after a node that has value NUM

Step 1: IF AVAIL = NULL, then
                    Write OVERFLOW
                    Go to Step 11
          [END OF IF]
Step 2: SET New_Node = AVAIL
Step 3: SET AVAIL = AVAIL->NEXT
Step 4: SET New_Node->DATA = VAL
Step 5: SET PTR = START
Step 6: Repeat Step 8 while PTR->DATA != NUM
Step 7:             SET PTR = PTR->NEXT
          [END OF LOOP]
Step 8: New_Node->NEXT = PTR->NEXT
Step 9: SET New_Node->PREV = PTR
Step 10: SET PTR->NEXT = New_Node
Step 11: EXIT
```

4) Deleting the First Node

```
Algorithm to delete the first node from the doubly linked list

Step 1: IF START = NULL, then
                    Write UNDERFLOW
                    Go to Step 6
          [END OF IF]
Step 2: SET PTR = START
Step 3: SET START = START->NEXT
Step 4: SET START->PREV = NULL
Step 5: FREE PTR
Step 6: EXIT
```

5) Deleting the Last Node

```
Algorithm to delete the last node of the doubly linked list

Step 1: IF START = NULL, then
                    Write UNDERFLOW
                    Go to Step 7
          [END OF IF]
Step 2: SET PTR = START
Step 3: Repeat Step 4 and 5 while PTR->NEXT != NULL
Step 4:             SET PTR = PTR->NEXT
          [END OF LOOP]
Step 5: SET PTR->PREV->NEXT = NULL
Step 6: FREE PTR
Step 7: EXIT
```

6) Deleting the Node After a Given Node

```
Algorithm to delete the node after a given node from the doubly linked list

Step 1: IF START = NULL, then
                    Write UNDERFLOW
                    Go to Step 9
          [END OF IF]
Step 2: SET PTR = START
Step 3: Repeat Step 4 while PTR->DATA != NUM
Step 4:             SET PTR = PTR->NEXT
          [END OF LOOP]
Step 5: SET TEMP = PTR->NEXT
Step 6: SET PTR->NEXT = TEMP->NEXT
Step 7: SET TEMP->NEXT->PREV = PTR
Step 8: FREE TEMP
Step 9: EXIT
```

d) Circular Doubly Linked List Operations:

1) Inserting a Node at the Beginning

```
Algorithm to insert a new node in the beginning of the circular doubly
linked list

Step 1: IF AVAIL = NULL, then
                    Write OVERFLOW
                    Go to Step 11
          [END OF IF]
Step 2: SET New_Node = AVAIL
Step 3: SET AVAIL = AVAIL->NEXT
Step 4: SET New_Node->DATA = VAL
Step 6: SET START->PREV->NEXT = new_node;
Step 7: SET New_Node->PREV = START->PREV;
Step 8: SET START->PREV= new_Node;
Step 9: SET new_node->next = START;
Step 10: SET START = New_Node
Step 11: EXIT
```

2) Inserting a Node at the End

```
Algorithm to insert a new node at the end of the circular doubly linked
list

Step 1: IF AVAIL = NULL, then
                    Write OVERFLOW
                    Go to Step 11
          [END OF IF]
Step 2: SET New_Node = AVAIL
Step 3: SET AVAIL = AVAIL->NEXT
Step 4: SET New_Node->DATA = VAL
Step 5: SET New_Node->Next = START
Step 6: SET New_Node->PREV = START->PREV
Step 7: EXIT
```

3) Inserting a Node after Node that has Value NUM

```
Algorithm to insert a new node after a node that has value NUM

Step 1: IF AVAIL = NULL, then
                    Write OVERFLOW
                    Go to Step 11
          [END OF IF]
Step 2: SET New_Node = AVAIL
Step 3: SET AVAIL = AVAIL->NEXT
Step 4: SET New_Node->DATA = VAL
Step 5: SET PTR = START
Step 6: Repeat Step 8 while PTR->DATA != NUM
Step 7:                SET PTR = PTR->NEXT
          [END OF LOOP]
Step 8: New_Node->NEXT = PTR->NEXT
Step 9: SET PTR->NEXT->PREV = New_Node
Step 9: SET New_Node->PREV = PTR
Step 10: SET PTR->NEXT = New_Node
Step 11: EXIT
```

4) Deleting the First Node

```
Algorithm to delete the first node from the circular doubly linked list

Step 1: IF START = NULL, then
                    Write UNDERFLOW
                    Go to Step 8
          [END OF IF]
Step 2: SET PTR = START
Step 3: SET PTR->PREV=>NEXT= PTR->NEXT
Step 4: SET PTR->NEXT->PREV = PTR->PREV
Step 5: SET START = START->NEXT
Step 6: FREE PTR
Step 7: EXIT
```

5) Deleting the Last Node

```
Algorithm to delete the last node of the circular doubly linked list

Step 1: IF START = NULL, then
                    Write UNDERFLOW
                    Go to Step 8
          [END OF IF]
Step 2: SET PTR = START->PREV
Step 5: SET PTR->PREV->NEXT = START
Step 6: SET START->PREV = PTR->PREV
Step 7: FREE PTR
Step 8: EXIT
```

6) Deleting the Node After a Given Node

```
Algorithm to delete the node after a given node from the circular doubly
linked list

Step 1: IF START = NULL, then
                    Write UNDERFLOW
                    Go to Step 9
          [END OF IF]
Step 2: SET PTR = START
Step 3: Repeat Step 4 while PTR->DATA != NUM
Step 4:             SET PTR = PTR->NEXT
          [END OF LOOP]
Step 5: SET TEMP = PTR->NEXT
Step 6: SET PTR->NEXT = TEMP->NEXT
Step 7: SET TEMP->NEXT->PREV = PTR
Step 8: FREE TEMP
Step 9: EXIT
```

Conclusion