**Algorithms**

**Algorithm to insert a node at a particular position:**

Step 1: IF AVAIL = NULL, then

Write OVERFLOW

Go to Step 8

[END OF IF]

Step 2: IF K=1, then

SET New\_Node = AVAIL //allocate space for new node.

SET AVAIL = AVAIL->NEXT

SET New\_Node->DATA = VAL //Set data part with given value

SET New\_Node->Next = START //Next part initialized with the addr of first node of list

SET START = New\_Node

[END OF IF]

GOTO Step 8

Step 3: IF K=total\_number\_of\_nodes+1, then

SET New\_Node = AVAIL

SET AVAIL = AVAIL->NEXT

SET New\_Node->DATA = VAL

SET New\_Node->Next = NULL

SET PTR = START

Repeat the next step while PTR->NEXT != NULL

SET PTR = PTR ->NEXT

[END OF LOOP]

SET PTR->NEXT = New\_Node

[END OF IF]

GOTO Step 8

Step 4: SET PTR = START

Step 5: Repeat the step 6 and 7 while PTR->NEXT != NULL

Step 6: IF POS==K-1, then

SET New\_Node = AVAIL

SET AVAIL = AVAIL->NEXT

SET New\_Node->DATA = VAL

New\_Node->NEXT = PTR->NEXT

PTR->NEXT=New\_Node;

[END OF IF]

GOTO Step 8

Step 7: POS=POS+1

[END OF LOOP]

Step 8: EXIT

**Algorithm to insert a node after a given value:**

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

**Algorithm to insert a node before a given value:**

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 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

**Algorithm to insert a node at the beginning:**

Step 1: IF AVAIL = NULL, then

Write OVERFLOW

Go to Step 7

[END OF IF]

Step 2: SET New\_Node = AVAIL //allocate space for new node.

Step 3: SET AVAIL = AVAIL->NEXT

Step 4: SET New\_Node->DATA = VAL //Set data part with given value

Step 5: SET New\_Node->Next = START //Next part initialized with the addr of first node of list

Step 6: SET START = New\_Node // Make new node as a Start node of list

Step 7: EXIT

**Algorithm to insert a node at the end:**

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

**Algorithm to delete a node at a particular position:**

Step 1: IF START = NULL, then

Write UNDERFLOW

Go to Step 10

[END OF IF]

Step 2: SET PREPTR=START

Step 3: SET PTR=START

Step 4: Repeat steps 5, 6, 7, 8 and 9 WHILE PTR!=NULL

Step 5: IF POS=K AND K=1, then

START=START->NEXT

[END OF IF]

GOTO Step 10

Step 6: IF POS=K, then

PREPTR->NEXT = PTR->NEXT

[END OF IF]

GOTO Step 10

Step 7: IF POS!=1, then

PREPTR = PTR

[END OF IF]

Step 8: PTR=PTR->NEXT

Step 9: POS=POS+1

[END OF LOOP]

Step 10: EXIT

**Algorithm to create a linked list:**

Step 1: INPUT the number of nodes N

Step 2: SET I=1

Step 3: Repeat Step 4, 5 and 6 while I<=N

Step 4: INPUT value to be entered in the node

Step 5: INSERT the value at the end of the list by calling the respective function

Step 6: I=I+1

Step 7: SET START = first node of this created list

Step 8: END

Algorithm to delete a node at the beginning:

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

**Algorithm to delete a node at the end:**

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

**Algorithm to delete a node after a given value:**

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 PREPTR->DATA != NUM

Step 5: SET PREPTR = PTR

Step 6: SET PTR = PTR->NEXT

[END OF LOOP]

Step7: SET TEMP = PTR

Step 8: SET PREPTR->NEXT = TEMP->NEXT

Step 9: FREE TEMP

Step 10: EXIT

**Algorithm to delete a node before a given value:**

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 PTR->NEXT->DATA != NUM

Step 5: SET PREPTR = PTR

Step 6: PTR=PTR->NEXT

[END OF LOOP]

Step 7: PREPTR->NEXT=PTR->NEXT

Step 8: FREE PTR

Step 9: EXIT

**Algorithm to update a node at a particular position:**

Step 1: IF START = NULL, then

Write UNDERFLOW

Go to Step 10

[END OF IF]

Step 2: IF K=1, then

START->DATA=VAL

[END OF IF]

Step 3: SET PTR=START

Step 4: Repeat steps 5, 6, 7, 8 and 9 WHILE PTR!=NULL

Step 5: IF POS=K, then

PTR->DATA=VAL

[END OF IF]

GOTO Step 10

Step 8: PTR=PTR->NEXT

Step 9: POS=POS+1

[END OF LOOP]

Step 10: EXIT

**Algorithm to update a node at the beginning:**

Step 1: IF START = NULL, then

Write UNDERFLOW

Go to Step 3

[END OF IF]

Step 2: START->DATA = VAL

Step 3: EXIT

**Algorithm to update a node at the end:**

Step 1: IF START = NULL, then

Write UNDERFLOW

Go to Step 6

[END OF IF]

Step 2: SET PTR = START

Step 3: Repeat Step 4 WHILE PTR->NEXT!=NULL

Step 4: PTR=PTR->NEXT

[END OF LOOP]

Step 5: PTR->DATA = VAL

Step 6: EXIT

**Algorithm to update a node after a given value:**

Step 1: IF START = NULL, then

Write UNDERFLOW

Go to Step 6

[END OF IF]

Step 2: SET PTR = START and PREPTR=START

Step 3: Repeat Step 4 and 5 WHILE PREPTR->DATA!=K

Step 4: PREPTR=PTR

Step 5: PTR=PTR->NEXT

[END OF LOOP]

Step 5: PTR->DATA = VAL

Step 6: EXIT

**Algorithm to update a node before a given value:**

Step 1: IF START = NULL, then

Write UNDERFLOW

Go to Step 6

[END OF IF]

Step 2: SET PTR = START and PREPTR=START

Step 3: Repeat Step 4 and 5 WHILE PTR->DATA!=K

Step 4: PREPTR=PTR

Step 5: PTR=PTR->NEXT

[END OF LOOP]

Step 5: PREPTR->DATA = VAL

Step 6: EXIT

**Algorithm to search for a particular node:**

Step 1: [INITIALIZE] SET PTR = START

Step 2: Repeat Step 3 while PTR != NULL

Step 3: IF VAL = PTR->DATA

PRINT ‘ELEMENT FOUND’

Go To Step 5

ELSE

SET PTR = PTR->NEXT

[END OF IF]

[END OF LOOP]

Step 4: PRINT ‘ELEMENT NOT FOUND’ //if search is unsuccessful

Step 5: EXIT

**Algorithm to reverse a linked list:**

Step 1: IF START = NULL, then

Write UNDERFLOW

Go to Step 12

[END OF IF]

Step 2: prevNode = START

Step 3: START = START->NEXT

Step 4: curNode = START

Step 5: prevNode->NEXT = NULL

Step 6: Repeat Steps 7, 8, 9 and 10 while START!= NULL

Step 7: START=START->NEXT

Step 8: curNode->NEXT = prevNode

Step 9: prevNode = curNode

Step 10: curNode = START

[END OF LOOP]

Step 11: START = prevNode

Step 12: EXIT

**Algorithm to count nodes in a linked list:**

Step 1: [INITIALIZE] SET = 0

Step 2: [INITIALIZE] SET PTR = START

Step 3: Repeat Steps 4 and 5 while PTR != NULL

Step 4: SET=+ 1

Step 5: SET PTR = PTR NEXT

[END OF LOOP]

Step 6: Write COUNT

Step 7: EXIT

**Algorithm to display the elements of a linked list:**

Step 1: [INITIALIZE] SET PTR = START

Step 2: Repeat Steps 3 and 4 while PTR != NULL

Step 3: PRINT PTR->DATA

Step 4: SET PTR = PTR->NEXT

[END OF LOOP]

Step 5: EXIT

**Algorithm to merge two linked list:**

Step 1. REPEAT these steps while traversing both lists

IF LIST1->DATA < LIST2->DATA

Add LIST1->DATA to new LIST

LIST1=LIST1->NEXT

IF LIST2->DATA < LIST1->DATA

Add LIST2->DATA to new LIST

LIST2=LIST2->NEXT

Step 2: IF LIST1 still has nodes remaining,

Add all of them in the new LIST

Step 3: IF LIST2 still has nodes remaining,

Add all of them in the new LIST

Step 4: Return the resultant LIST

**Algorithm to sort a linked list:**

Step 1: IF START = NULL, then

Write UNDERFLOW

Go to Step 9

[END OF IF]

Step 2: SET TEMP1 = START

Step 3: Repeat Steps 4, 5, 6, 7 and 8 while TEMP1 != NULL

Step 4: SET TEMP2 = TEMP1->NEXT

Step 5: Repeat Steps 6 and 7 while TEMP2 != NULL

Step 6: IF TEMP2->DATA < TEMP1->DATA, then

Swap TEMP1->DATA and TEMP2->DATA

[END OF IF]

Step 7: TEMP2=TEMP2->NEXT

[END OF INNER LOOP]

Step 8: TEMP1=TEMP1->NEXT

[END OF OUTER LOOP]

Step 9: EXIT

**Algorithm to concatenate a linked list:**

Step 1: IF LIST1= NULL, then

RETURN LIST2

Go to Step 8

[END OF IF]

Step 2: IF LIST2= NULL, then

RETURN LIST1

Go to Step 8

[END OF IF]

Step 3: SET PTR = LIST1

Step 4: Repeat Step 5 while PTR->NEXT != NULL

Step 5: SET PTR = PTR->NEXT

[END OF LOOP]

Step 6: PTR->NEXT=LIST2

Step 7: RETURN LIST1

Step 8: EXIT