5) WAP to Implement singly linked list with following operations

a) Create a Linked List

b) Deletion of first element, specified element and last element in the list.

c) Display the contents of the linked list.

Pseudocode :

```
Function deleteFirst ( )
    If head == NULL THEN
    PRINT "List is empty"
     RETURN
  END IF

temp ← head
head ← head.next
Free temp
Print "first node deleted"
END Function

Function deleteLast ( )
  If head == NULL THEN
  Print "List is empty"
  Return
  End if
If head.next == NULL THEN
 Print "Deleted element: ", head.data
 Free head
 head ← NULL
 Return
End if
```

```
temp ← head
While temp.next.next != NULL DO
    temp ← temp.next
END While

Print "Deleted element:", temp.next.data
Free temp.next
temp.next ← NULL
ENDD Function

Function deleteValue (value)
    IF head == NULL THEN
    Print "List is empty"
    Return
    End if
IF head.data == value THEN
    temp ← head
    head ← head.next
    Free temp
    Print "Element deleted :", value
    Return
    End if

    temp ← head
    While temp.next != NULL AND
temp.next.data != value DO
    temp ← temp.next
END WHILE
```

IF temp.next = = NULL THEN
Print "Element not found"
Return
END IF

delNod ← temp.next
temp.next ← delNode.next
Free delNode
Print "Element deleted :", value
END Function

MG
17/11/25

code:

```
Void delete first ()
{
    if (head == NULL) {
        printf ("List is empty\n");
        return;
    }
    Struct Node * temp = head;
    head = head -> next;
    free (temp);
    printf ("First node = deleted.\n");

void deleteLast( )
{
    if (head == NULL) {
        printf ("List is empty\n");
        return;
    }
    if (head -> next == NULL) {
        free (head);
```

Scanned with CamScanner

```
printf("Last node deleted.\n");
return;
}
struct Node *temp = head, *prev = NULL;
while (temp->next != NULL) {
    prev = temp;
    temp = temp->next;
}
prev->next = NULL;
free(temp);
printf("Last node deleted.\n");

void deleteValue(int value)
    if (head == NULL) {
        printf("List is empty\n");
        return;
    }
    struct Node *temp = head, *prev = NULL;
    if (head->data == value) {
        head = head->next;
        free(temp);
        printf("Node with value %d deleted.\n",
               value);
        return;
    }
```

Scanned with CamScanner

```
while (temp != NULL && temp->data != value){
    prev = temp;
    temp = temp->next;
}
if (temp == NULL) {
    printf ("Value not found.\n");
    return;
}
prev->next = temp->next;
free (temp);
printf ("Node with value %d deleted.\n",
                                value);
}

void display()
{
    struct Node * temp = head;
    while (temp != NULL) {
        printf("%d ", temp->data);
    }
    printf("\n");
}
int main() {
    int n, choice, data, pos, value;
    printf ("Enter number of initials nodes to create:");
    scanf ("%d", &n);
    printf ("Enter %d elements:", n);
    create (n);
```

```
printf("\n LinkedList Menu\n");
printf("1. Insert at Begining \n");
printf("2. Insert at End \n");
printf("3. Insert at Position\n");
printf("4. Display \n");
printf("5. Delete first Node\n");
printf("6. Delete Last Node \n");
printf("7. Delete Node by Value \n");
printf("8. Exit \n");
while (1) {
    printf("\n Enter your choice : ");
    sconf("%d", &choice);
switch (choice) {
    case 1:
        printf("Enter data : ");
        scanf("%d", &data);
        insert At Beginning (data);
        break;
    case 2;
        printf("Enter data : ");
        scanf("%d &data);
        insert At End (data);
        break;
    case 3;
        printf("Enter data : ");
        scanf("%d", &data);
```

```
        printf("Enter position: ");
        scanf("%d", &pos);
        insertAtPosition(data, pos);
        break;
    case 4:
        printf("Linked List:");
        display();
        break;
    case 5:
        deleteFirst();
        break;
    case 6:
        deleteLast();
        break;
    case 7:
        printf("Enter value to delete: ");
        scanf("%d", &value);
        deleteValue(value);
        break;
    case 8:
        exit(0);
    default:
        printf("Invalid choice\n");
    }
}
return 0;
}
```

Output:

```
Enter number of initial nodes to create : 4
Enter 4 elements: 1 2 3 4

Linked List Menu
1. Insert at Beginning.
2. Insert at End.
3. Insert at Position.
4. Display.
5. Delete First Node.
6. Delete Last Node.
7. Delete Node by Value
8. Exit.

Enter your choice : 5
First Node deleted.
Enter your choice : 6
Last node deleted.
Enter your choice : 7
Enter value to delete : 2
Node with value 2 deleted.

Enter your choice : 4
Linked List : 3
Enter your choice : 8

Process returned 0.
```

MG
24/11/25.