Aim : Write a C program in C to implement linked list ? Implement a Program for performing the operation as menu driven program for the insertion, Deletion & display function respectively.

Objective : To implement menu driven program in linked list, for performing operations like insertion, deletion & display.

Aim : Write a C Program in C to implement linked list ? Implement a Program for performing the operation as a menu Driven program for the insertion, Deletion & Display Function respectively.

Objective : To implement menu driven program in linked list, for performing operations like insertion, deletion & Display.

Theory : A linked list is a data Structure consisting of nodes, where each node points to the next node in the sequence.

The Basic types of linked list are :

i) Single Linked List
ii) Double Linked List
iii) Circular Linked List

Common operations on linked list are :

1) Insertion :

- At the beginning : Add a new node at the start of the list.
- At the End : Append a new node at the end of the List.
- In the middle : Insert a new node at a specific position.

2> Deletion :

- At the beginning : Remove the first node.
- At the end : Delete the last node.
- In the middle : Remove a node from a Sperific position.

3> Display :

- Display the data in specific nodes
- Display complete linked list.

Code :

```
#include<stdio.h>
#include<stdlib.h>

struct node {
  int info;
 struct node * link;
};
struct node* start = NULL;

void display () {
  struct node* temp;
if (start == NULL) {
  printf("\n The list is Empty \n");
}
temp = start;
```

```
printf("\n The linked list is :");
while (temp != NULL) {
    printf ("%d -> ", temp->info);
    temp = temp -> link;
}
printf("\n NULL");
}

void createlist () {
  if (start == NULL) {
    int n;
    printf ("Enter the number of nodes : \n");
    scanf ("%d", &n );

    if (n != 0) {
    int data ;
    struct node* newnode;
    struct node* temp;
    newnode = malloc (sizeof (struct node ));
    start = newnode ;
    temp = start ;

    printf ("\n Enter number to be inserted ");
    scanf("%d", &data );
    start -> info = data;
```

```
for(int i = 2 ; i <= n ; i++ ){
 newnode = malloc(sizeof(struct node ));
 temp --> link = newnode ;
 printf ("\n Enter number to be inserted \n");
 Scanf (" %d", &data);
 newnode -> info = data;
 temp = temp -> link;
 }
 }
 printf ("The list is created \n");
}
 else {
        printf("The list is already created \n");
}

void insertAtFront( ){
    int data;
    Struct node* temp;
    temp = malloc (sizeof(Struct node));
    printf("\n Enter number to be
              inserted");
   Scanf (" %d", &data );
    temp -> info = data;
    temp -> link = start ;
    start = temp ;
}
```

```
void insertAtEnd ( ) {
    int data;
    struct node *temp , *head ;
    temp = malloc (sizeof (struct node ));
    printf ("\n Enter number to be
                inserted ");
    scanf ("%d", & data );

    temp --> link = 0 ;
    temp --> info = data ;
   head = start ;
    while ( head -> link != NULL) {
          head = head -> link ;
       }
   head --> link = temp ;
}

void insert_Atposition ( ) {
    struct node * temp, * newnode ;
    int pos , data , i =1 ;
     newnode = malloc (sizeof (struct node));
    printf ("\n Enter position & data to
                be inserted ");
    scanf ("%d %d", &pos , &data );
```

```
temp = Start ;
newnode -> info = data ;
newnode -> link = 0 ;
while ( i < pos - 1 ) {
        temp = temp -> link ;
        i++ ;
}
  newnode -> link = temp -> link ;
  temp -> link = newnode ;
}

void deletefirst () {
    Struct node* temp ;
    if (start == NULL) {
        printf("In List is empty ");
    } else {
            temp = start ;
            start = start -> link ;
            free (temp);
        }
    }

void deleteEnd () {
    struct node *temp , *prevnode ;
```

```
if (start == NULL) {
    printf("List is empty \n");
} else {
        temp = start ;
    while (temp -> link != 0) {
        prevnode = temp ;
        temp = temp -> link ;
    }
    free (temp);
    prevnode -> link = 0;
}
}
void deletePosition () {
    struct node *temp, *position ;
    int i = 1, pos ;

    if (start == NULL) {
        printf ("\n List is Empty ");
    } else {
        printf("Enter index no: ");
        Scanf ("%d", &pos);
        position = malloc (sizeof (struct node));
        temp = start ;
        while (i < pos - 1) {
            temp = temp -> link ;
            i++;
        }
```

```
        position = temp -> link;
        temp -> link = position -> link;
        free(position);
        }
    }
int main () {
    createlist();
    int ch;
    while (1){
        printf("\t 1. Insert at start\n");
        printf("\t 2. Insert at End \n");
        printf("\t 3. Insert at any Position \n");
        printf("\t 4. Delete at start \n");
        printf("\t 5. Delete at End \n");
        printf("\t 6. Delete at any position \n");
        printf("\t 7. Display linked List \n");
        printf(" exit \n");

        printf("Enter your choice \n");
        scanf("%d", &ch);

        switch(ch) {
            case 1: insertAtfront();
                    break;
```

```
        case 2: insertAtEnd ();
                break;
        case 3: insertAtposition ();
                break;
        case 4: insertdeletefirst();
                break;
        case 5: deleteEnd();
                break;
        case 6: deletePosition ();
                break;
        case 7: display ();
                break;
        case 8: exit (1);
                break;
        default : printf ("Invalid choice\n");
            }
        }
    }
```

Output:

```
Enter the number of nodes : 4
Enter number to be inserted : 10
Enter number to be inserted : 20
Enter number to be inserted : 30
Enter number to be inserted : 40
```

The list is Created

1. For Insertion at starting
2. For Insertion at End
3. For Insertion at any position.
4. For deleting First element
5. For deleting Last element
6. For deleting a element at any position.
7. Display the linked list
8. To Exit.

Enter Choice 7

The linked list is 10 → 20 → 30 → 40 → NULL

⋮

Algorithm: i] Struct Defination

Define a structure node with two feilds, "info" to store data & "link" to store the address of next node.

ii] Global variables

Declare a global variable "Start" to keep track of beginning of linked list.

iii] Display List Function

Define a Function display, that traverses the linked list.

Conclusion : Program of linked list is implemented & executed successfully using C programming Langauge.

iv. Create a list Function :

v. Create a function insertAtFront

vi. Create a function insertAtEnd

vii. Create a function insertAtPosition.

viii. Create a function deleteFirst

ix. Create a function deleteEnd

x. Create a function deletePosition

xi. All those function will implement it's respective work acc^{n} to it's name.

xii. Main Function :

This function invokes the other functions.

Conclusion: Program of Linked List is implemented & executed successfully using C Programming Langauge.