LINKED LIST

AIM:

To create a single linked list that will create,insert using struct.

ALGORITHM:

· Include the required header files stdio.h and stdlib.h.

· Define a structure LinkedList with members data and next.

· Declare functions for node creation, front insertion, back insertion, and list display.

· Read an integer value from the user to create the head node.

· Allocate memory dynamically using malloc for the head node.

· Store the input value in the data field and set next to NULL.

· Display the linked list.

· Read a value from the user for insertion at the front.

· Create a new node and link it before the head node.

· Update the head pointer to the new node.

· Display the updated linked list.

· Read a value from the user for insertion at the back.

· Traverse the linked list until the last node is reached.

· Link the new node to the end of the list.

· Display the final linked list.

PROGRAM:

```c
/*
*  Linked List Implementation in C
*  Author   : MUTHUGANESH S
*  Date     : 18/01/2026
*  Filename : LinkedList.c
*  retval   : void
*/

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

struct LinkedList
{
    int data;
    struct LinkedList* next;
```

```c
};

struct LinkedList* CreateNewNode(int Data);
void InsertFront(struct LinkedList** Head, int Data);
void InsertBack(struct LinkedList** Head, int Data);
void PrintList(struct LinkedList* Head);

int main(void){
    int Value;

    printf("Enter the value to create linked list: ");
    scanf("%d",&Value);

    // Create the head node using the function
    struct LinkedList* Head=CreateNewNode(Value);
    PrintList(Head);

    // Insert a new node at the front
    printf("\nEnter the value to insert at front: ");
    scanf("%d",&Value);

    InsertFront(&Head,Value);
    PrintList(Head);

    // Insert a new node at the back
    printf("\nEnter the value to insert at back: ");
    scanf("%d",&Value);

    InsertBack(&Head,Value);

    // Print the linked list
    PrintList(Head);

}

// Function to create a new node
struct LinkedList* CreateNewNode(int Data){

    struct LinkedList* NewNode = (struct LinkedList*)malloc(sizeof(struct
LinkedList));

    NewNode->data = Data;
    NewNode->next = NULL;

    return NewNode;
}

// Insert a new node at the front of the linked list
void InsertFront(struct LinkedList** Head, int Data){

    struct LinkedList* NewNode = CreateNewNode(Data);

    NewNode->next = *Head;
    *Head = NewNode;
}
```

```c
// Insert a new node at the back of the linked list
void InsertBack(struct LinkedList** Head, int Data){
    struct LinkedList* NewNode = CreateNewNode(Data);

    if (*Head == NULL) {
        *Head = NewNode;
        return;
    }
    struct LinkedList* Temp = *Head;

    while (Temp->next != NULL) {
        Temp = Temp->next;
    }
    Temp->next = NewNode;
}

// Function to print the linked list
void PrintList(struct LinkedList* Head){

    struct LinkedList* Temp = Head;

    printf("\nLinked List: ");

    while (Temp != NULL) {
        printf("%d -> ", Temp->data);
        Temp = Temp->next;
    }
    printf("NULL\n");
}
```

OUTPUT: