Name : Sakshi . P. Khandoba
Sem : 3rd Section : C
Batch : 2
USN : 1BM19CS139

papergrid
Date: 23 / 11 / 20

# LAB PROGRAM 7

Write a program to implement Linked List to
a) Insert a node at the front of the list
b) Insert a node at the end of the list
c) Insert a node after a given node (at a position)
d) Delete a node at the front of the list
e) Delete a node at the end of the list
f) Delete a node after a given node (specified position)
g) Display the linked list

```
#include<stdio.h>
#include<stdlib.h>
struct node
{
   int data;
   struct node *next;
};
struct node *head;
void beg_insert()
{
   struct node *ptr;
   int item;
   ptr = (struct node *) malloc (sizeof (struct node *));
   if(ptr == NULL)
   {
      printf(" List overflow.");
   }
   else
   {
      printf("\nEnter the value:");
      scanf("%d", &item);
      ptr -> data = item;
```

```
        ptr -> next = head;
        head = ptr;                    at the front
        printf("\n Node is inserted");
    }
}
void end_insert()
{
    struct node *ptr, *temp;
    int item;
    ptr = (struct node *)malloc(sizeof(struct node));
    if (ptr == NULL)
        printf(" List Overflow");
    else
    {
        printf("\n Enter the value : ");
        scanf("%d", &item);
        ptr -> data = item;
        if (head == NULL)
        {
            ptr -> next = NULL;
            head = ptr;
            printf(" Node is inserted.");
        }
        else
        {
            temp = head;
            while (temp -> next != NULL)
            {
                temp = temp -> next;
            }
            temp -> next = ptr;
            ptr -> next = NULL;          at the end
            printf("\n Node is inserted");
        }
    }
}
```

```
void random_insert()
{
    int i, loc, item;
    struct node *ptr, *temp;
    ptr = (struct node *) malloc (sizeof (struct node));
    if(ptr == NULL)
        printf (" List overflow. ");
    else
    {
        printf(" \n Enter the value : ");
        scanf("%d", &item);
        ptr -> data = item;
        printf(" \n Enter the location of the list
          after which you want to insert : ");
        scanf("%d", &loc);
        temp = head;
        for(i=0; i<loc; i++)
        {
            temp = temp -> next;
            if (temp == NULL)
            {
                printf(" \n Cant Insert node ");
                return;
            }
        }
        ptr -> next = temp -> next;
        temp -> next = ptr;                    at the specified position
        printf(" \n Node is inserted^");
    }
}
void beg_delete()
{
    struct node *ptr;
```

```
if (head == NULL)
{
    printf (" List is empty ");
}
else
{
    ptr = head;
    head = ptr -> next;
    free (ptr);
    printf (" \n Node is deleted from the front ");
}
}
void end_delete()
{
    struct node *ptr, *ptr1;
    if (head == NULL)
        printf("\n List is empty. ");
    else if (head -> next == NULL)
    {
        head = NULL;
        free(head);
        printf (" The only node of the list is deleted");
    }
    else
    {
        ptr = head;
        while (ptr -> next != NULL)
        {
            ptr1 = ptr;
            ptr = ptr -> next;
        }
        ptr1 -> next = NULL;
        free (ptr);
```

```
        printf ("\n Node is deleted from the end");
    }
}
Void random_delete()
{
    struct node *ptr, *ptr1;
    int loc, i;
    printf ("\n Enter the location of the node after
            which you want to perform deletion ");
    scanf ("%d", &loc);
    ptr = head;
    for (i=0; i<loc ; i++)
    {
        ptr1 = ptr;
        ptr = ptr ->next;
        if( ptr == NULL)
        {
            printf ("\n Can't delete the node ");
            return;
        }
    }
    ptr1 -> next = ptr -> next;
    free (ptr);
    printf ("\n Node is deleted from the position %d",
                                        loc+1);
}
void display()
{
    struct node * ptr;
    ptr = head;
    if (ptr == NULL)
        printf (" Nothing to print. List is empty");
    else
```

```
    {
        printf("\n List values are: ");
        while(ptr != NULL)
        {
            printf("\n %d", ptr -> data);
            ptr = ptr -> next;
        }
    }
}
void main()
{
    int choice = 0;
    while (choice != 8)
    {
        printf("Choose an option");
        printf("\n1. Insert at front \n2: Insert
        at end \n3. Insert at specified location
        \n4. Delete from front \n5. Delete from
        end \n6: Delete from specified location
        \n7: Display list contents \n8: Exit ");
        printf("Enter your choice: ");
        scanf("%d", &choice);
        switch(choice)
        {
            case 1: beg_insert();
                    break;
            case 2: end_insert();
                    break;
            case 3: random_insert();
                    break;
            case 4: beg_delete();
                    break;
            case 5: end_delettte();
                    break;
```

```
        case 6: random_delete();
                break;
        case 7: display();
                break;
        case 8: exit();
        default: exit();
    }
  }
}
```