Institute of Computer Technology
B. Tech Computer Science and Engineering
Subject: DS (2CSE302)

PRACTICAL-14

AIM: - Implement the scenario based on linked list.

(This practical is in continuation of previous practical scenario, so append code of below scenario along with previous practical)

Ques. Inspire.Pvt.Ltd company is selling soft toys for children. Ridham is working in HR department of Inspire.Pvt.Ltd company. She needs to take order from the client however sometimes clients are asked to her for prepare details in ascending order after given order of some items. So as per the requirement of client she needs to the details of order as per following scenario. Refer it and provide appropriate solution for it:

1. Insert new item at the first position
2. Insert new item at the last position
3. Insert new item before the given value of node
4. Insert new item after the given value of node
5. Search item from the list
6. Display all elements of the list
7. Count total number of items
8. Delete item from the first position
9. Delete item from the last position
10. Delete particular item
11. Delete item before the given item
12. Delete item after the given item

Reference:

```
Enter the choice:
6
List is:
12
18
23
25
Enter the choice:
1
Insert item which you want to inset at first position:
10
Enter the choice:
```

2
Insert item which you want to insert at last position:
32
Enter the choice:
3
Insert item which you want to insert:
16
Insert item before where you want to insert:
18
Enter the choice:
6
List is:
10
12
16
18
23
25
32
Enter the choice:
8
Deleted item from the first position
Enter the choice:
9
Deleted item from the last position
Enter the choice:
10
Insert item which you want to delete:
16
Enter the choice:
11
Insert item before which you want to delete the item:
23
Enter the choice:
12
Insert item after which you want to delete the item:
12
Enter the choice:

6
List is:
12
25

SOLUTION

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

struct node
{
   int data;
   struct node *next;
};
struct node *head;

//Function Declaration
void insert_F();
void insert_L();
void insert_B();
void insert_A();
void search();
void display();
void count();
void delete_F();
void delete_L();
void delete_pr();
void delete_B();
void delete_A();

//functions
void insert_F()
{
        printf("\n+++++| INSERT FIRST |+++++\n");
        struct node *ptr;
   int item;
   ptr = (struct node *) malloc(sizeof(struct node *));
   if(ptr == NULL)
   {
      printf("\nOVERFLOW");
   }
   else
   {
      printf("\nEnter value you want to insert: ");
```

```
        scanf("%d",&item);
        ptr->data = item;
        ptr->next = head;
        head = ptr;
        printf("\nNode inserted");
    }
}

void insert_L()
{
        printf("\n+++++| INSERT LAST |+++++\n");
        struct node *ptr,*temp;
    int item;
    ptr = (struct node*)malloc(sizeof(struct node));
    if(ptr == NULL)
    {
        printf("\nOVERFLOW");
    }
    else
    {
        printf("\nEnter value you want to insert: ");
        scanf("%d",&item);
        ptr->data = item;
        if(head == NULL)
        {
            ptr -> next = NULL;
            head = ptr;
            printf("\nNode inserted");
        }
        else
        {
            temp = head;
            while (temp -> next != NULL)
            {
              temp = temp -> next;
            }
            temp->next = ptr;
            ptr->next = NULL;
            printf("\nNode inserted");

        }
    }
}
```

```
void insert_B()
{
        printf("\n+++++| INSERT BEFORE |+++++\n");
        int yash, prajapati;
        struct node *new_node, *ptr, *previous;
        printf("Insert item which you want to insert: ");
        scanf("%d", &yash);
        printf("Insert item before where you want to insert: ");
        scanf("%d", &prajapati);
        new_node = (struct node *)malloc(sizeof(struct node));
        new_node->data = yash;
        ptr = head;
        while (ptr->data != prajapati)
        {
                previous = ptr;
                ptr = ptr->next;
        }
        previous->next = new_node;
        new_node->next = ptr;
}

void insert_A()
{
        printf("\n+++++| INSERT AFTER |+++++\n");
        int yash,prajapati;
   struct node *new_node,*ptr,*after;
   printf("Insert item which you want to insert: ");
   scanf("%d",&yash);
   printf("Insert item after where you want to insert: ");
   scanf("%d",&prajapati);
   new_node=(struct node *)malloc(sizeof(struct node));
   new_node->data = yash;
   ptr = head;

   while(ptr->data != prajapati)
   {
      ptr = ptr->next;
   }
   new_node->next = ptr->next;
        ptr->next = new_node;
}

void search()
{
```

```
        int YASH,pos=0;
    struct node *ptr;
    printf("\nInsert item which you want to search: ");
    scanf("%d",&YASH);
    ptr=head;
    while(ptr->next!=NULL)
    {
       if(ptr->data==YASH)
       {
          printf("\nItem is available at the %d position of the list",pos+1);
                        break;
       }
       else
       {
          pos++;
          ptr=ptr->next;
       }
    }
}

void display()
{
        struct node *ptr;
    ptr = head;
    if(ptr == NULL)
    {
       printf("Empty !!!");
    }
    else
    {
       printf("\n+++++| D I S P L A Y |+++++\n\n");
       while (ptr!=NULL)
       {
          printf("%d ",ptr->data);
          ptr = ptr -> next;
       }
                printf("\n");
    }
}

void count()
{
        int count=1;
    struct node *ptr;
```

```
    ptr=head;
    while(ptr->next!=NULL)
    {
        count++;
        ptr=ptr->next;
    }
    printf("Total number of items are:\n%d",count);
}

void delete_F()
{
        struct node *ptr;

    if(head == NULL)
    {
        printf("\nEmpty!!!");
    }
    else
    {
        ptr = head;
        head = head->next;

        printf("\nData deleted: %d\n", ptr->data);

        free(ptr);

        printf("\nSuccessfully deleted first node.\n");
    }
}

void delete_L()
{
        struct node *ptr, *temp;

    if(head == NULL)
    {
        printf("\nEmpty!!!");
    }
    else
    {
        ptr = head;
        temp = head;

        while(ptr->next != NULL)
```

```
        {
           temp = ptr;
           ptr = ptr->next;
        }

        if(ptr == head)
        {
           head = NULL;
        }
        else
        {
           temp->next = NULL;
        }

        free(ptr);

        printf("\nSuccessfully deleted last node.\n");
     }
}

void delete_pr()
{
        if(head!=NULL)
        {
                int Yash;
                printf("\nEnter value to delete: ");
                scanf("%d",&Yash);
                struct node *new_node,*ptr;
                new_node=(struct node *)malloc(sizeof(struct node));
                new_node=head;
                ptr=head;
                while(new_node->data!=Yash)
                {
                        ptr=new_node;
                        new_node=new_node->next;
                }
                ptr->next=new_node->next;
                printf("\nDeleted data: %d",Yash);
                free(new_node);
        }
        else
        {
                printf("\nEmpty!!!\n");
        }
```

```
}

void delete_B()
{
        if(head!=NULL)
        {
   struct node *new_node,*ptr,*temp1,*temp2;
   new_node=(struct node *)malloc(sizeof(struct node));
   new_node=head;
        int yash;
        printf("\nEnter value before which you have delete node: ");
        scanf("%d",&yash);
        while(new_node->data!=yash)
        {
                ptr=new_node;
                new_node=new_node->next;
        }
        temp1=head;
        while(temp1!=ptr)
        {
                temp2=temp1;
                temp1=temp1->next;
        }
        temp2->next=new_node;
        printf("\nDeleted data: %d",ptr->data);
        free(ptr);
}
else
{
        printf("\nEmpty!!!\n");
}
}
void delete_A()
{
        if(head!=NULL)
        {
                struct node *new_node,*ptr,*temp1;
                new_node=(struct node *)malloc(sizeof(struct node));
                new_node=head;
                int yash;
                printf("\nEnter value after which you have delete node: ");
                scanf("%d",&yash);
                while(new_node->data!=yash)
                {
```

```
                new_node=new_node->next;
            }
            ptr=new_node->next;
            temp1=ptr->next;
            new_node->next=temp1;
            printf("\nDeleted data: %d",ptr->data);
            free(ptr);
        }
        else
        {
            printf("\nEmpty!!!\n");
        }
}

int main()
{
    int ch;
    while (1)
    {
        RETRY:
        printf("\n+++++| M E N U |+++++\n");
        printf("\n1. Insert new item at the first position");
        printf("\n2. Insert new item at the last position");
        printf("\n3. Insert new item before the given value of node");
        printf("\n4. Insert new item after the given value of node");
        printf("\n5. Search item from the list");
        printf("\n6. Display all elements of the list");
        printf("\n7. Count total number of items");
        printf("\n8. Delete item from the first position");
        printf("\n9. Delete item from the last position");
        printf("\n10. Delete particular item");
        printf("\n11. Delete item before the given item");
        printf("\n12. Delete item after the given item");
        printf("\n13. EXIT");
        printf("\n\nEnter your choice: ");
        scanf("%d",&ch);

        switch (ch)
        {
        case 1:
            insert_F();
            break;
        case 2:
            insert_L();
```

```
                break;
        case 3:
                insert_B();
                break;
        case 4:
                insert_A();
                break;
        case 5:
                search();
                break;
        case 6:
                display();
                break;
        case 7:
                count();
                break;
        case 8:
                delete_F();
                break;
        case 9:
                delete_L();
                break;
        case 10:
                delete_pr();
                break;
        case 11:
                delete_B();
                break;
        case 12:
                delete_A();
                break;
        case 13:
                printf("+++++| EXITING SYSTEM |+++++\n");
                exit(0);
                break;

        default:
        printf("\nINVALID CHOICE !! Please try again.....\n");
        goto RETRY;
                break;
        }
}

return 0;
```

}

OUTPUT