**Assignment No. 2**

**Name: Bhavin Ratansing Patil**

**Roll No.: 26 SEDA**

**Q.1 Write a menu driven c program implementing the operations like Insertion, deletion and search on Single Linked List.**

**Algorithms:**

1. **For Create Function**

struct node \* temp, \*r

temp = malloc(sizeof(struct node))

enter the data

scanf(“%d”,&temp->data)

temp->next=NULL

if(head==NULL)

head=temp

else

r=temp

while(r->next!=NULL)

r=r->next

r->next = temp

1. **For Insertion at first**

struct node \* temp

temp= malloc(sizeof(struct node))

enter the data for first node

scanf(“%d”,&temp->)

temp->next=head

head=temp

1. **For Insertion at Index**

struct node \* temp

temp= malloc(sizeof(struct node))

int index=0;

enter the index and data

scanf(“%d%d”,&index,&temp->data)

struct node \*p = head;

Int i=0

While(i!=index-1)

{p=p->next;

i++;}

temp->next= p->next

p->next=temp

1. **For Insertion at End**

struct node \* temp

temp= malloc(sizeof(struct node))

enter the data for end node

scanf(“%d”,&temp->data)

struct node \* p = head

If(head==NULL)

{

temp->next=NULL;

head=temp;

}

else

{

while(p->next != NULL)

p=p->next

p->next=temp

temp->next=NULL

}

1. **For deletion of first node**

struct Node \*temp

temp= head

head= head->next;

free(temp)

1. **For deletion of Last node**

struct node \*p,\*q

p=head

q=head->next;

while(q->next!=NULL)

{p=p->next; q=q->next}

p->next = NULL

free(q)

1. **For deletion at Index**

struct node \*p,\*q

p=head

q=head->next

int index

enter the index of node

scanf(“%d”,&index)

for(int i=0;i<index;i++)

{p=p->next; q=q->next}

p->next=q->next;

free(q)

1. **Searching the node**

struct node \*temp

int s, index=1

enter the data

scanf(“%d”,&s)

temp=head

while(temp!=NULL)

{

If(temp->data==s)

{print(element found);break;}

Temp=temp->next

Index++

}

If(temp==NULL)

print(not found)

**Applications:**

* Dynamic Memory Allocation
* Memory management
* Implementation of stacks and queues
* Linked allocation of files
* Real world applications like Image viewer and Music player

**Program:**

#include <stdio.h>

#include <stdlib.h>

#include <malloc.h>

void deleteAtFirst();

void deleteAtEnd();

void deleteAtIndex();

struct Node

{

    int data;

    struct Node \*next;

} \* head;

void createList()

{

    struct Node \*temp, \*r;

    int n, index = 1;

    printf("Enter the Number of Elements you want in the List: ");

    scanf("%d", &n);

    for (int i = 0; i < n; i++)

    {

        temp = (struct Node \*)malloc(sizeof(struct Node));

        printf("Enter data at Node %d: ", index);

        index++;

        scanf("%d", &temp->data);

        temp->next = NULL;

        if (head == NULL)

            head = temp;

        else

        {

            r = head;

            while (r->next != NULL)

                r = r->next;

            r->next = temp;

        }

    }

}

void displayList()

{

    struct Node \*show;

    int n = 1;

    if (head == NULL)

        printf("\n\nList is Empty!\n\n");

    else

    {

        show = head;

        while (show != NULL)

        {

            printf("Data of Node %d: %d\n", n, show->data);

            show = show->next;

            n++;

        }

    }

}

void insertAtFirst()

{

    struct Node \*temp;

    temp = (struct Node \*)malloc(sizeof(struct Node));

    printf("\n\nEnter the element which you want at the Beginning : ");

    scanf("%d", &temp->data);

    temp->next = head;

    head = temp;

    printf("==========================================");

    printf("\n\nList after inserting at the Beginning :\n\n");

    displayList();

    printf("==========================================");

}

void insertAtEnd()

{

    struct Node \*temp;

    temp = (struct Node \*)malloc(sizeof(struct Node));

    printf("\n\nEnter the element which you want at the End : \n");

    scanf("%d", &temp->data);

    struct Node \*p = head;

    if (head == NULL)

    {

        temp->next = NULL;

        head = temp;

    }

    else

    {

        while (p->next != NULL)

        {

            p = p->next;

        }

        p->next = temp;

        temp->next = NULL;

    }

    printf("==========================================");

    printf("\n\nList after inserting at the End :\n\n");

    displayList();

    printf("==========================================");

}

void insertAtIndex()

{

    struct Node \*temp;

    int Index = 0;

    temp = (struct Node \*)malloc(sizeof(struct Node));

    printf("\n\nEnter the Index where you want to add element: ");

    scanf("%d", &Index);

    printf("\n\nEnter the element which you want at %d : ", Index);

    scanf("%d", &temp->data);

    struct Node \*p = head;

    int i = 0;

    while (i != Index - 1)

    {

        p = p->next;

        i++;

    }

    temp->next = p->next;

    p->next = temp;

    printf("==========================================");

    printf("\n\nList after inserting at the given Index :\n\n");

    displayList();

    printf("==========================================");

}

void deleteAtFirst()

{

    struct Node \*temp;

    temp = head;

    head = head->next;

    free(temp);

}

void deleteAtEnd()

{

    struct Node \*p, \*q;

    p = head;

    q = head->next;

    while (q->next != NULL)

    {

        p = p->next;

        q = q->next;

    }

    p->next = NULL;

    free(q);

}

void deleteAtIndex()

{

    struct Node \*p, \*q;

    p = head;

    q = head->next;

    int Index;

    printf("Enter the Index of the element which you want to delete from the list: ");

    scanf("%d", &Index);

    for (int i = 0; i < Index - 1; i++)

    {

        p = p->next;

        q = q->next;

    }

    p->next = q->next;

    free(q);

}

void searchInList()

{

    struct Node \*temp;

    int s, index = 1;

    printf("\nEnter the element you want to seach: ");

    scanf("%d", &s);

    temp = head;

    while (temp != NULL)

    {

        if (temp->data == s)

        {

            printf("\nElement Found and the element is at %d in the List\n\n", index);

            break;

        }

        temp = temp->next;

        index++;

    }

    if (temp == NULL)

        printf("\n\nYour Element is not in the List\n\n");

}

int main()

{

    head = NULL;

    int data, c;

    int choice;

    do

    {

        printf("\n\nEnter the choice code for performing operations:\n\n1.Create\t2.Insert\t3.Delete\t4.Search\t5.Exit\n\n");

        scanf("%d", &choice);

        switch (choice)

        {

        case 1:

            createList();

            printf("\n\nData in Nodes before performing operations:\n\n");

            displayList();

            break;

        case 2:

            do

            {

                printf("\n\nSelect where do you want to add the element:\n1.At Beginning\n2.At End\n3.At Index\n4.Back\n\n");

                scanf("%d", &c);

                switch (c)

                {

                case 1:

                    insertAtFirst();

                    break;

                case 2:

                    insertAtEnd();

                    break;

                case 3:

                    insertAtIndex();

                    break;

                default:

                    break;

                }

            } while (c != 4);

            break;

        case 3:

            do

            {

                printf("\n\nSelect from where do you want to delete the element:\n1.At Beginning\n2.At End\n3.At Index\n4.Back\n\n");

                scanf("%d", &c);

                switch (c)

                {

                case 1:

                    deleteAtFirst();

                    printf("==========================================");

                    printf("\n\nList after deleting at the Beginning :\n\n");

                    displayList();

                    printf("==========================================");

                    break;

                case 2:

                    deleteAtEnd();

                    printf("==========================================");

                    printf("\n\nList after deleting at the End :\n\n");

                    displayList();

                    printf("==========================================");

                    break;

                case 3:

                    deleteAtIndex();

                    printf("==========================================");

                    printf("\n\nList after deleting at the given Index :\n\n");

                    printf("==========================================");

                    displayList();

                    break;

                default:

                    break;

                }

            } while (c != 4);

            break;

        case 4:

            searchInList();

            break;

        default:

            break;

        }

    } while (choice != 5);

    return 0;

}

**Output:**



   