SSN COLLEGE OF ENGINEERING (Autonomous)

Affiliated to Anna University

DEPARTMENT OF CSE

UCS308 Data Structures Lab

Assignment 4

Implementation of Doubly Linked List

Register Number : 185001131

Name : Sai Charan B

Class : CSE – B

Create a doubly linked list to store set of student names

Perform the following operations using a menu driven program

1. Insert student name in the front of the list
2. Insert student name at the end of the list
3. Insert a record after a given name in the list
4. Search a given student in the list
5. Delete a given student
6. Display all student names
7. Display the students in alphabetical order

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

typedef struct Node *PtrToNode;
typedef PtrToNode List;
typedef PtrToNode Position;

struct Node
{
	char name[30];
	Position Next;
	Position Prev;
};

int IsEmpty( List L )
{
	return L->Next == NULL;
}

void Find( char X[], List L )
{
	int i=1;
	Position P;
	P = L->Next;
	while( (P->Next) != NULL && strcmp(P->name,X)!=0 )
	{
	P = P->Next;
	i++;
	}
	printf("number in position %d",i);
}
```

```
void Create( char X[], List L, Position P )
{
      Position Temp;
      while(P->Next!=NULL)
            P=P->Next;
      Temp=malloc( sizeof( struct Node ) );
      strcpy(Temp->name, X);
      Temp->Next = NULL;
      Temp->Prev = P;
      P->Next = Temp;
}

void PrintList( const List L )
{
      Position P = L;
      if( IsEmpty( L ) )
    printf( "Empty list\n" );
    else
    {
    do
    {
    P = P->Next;
    printf( "%s ", P->name );
    } while( !(P->Next == NULL) );
    printf( "\n" );
    }
}

void addbeg(List L, char X[])
{
      Position Temp;
      Temp=malloc( sizeof( struct Node ) );
      strcpy(Temp->name, X);
      Temp->Next = L->Next;
      Temp->Prev = L;
      L->Next=Temp;
}

void addafter(List L, char X[], int Location)
{
      Position Temp,P;
      int i;
      P=L;
      for (i=0;i<=Location;i++)
      {
      P=P->Next;
      if(P==NULL)
      {
            printf("There are less than %d Elements", Location);
            return;
      }
      }
      P=P->Prev;
      Temp = malloc( sizeof( struct Node ) );
      strcpy(Temp->name, X);
      Temp->Next = P->Next;
      Temp->Prev = P;
      Temp->Next->Prev=Temp;
      P->Next=Temp;
}

void delete(List L, char X[])
{
      if(L==NULL)
      {
            printf("\n %s is not present in the list", X);
            return;
        }
        Position P,temp;
        P = L->Next;
      while( P!= NULL && (strcmp(P->name,X)!=0) )
        {
        P = P->Next;
        }
        temp = P;
        temp -> Next -> Prev = temp -> Prev;
        temp -> Prev -> Next = temp -> Next;
        free(temp);
}


void PrintAlph(const List L)
{
        Position P = L;
        if( IsEmpty( L ) )
      printf( "Empty list\n" );
      else
      {
        Position A,B;
        A = P->Next;
        char temp[30];
        while(A!=NULL)
        {
              B = A->Next;
              while(B!=NULL)
              {
                    if((strcmp(A->name,B->name)>0))
                    {
                         strcpy(temp,A->name);
                         strcpy(A->name,B->name);
                         strcpy(B->name,temp);
                    }
                    B = B->Next;
              }
              A = A->Next;
        }
      }
      Position t = L;
      do
      {
      t = t->Next;
      printf( "%s ", t->name );
      } while( !(t->Next == NULL) );
      printf( "\n" );


int main()
{
        List L;
        Position P;
        int i;
        L = malloc( sizeof( struct Node ) );
        L->Next = NULL;
        L->Prev=NULL;
        P = L;
        char name[30];
        do
        {
        printf("\n 1. Insert Student name at front");
        printf("\n 2. Insert Student name at end");
        printf("\n 3. Insert After a given record");
        printf("\n 4. Search for student");
        printf("\n 5. Delete a given student");
        printf("\n 6. Display all student details");
        printf("\n 7. Display the students in alphabetical order \n");
        scanf("%d",&i);
        if(i==1)
        {
              printf("\n Enter Student Name to insert at front: ");
              scanf("%s",name);
              addbeg(L,name);

        }
        else if(i==2)
        {
              printf("\n Enter Student Name to insert at rear: ");
              scanf("%s",name);
              Create(name, L,P);
        }
        else if(i==3)
        {
              int loc;
              printf("\n Enter position to add after the record: ");
              scanf("%d",&loc);
              printf("\n Enter Student name to insert: ");
              scanf("%s",name);
              addafter(L,name,loc);
        }
        else if(i==4)
        {
              printf("\n Enter name of student to find position: ");
              scanf("%s",name);
              Find(name, L);
        }
```

```
     else if(i==5)
     {
           printf("\n Enter name of student to delete: ");
           scanf("%s",name);
           delete(L, name);
     }
     else if(i==6)
     {
           PrintList(L);
     }
     else if(i==7)
     {
           PrintAlph(L);
     }
     printf("\nDo you want to continue (1. Yes / 2. No)");
     scanf("%d",&i);
     }while(i!=2);
     return 0;
}
```

Output:

```
  1. Insert Student name at front
 2. Insert Student name at end
 3. Insert After a given record
 4. Search for student
 5. Delete a given student
 6. Display all student details
 7. Display the students in alphabetical order
1
```

```
 Enter Student Name to insert at front: A

Do you want to continue (1. Yes / 2. No)1

 1. Insert Student name at front
 2. Insert Student name at end
 3. Insert After a given record
 4. Search for student
 5. Delete a given student
 6. Display all student details
 7. Display the students in alphabetical order
2

 Enter Student Name to insert at rear: B

Do you want to continue (1. Yes / 2. No)1

 1. Insert Student name at front
 2. Insert Student name at end
 3. Insert After a given record
 4. Search for student
 5. Delete a given student
 6. Display all student details
 7. Display the students in alphabetical order
3

 Enter position to add after the record: 1

 Enter Student name to insert: AB

Do you want to continue (1. Yes / 2. No)1

 1. Insert Student name at front
 2. Insert Student name at end
 3. Insert After a given record
 4. Search for student
 5. Delete a given student
 6. Display all student details
 7. Display the students in alphabetical order
3 4

 Enter name of student to find position: A AB
number in position 2
Do you want to continue (1. Yes / 2. No)1

 1. Insert Student name at front
 2. Insert Student name at end
 3. Insert After a given record
 4. Search for student
 5. Delete a given student
 6. Display all student details
 7. Display the students in alphabetical order
6
A AB B

Do you want to continue (1. Yes / 2. No)1

 1. Insert Student name at front
 2. Insert Student name at end
 3. Insert After a given record
 4. Search for student
 5. Delete a given student
 6. Display all student details
 7. Display the students in alphabetical order
7
A AB B

Do you want to continue (1. Yes / 2. No)2
```