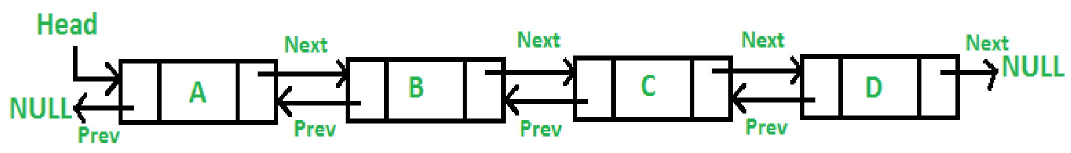
doubly linked list

Write a program to implement the operations of a doubly linked list.

Prerequisite: Basic Knowledge and operations of a Doubly Linked list.

Description: A Doubly Linked List is a type of linked list where each node contains three parts:

1. Data (the actual value)
2. A pointer to the next node
3. A pointer to the previous node



Program:

```
#include<stdio.h>

#include<stdlib.h>

struct node{

   int data;

   struct node *next;

   struct node *prev;

};

struct node *head=NULL;

void create(){

   struct node *nn,*temp;

   nn=(struct node*)malloc(sizeof(struct node));

   printf("enter the value");

scanf("%d",&nn->data);
```

```
    nn->next=NULL;
    nn->prev=NULL;
    if(head==NULL){
        head=nn;
    }
    else{
        temp=head;
        while(temp->next!=NULL){
            temp=temp->next;
        }
        temp->next=nn;
        nn->prev=temp;
    }
}
void insertbeg(){
    struct node *nn,*temp;
    nn=(struct node*)malloc(sizeof(struct node));
    printf("enter the value");
scanf("%d",&nn->data);
    nn->next=NULL;
    nn->prev=NULL;
    if(head==NULL){
        head=nn;
    }
    else{
        head->prev=nn;
        nn->next=head;
        head=nn;
    }
}
void insertspe(){
    struct node *nn,*temp;
    int pos,i;
    nn=(struct node*)malloc(sizeof(struct node));
    printf("enter the value");
scanf("%d",&nn->data);
    nn->next=NULL;
     nn->prev=NULL;
    if(head==NULL){
        head=nn;
    }
    else{
    temp=head;
    printf("enter the position where you want to insert\n");
    scanf("%d",&pos);
        if(pos==0){
       head->prev=nn;
        nn->next=head;
        head=nn;
        return;
    }

    for(i=0;i<pos-1&&temp!=NULL;i++){
        temp=temp->next;
    }
    if(temp==NULL){
        printf("you have entered wrong position\n");
        return;
    }
    else if(temp->next==NULL){
        temp->next=nn;
        nn->prev=temp;
        return;
    }

    temp->next->prev=nn;
    nn->next=temp->next;
    nn->prev=temp;
    temp->next=nn;
}
}

void insertlast(){
       create();
   }
void deletebeg(){
    struct node *temp;
    if(head==NULL){
        printf("there is no node formed\n");
    }
    else{
        temp=head;

        head=temp->next;

        head->prev=NULL;

        free(temp);

    }

}

void deleteend(){

    struct node *temp1,*temp2;

    if(head==NULL){

        printf("there is no node formed\n");

    }

    else{

        temp1=head;

        if(temp1->next==NULL){

            head=NULL;

            free(temp1);


        }


        while(temp1->next!=NULL){

            temp2=temp1;

            temp1=temp1->next;


        }

    temp2->next=temp1->next;

    free(temp1);

        }
```

```
}
void deletespe(){
    struct node *temp1,*temp2;
    int pos,i;
    if(head==NULL){
        printf("there is no node formed\n");
    }
    else{
        temp1=head;
 printf("enter the position where you want to delete\n");
    scanf("%d",&pos);
    if(pos==0){
       head=temp1->next;
        head->prev=temp1->prev;
        free(temp1);
        return;
    }
        for(i=0;i<pos&&temp1!=NULL;i++){
            temp2=temp1;
            temp1=temp1->next;
         }
    if(temp1==NULL){
         printf("you have entered wrong position\n");
         return;
         }
         else if(temp1->next==NULL){
          temp2->next=temp1->next;
          free(temp1);
          return;
         }

         temp1->next->prev=temp2;
         temp2->next=temp1->next;
         free(temp1);
    }
}
void search(){
    struct node *temp;
    int key,found=0;
    printf("enter key");
    scanf("%d",&key);
      if(head==NULL){
        printf("there is no node formed\n");
        }
    else{
        temp=head;

    while(temp!=NULL){
        if(temp->data==key){
            found=1;
        }
        temp=temp->next;
         }
         if(found==1){
            printf("element is found in the list\n");
         }
            else{
                printf("element is not found in the list\n");


            }
         }
}
void display(){
    struct node *temp;
    if(head==NULL){
        printf("list is empty\n");
    }
    else{
       temp=head;
       while(temp!=NULL){
           printf("%d\t",temp->data);
           temp=temp->next;
       }
    }


}
int main(){
    int choice;
    int nodes,i;
    printf("1.create\n2.insert at begining\n3.insert at specific position\n4.insert at end\n5.delete at
begining\n6.delete at specific position\n7.delete at end\n 8.search\n9.display\n10.exit\n");

    while(1){

        printf("enter choice");

        scanf("%d",&choice);

        switch(choice){

            case 1:

            printf("enter no of nodes you want to create\n");

    scanf("%d",&nodes);

                for( i=0;i<nodes;i++){

                        create();

                        }

                break;

            case 2:insertbeg();

                break;

            case 3:insertspe();

                break;

            case 4:insertlast();

                break;

            case 5:deletebeg();

                break;

            case 6:deletespe();

                break;

            case 7:deleteend();

                break;

            case 8:search();

                break;
            case 9:display();
                   break;
            case 10:printf("exiting....\n");
                    display();
                   return -1;
            default:printf("invalid choice\n");
                   break;


        }
    }


    return 0;
}
```