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

struct node
    {
        int data;
        struct node *link;
    } ;
    typedef struct node *npt;
    npt insert_beg(npt root,int x);
    npt delete_beg(npt root);
    npt delete_end(npt root);
    npt delete_mid(npt root,int pos);
    void display(npt root);

int main()
    {
        npt root=NULL;
        int x,ch,pos;
        do
        {
            printf("\n select your choice");
```

```
            printf("\n 1.insert  2.delbeg\n
3.delend\n 4.delmid\n 5.display \n6.exit");
            printf("\n enter your choice");
            scanf("%d",&ch);
            switch(ch)
            {

case 1:
                        printf("\n enter element");
                        scanf("%d",&x);
                        root=insert_beg(root,x);
                        break;
case 2:
                        root=delete_beg(root);
                        break;
case 3:
                        root=delete_end(root);
                        break;
case 4:
                        printf("\n enter pos");
                             scanf("%d",&pos);

root=delete_mid(root,pos);
```

```
                        break;
case 5:
                    display(root);
                    break;
case 6:
                    exit(1);


            }
        }while(ch<=6);
    }

npt insert_beg(npt root,int x)
    {
        npt p;
        p=(npt)malloc(sizeof(struct node));
        p->data=x;
        if(root==NULL)
        {
        p->link=NULL;
        root=p;
        }
        else{
```

```
        p->link=root;
        root=p;
        }

        return root;
    }



void display(npt root)
    {
        npt t;
        if(root==NULL)
    {
            printf("\n linked list is empty");
        }
        t=root;
        while(t!=NULL)
        {
            printf("%4d",t->data);
            t=t->link;
        }
    }
```

```
npt delete_beg(npt root)
{
    npt t;
    if(root==NULL)
        printf("deletion not possible");
    else  if(root->link==NULL)
        root=NULL;
    else
    {
        t=root;
        root=root->link;
        t->link=NULL;
        free(t);
    }
    return root;
}
npt delete_end(npt root)
{
    npt ln,sln;
    if(root==NULL)
        printf("deletion not possible");
    else if(root->link==NULL)
```

```
            root=NULL;
        else
        {
            ln=root;
            while(ln->link!=NULL)
            {
                sln=ln;
                ln=ln->link;
            }
            sln->link=NULL;
            free(ln);
        }
        return root;
}
npt delete_mid(npt root,int pos)
{
        npt t,st;
        int c=1;
        if(root==NULL)
            printf("deletion not possible");
        else if(root->link==NULL)
            root=NULL;
        else if(pos==1)
```

```
        delete_beg(root);
    else
    {
    t=root;
    while(c<pos && t!=NULL)
    {
        printf("in while loop");
        st=t;
        t=t->link;
        printf("%d %d",st->data,t->data);
        c++;
    }
    st->link=t->link;
    t->link=NULL;
    free(t);
    }
    return root;
}
```