**NAME:ADITYA BAHL**

**SEC:B**

**ROLLNO:25**

**-------------------------------------------------------------------------------------------------------------------------------**

**SINGLE LINKED LIST**

#include<stdio.h>

#include<stdlib.h>

typedef struct node

{

int info;

struct node \*next;

}nodetype;

void insert(nodetype \*\*,nodetype\*\*);

void display(nodetype \*);

void delete(nodetype \*\*);

int main()

{

nodetype \*l=NULL,\*r=NULL;

int ch,x;

do

{

printf ("Enter your choice:\n1.FOR INSERTING A NODE IN SINGLE LINKED LIST\n2.FOR DISLPAY THE LINKED LIST\n3.FOR DELETING A NODE\n4.FOR EXIT : ");

scanf("%d",&ch);

switch(ch)

{

case 1:

insert(&l,&r);

break;

case 2:

display(l);

break;

case 3:

delete(&l);

nodetype \*t=l;

while(t->next!=NULL)

t=t->next;

r=t;

break;

case 4:

return 0;

break;

default:

printf("please enter correct choice\n");

break;

}

}while(x!=4);

}

void insert(nodetype \*\*l,nodetype \*\*r)

{

nodetype \*p=NULL;

int x;

p=(nodetype\*)malloc(sizeof(nodetype));

if(p!=NULL)

printf("enter value in node:");

scanf("%d",&x);

p->info=x;

if((\*r)==NULL)

{

(\*l)=(\*r)=p;

}

else

{

(\*r)->next=p;

(\*r)=p;

(\*r)->next=NULL;

}

}

void display (nodetype \*l)

{

if(l!=NULL)

{

while(l!=NULL)

{

printf("%d ",l->info);

l=l->next;

}

}

else

printf("linked list is empty");

printf("\n");

}

void delete(nodetype \*\*l)

{

int a;

printf("enter the key to be searched:");

scanf("%d",&a);

nodetype \*p,\*temp;

p=(\*l);

temp=p->next;

if((\*l)->info==a)

{

(\*l)=(\*l)->next;

free(p);

}

else

{

while(a!=temp->info && temp!=NULL)

{

p=p->next;

temp=temp->next;

}

If (temp! =NULL)

{

p->next=temp->next;

free(temp);

}

else

{

printf("node not found\n");

}

}

}

**CIRCULAR LINKED LIST**

#include <stdio.h>

#include <stdlib.h>

typedef struct node

{

int info;

struct node \*next;

}nodetype;

void cirinsert(nodetype \*\*);

void delete(nodetype \*\*);

void display(nodetype \*);

int main()

{

nodetype \*l=NULL;

int ch;

do

{

printf("Enter your choice: 1.FOR MAKING A CIRCULAR LINKED LIST 2.FOR DISLPAY A NODE 3.FOR DELETING A NODE 4.FOR EXIT: ");

scanf("%d",&ch);

switch(ch)

{

case 1:

cirinsert(&l);

break;

case 2:

display(l);

break;

case 3:

delete(&l);

break;

case 4:

return 0;

break;

default:

printf("please enter correct choice\n");

break;

}

}while(ch!=4);

}

void cirinsert(nodetype \*\*l)

{

nodetype \*p=NULL;

int n;

p=(nodetype\*)malloc(sizeof(nodetype));

if(p!=NULL)

printf("enter the value in node:");

scanf("%d",&n);

p->info=n;

if((\*l)==NULL)

{

(\*l)=p;

(\*l)->next=p;

}

else

{

p->next=(\*l)->next;;

(\*l)->next=p;;

(\*l)=p;

}

}

void display(nodetype \*l)

{

nodetype \*t=l;

do

{

printf("%d ",t->info);

t=t->next;

}while(t!=l);

printf("\n");

}

void delete(nodetype \*\*l)

{

int n;

nodetype \*p,\*temp;

printf("enter the key to be searched:");

scanf("%d",&n);

p=(\*l);

temp=(\*l)->next;

while(temp->next!=(\*l)->next)

{

p=p->next;

temp=temp->next;

}

if((\*l)->info==n)

{

(\*l)=(\*l)->next;

p->next=(\*l);

free(temp);

(\*l)=p;

}

else

{

p=(\*l);

temp=(\*l)->next;

while(temp->info!=n&&temp!=(\*l))

{

p=p->next;

temp=temp->next;

}

if(temp!=(\*l))

{

p->next=temp->next;

free(temp);

}

}

}

**DOUBLE LINKED LIST**

#include <stdio.h>

#include <stdlib.h>

typedef struct node

{

struct node \*next;

int info;

struct node \*prev;

}nodetype;

void insert(nodetype \*\*hd,nodetype \*\*tal)

{

nodetype \*p=NULL;

int n;

p=(nodetype\*)malloc(sizeof(nodetype));

if(p==NULL)

{

printf("not enogh memory\n");

}

else

{

printf("enter the no. in doubly linked list:");

scanf("%d",&n);

p->info=n;

p->next=NULL;

if(\*tal==NULL)

{

\*tal=p;

\*hd=p;

p->prev=NULL;

}

else

{

(\*tal)->next=p;

p->prev=\*tal;

(\*tal)=p;

}

}

}

void forward(nodetype \*hd)

{

if(hd==NULL)

printf("linked list is empty\n");

else

{

while(hd!=NULL)

{

printf("%d ",hd->info);

hd=hd->next;

}

printf("\n");

}

}

void reverse(nodetype \*tal)

{

if(tal==NULL)

printf("linked list is empty\n");

else

{

while(tal!=NULL)

{

printf("%d ",tal->info);

tal=tal->prev;

}

printf("\n");

}

}

void deletenode(nodetype \*\*hd,nodetype \*\*tal)

{

int key;

printf("enter the key to be found and to delete:");

scanf("%d",&key);

nodetype \*t;

t=\*hd;

while(key!=t->info && t->next!=NULL)

{

t=t->next;

}

if(key!=t->info && t->next==NULL)

{

printf("node not found\n");

}

else

{

if(t==(\*hd))

{

\*hd=(\*hd)->next;

(\*hd)->prev=NULL;

}

else if(t==\*tal)

{

\*tal=(\*tal)->prev;

(\*tal)->next=NULL;

}

else

{

(t->prev)->next=t->next;

(t->next)->prev=t->prev;

}

free(t);

}

}

int main()

{

nodetype \*head=NULL,\*tail=NULL;

int ch;

do

{

printf("enter your choice\n1.Insert the element 2.Forward printing 3.Reverse printing 4.Deleting node 5.Exit:");

scanf("%d",&ch);

if(ch==1)

{

insert(&head,&tail);

}

else if(ch==2)

{

forward(head);

}

else if(ch==3)

{

reverse(tail);

}

else if(ch==4)

{

deletenode(&head,&tail);

}

else if(ch==5)

return 0;

else

printf("please enter correct choice\n");

}while(ch!=5);

return 0;

}