SINGLY LINKED LIST

CODE:

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
struct list {
int info;
struct list *next;
};
typedef struct list node;
node *start;

void insert_node_begining()
{
node *item;
item = (node *)malloc(sizeof(node));
printf("Enter the data to insert:");
scanf("%d",&item->info);
if(start == NULL){
item->next=NULL;
start= item;
}else {
item->next = start;
start = item;
}
}

void insert_node_last(){
node *item, *temp;
item = (node *)malloc(sizeof(node));
printf("Enter the data to insert:");
scanf("%d",&item->info);
item->next= NULL;
if(start==NULL)
{
start = item;
}else {
temp=start;
while (temp->next!=NULL) {
temp=temp->next;
}

temp->next=item;
item->next=NULL;

}
}

void insert_node_specific(){
node *item,*temp;
item = (node *)malloc(sizeof(node));
int key;
printf("Enter the number after which you want to put your data:");
scanf("%d",&key);
printf("Enter the actual data to insert:");
scanf("%d",&item->info);
temp= start;
while (temp->next!=NULL && temp->info!=key) {
temp = temp->next;
}
item->next = temp->next;
temp->next = item;


}

void show_data(){
node *temp;
printf("\nElements in the lists are:\n");
temp = start;
while (temp!=NULL) {
printf("%d\t",temp->info);
temp=temp->next;
}
}

int main(){
```

```c
int ch;
do {
system("clear");
printf("\n1.Enter at first \t2.Enter at last \t3.Enter at anywhere \t4. ShowData \n");
printf("Enter your choice :");
scanf("%d",&ch);
switch (ch) {
case 1:
insert_node_begining();
break;
case 2:
insert_node_last();
break;
case 3:
insert_node_specific();
break;
case 4:
show_data();
break;
case 5:
exit(0);
break;
default:
printf("choose from the option:");
}
}while (ch!=5);

return 0;
}
```

DOUBLY LINKED LIST

Code:

```
#include <stdio.h>
#include <stdlib.h>
struct node
{
int info;
struct node *next;
struct node *prev;
};
struct node *head=NULL,*ptr,*newptr;

struct node* getnode()
{
struct node* np;
np=(struct node*)malloc(sizeof(struct node));
printf("Enter the data:");
scanf("%d",&np->info);
np->next =NULL;
np->prev = NULL;
return np;
};

void createList(){
char ch;
do {
newptr = getnode();
if (head == NULL) {
head = newptr;
}else {
ptr->next = newptr;
newptr->prev = ptr;
}
ptr = newptr;
printf("Would you like to add more data:(Y/N)");
scanf(" %c",&ch);
}while (ch == 'y' || ch=='Y');
}
void display(){
ptr = head;
printf("Data in Doubly Link List are:\n");
while (ptr!=NULL) {
printf("%d\t",ptr->info);
ptr=ptr->next;
}

}
void insertAtFront()
{
newptr = getnode();
if (head != NULL)
{
newptr->next = head;
head->prev = newptr;
}
printf("%d is inserted", newptr->info);
head = newptr;
}
void insertAtLast()
{
newptr = getnode();
if (head != NULL)
{
ptr = head;
while (ptr->next != NULL)
{
ptr = ptr->next;
}
ptr->next = newptr;
newptr->prev = ptr;
}
else
{
head = newptr;
}
```

```
printf("%d is inserted", newptr->info);
}
void insertAnywhere()
{
int key;
if (head != NULL)
{
newptr = getnode();
ptr = head;
printf("enter key:");
scanf("%d", &key);
while (ptr != NULL && ptr->info != key)
{
ptr = ptr->next;
}
if (ptr == NULL)
{
printf("\nnode with key does not exist");
}
else
{

newptr->next = ptr->next;
ptr->next = newptr;
ptr->next->prev = newptr;
newptr->prev = ptr;
printf("%d is inserted", newptr->info);
}
}
else
{
insertAtFront();
}
}
void insert()
{
int ch;
if (head != NULL)
{
```

```
do{
system("cls");
printf("1.INSERT at FRONT\n2.INSERT at LAST \n3.INSERT ANYWHERE \n4. PRESS
ENTER TO EXIT\n SELECT FROM OPTION :");
scanf("%d", &ch);
switch (ch)
{
case 1:
insertAtFront();
break;
case 2:
insertAtLast();
break;
case 3:
insertAnywhere();
break;
default:
printf("press enter");
return;
}
} while (ch!=4);
}
else
{
insertAtFront();
}
}

void deleteFromFront()
{
if (head == NULL)
{
printf("linked list is empty!!");
}
else
{
ptr = head;
if (head->next != NULL)
{
```

```
head = head->next;
head->prev = NULL;
}
else
{
head = NULL;
}
printf("%d is deleted", ptr->info);
free(ptr);
}
}
void deleteFromLast()
{
if (head == NULL)
{
printf("linked list is empty!!");
}
else
{
ptr = head;
while (ptr->next != NULL)
{
ptr = ptr->next;
}
printf("%d is deleted", ptr->info);
if (ptr->prev != NULL)
{
ptr->prev->next = NULL;
}
else
{
head = NULL;
}
free(ptr);
}
}
void deleteFromAnywhere()
{
int key;
```

```
if (head == NULL)
{
printf("linked list is empty");
}
else
{
printf("enter key");
scanf("%d", &key);
ptr = head;
while (ptr != NULL && ptr->info != key)
{
ptr = ptr->next;
}
if (ptr == NULL)
{
printf("data with %d does not exist", key);
}
else
{
if (ptr->prev != NULL)
{
if (ptr->next != NULL)
{
ptr->prev->next = ptr->next;
ptr->next->prev = ptr->prev;
}
else
{
ptr->prev->next = NULL;
}
}
else
{
if (ptr->next)
{
ptr->next->prev = NULL;
head = ptr->next;
}
else
{
head = NULL;
}
}
printf("%d is deleted", ptr->info);
free(ptr);
}
}
}
void delete()
{
int ch;
if (head != NULL)
{
while (1) {
system("cls");
printf("1.DELETE at FRONT\n2.DELETE at LAST\n3.DELTE ANYWHERE\nPRESS ENTER
TO EXIT\n Select your option:");
scanf("%d", &ch);
switch (ch)
{
case 1:
deleteFromFront();
break;
case 2:
deleteFromLast();
break;
case 3:
deleteFromAnywhere();
break;
default:
return;
}
}
}
else
{
printf("doubly linked list is empty!!");
}
```

```
}
int main(){
int c;
createList();
do {
printf("\n1.INSERT \t 2. DISPLAY \t 3. DELETE \t 4.EXIT \n Enter your choice:");
scanf("%d",&c);
switch (c) {
case 1:
insert();
break;
case 2:
display();
break;
case 3:
printf("Deletion will be provided soon!");
break;
case 4:
exit(0);
break;
default:
printf("No such option is available.");
}
}while (c!=4);
return 0;
}
```

Circular Linked List

Code :

```
#include <stdio.h>
#include <math.h>
#include <stdlib.h>
struct node
{
int info;
struct node *next;
};
struct node *head=NULL,*ptr,*newptr,*prevptr;
struct node* getnode()
{
struct node* np;
np=(struct node*)malloc(sizeof(struct node));
printf("Enter the data:");
scanf("%d",&np->info);
np->next =NULL;
return np;
};
void create(){
struct node *temp;
char ch;
do {
newptr = getnode();
if(head == NULL){
head = newptr;
}else {
temp->next = newptr;
}
newptr->next = head;
temp = newptr;
printf("Woul you like to add more data:(Y/N)");
scanf(" %c",&ch);
}while (ch == 'y' || ch=='Y');
}
```

```
void display(){
if(head!=NULL){
ptr = head;
printf("Datas in the circular list are:\n");
do {
printf("%d \t",ptr->info);
ptr = ptr->next;
}while (ptr!=head);
}else {
printf("Circular link list is empty !");
}
}

void insertInCList(){
int key;
newptr = getnode();
if (head != NULL) {
printf("Enter the key after which you want to put:");
scanf("%d",&key);

ptr = head;
do{
ptr = ptr->next;
}while (ptr->info !=key && ptr!=head);

if(ptr->info!=key){
printf("Node with key does not exist .");

}else {
printf("%d is inserted !",newptr->info);
newptr->next = ptr->next;
ptr->next = newptr;
}
}
else {
head = newptr;
newptr->next = head;
```

```c
printf("%d is inserted",newptr->info);
}
}

void deleteFromCList(){
ptr = head;
int key;
if(head != NULL)
{
printf("Enter the key to delete:");
scanf("%d",&key);
do{
prevptr = ptr;
ptr=ptr->next;
}while(ptr->info!=key && ptr != head);
if(ptr->info!=key)
{
printf("node with info does not exist.");
}else {
if(ptr==head){
if(ptr->next == head)
{
head=NULL;
}else {
head = ptr->next;
prevptr->next = ptr->next;
}
}else {
prevptr->next = ptr->next;
}
printf("%d is deleted.", ptr->info);
free(ptr);

}
}else {
printf("Link List is empty !");
}
}
int main(){
```

```
create();
int c;
do{
printf("\n1.INSERT \t 2. DISPLAY \t 3. DELETE \t 4. EXIT \n Enter from options: ");
scanf("%d",&c);
switch (c) {
case 1:
insertInCList();
break;
case 2:
display();
break;
case 3:
deleteFromCList();
break;
case 4:
exit(0);
default:
printf("No such option available !");
}
}while(1);
return 0;
}
```