**Assignment no. 5**

**Subject: Data Structure and Algorithm**

**Name: Shrirang Mhalgi**

**Roll No.:222006**

**Class-:S.E.**

**Division-: B**

**Batch-:B1**

# Title/Problem Statement-:

Write C++ program to store set of negative and positive numbers using linked list. Write functions to

A) Insert numbers

B) Delete nodes with negative numbers

c) Create two more linked lists using this list, one containing all positive numbers and other containing negative numbers

d) For two lists that are sorted; merge these two lists into third resultant list that is sorted

**CODE**

#include<iostream>

using namespace std;

struct node

{

int num;

struct node \*next;

}

\*start=NULL,\*last=NULL,\*ptr,\*temp,\*newptr,\*start1=NULL,

\*last1=NULL,\*start2=NULL,\*last2=NULL,\*head1=NULL,

\*end1=NULL,\*head2=NULL,\*end2=NULL,\*head=NULL,\*end=NULL;

void display(node \*np)

{

if(np==NULL)

cout<<"\nLIST IS EMPTY!!!\n";

else

{

cout<<"\nLIST CONTAINS :- \n";

while(np!=NULL)

{

cout<<np->num<<" --> ";

np=np->next;

}

cout<<"NULL\n";

}

}

void insert()

{

static int i=1;

cout<<"\nENTER THE NUMBERS :- \n";

ptr=new node;

ptr->next=NULL;

cout<<"\nNO "<<i++<<" : ";

cin>>ptr->num;

if(start==NULL)

{

start=last=ptr;

}

else

{

last->next=ptr;

last=ptr;

}

display(start);

}

void del()

{

temp=ptr=start;

while(temp!=NULL)

{

if(temp->num>=0)

{

ptr=temp;

last=temp;

temp=temp->next;

}

else

{

if(temp==start)

start=start->next;

else

ptr->next=temp->next;

delete temp;

temp=ptr=start;

}

}

}

void separate()

{

ptr=start;

while(ptr!=NULL)

{

if(ptr->num<0)

{

if(start1==NULL)

last1=start1=ptr;

else

{

last1->next=ptr;

last1=ptr;

}

}

else

{

if(start2==NULL)

last2=start2=ptr;

else

{

last2->next=ptr;

last2=ptr;

}

}

ptr=ptr->next;

}

last1->next=last2->next=NULL;

cout<<"\nLIST OF -VE NUMBERS :-\n";

display(start1);

cout<<"\nLIST OF +VE NUMBERS :-\n";

display(start2);

start=start1=start2=NULL;

}

int check(node\*np)

{

int count=0;

while(np!=NULL)

{

if(np->num<0)

{

count++;

}

np=np->next;

}

return count;

}

void ins1()

{

static int i=1;

ptr=new node;

ptr->next=NULL;

cout<<"\nNO "<<i++<<" : ";

cin>>ptr->num;

if(head1==NULL)

head1=end1=ptr;

else

{

end1->next=ptr;

end1=ptr;

}

display(head1);

}

void ins2()

{

static int i=1;

ptr=new node;

ptr->next=NULL;

cout<<"\nNO "<<i++<<" : ";

cin>>ptr->num;

if(head2==NULL)

head2=end2=ptr;

else

{

end2->next=ptr;

end2=ptr;

}

display(head2);

}

void merge()

{

node \*end;

ptr=head1;

temp=head2;

while(ptr!=NULL&&temp!=NULL)

{

newptr=new node;

newptr->next=NULL;

if(ptr->num<temp->num)

{

newptr->num=ptr->num;

if(head==NULL)

end=head=newptr;

else

{

end->next=newptr;

end=newptr;

}

ptr=ptr->next;

}

else if(ptr->num==temp->num)

{

newptr->num=ptr->num;

if(head==NULL)

end=head=newptr;

else

{

end->next=newptr;

end=newptr;

}

ptr=ptr->next;

temp=temp->next;

}

else

{

newptr->num=temp->num;

if(head==NULL)

end=head=newptr;

else

{

end->next=newptr;

end=newptr;

}

temp=temp->next;

}

}

if(temp!=NULL)

{

while(temp!=NULL)

{

newptr=new node;

newptr->next=NULL;

newptr->num=temp->num;

if(head==NULL)

end=head=newptr;

else

{

end->next=newptr;

end=newptr;

}

temp=temp->next;

}

}

else

{

while(ptr!=NULL)

{

newptr=new node;

newptr->next=NULL;

newptr->num=ptr->num;

if(head==NULL)

end=head=newptr;

else

{

end->next=newptr;

end=newptr;

}

ptr=ptr->next;

}

}

end1=head1=head2=end2=NULL;

}

int main()

{

node \*end;

int choice,count=0;

char ch;

do

{

cout<<"\n1. INSERT NUMBERS IN THE LIST\n";

cout<<"\n2. DELETE -VE NUMBERS \n";

cout<<"\n3. SEPARATING +VE AND -VE NUMBERS \n";

cout<<"\n4. MERGING TWO SORTED LISTS \n";

cout<<"\nCHOICE : ";

cin>>choice;

switch(choice)

{

case 1: cout<<"\nINSERT NUMBERS IN THE LIST:-\n";

do

{

insert();

cout<<"\nDO YOU WANT TO ENTER MORE NUMBERS (y/n) : ";

cin>>ch;

}while(ch=='y'||ch=='Y');

break;

case 2: if(start==NULL)

cout<<"\nEMPTY LIST!!!CANNOT DELETE NUMBERS!!!\n";

else

{

cout<<"\nLIST BEFORE DELETING :- \n";

display(start);

cout<<"\nLIST AFTER DELETING ALL -VE NUMBERS :-\n";

del();

display(start);

}

break;

case 3: count=check(start);

if(start==NULL)

cout<<"\nEMPTY LIST!!!CANNOT SEAPRATE LISTS!!!\n";

else if(count==0)

{

cout<<"\nNO NEGATIVE NUMBERS ARE PRESENT!!!\n";

display(start);

}

else

{

separate();

}

break;

case 4: cout<<"\nENTER VALUES :- \n";

cout<<"\n1st LIST : ";

do

{

ins1();

cout<<"WANT TO INSERT MORE NUMBERS? (y/n) : ";

cin>>ch;

}while(ch=='y'||ch=='Y');

cout<<"\n2nd LIST : ";

do

{

ins2();

cout<<"WANT TO INSERT MORE NUMBERS? (y/n) : ";

cin>>ch;

}while(ch=='y'||ch=='Y');

merge();

cout<<"\nTHE MERGED LIST :-\n";

display(head);

head=end=NULL;

break;

default: cout<<"\nWRONG CHOICE!!!\n";

}

cout<<"\nWANT TO CONTINUE (y/n) : ";

cin>>ch;

}while(ch=='y'||ch=='Y');

return 0;

}

**OUTPUT**

1. INSERT NUMBERS IN THE LIST

2. DELETE -VE NUMBERS

3. SEPARATING +VE AND -VE NUMBERS

4. MERGING TWO SORTED LISTS

CHOICE : 1

INSERT NUMBERS IN THE LIST:-

ENTER THE NUMBERS :-

NO 1 : 3

LIST CONTAINS :-

3 --> NULL

DO YOU WANT TO ENTER MORE NUMBERS (y/n) : y

ENTER THE NUMBERS :-

NO 2 : -2

LIST CONTAINS :-

3 --> -2 --> NULL

DO YOU WANT TO ENTER MORE NUMBERS (y/n) : y

ENTER THE NUMBERS :-

NO 3 : 7

LIST CONTAINS :-

3 --> -2 --> 7 --> NULL

DO YOU WANT TO ENTER MORE NUMBERS (y/n) : y

ENTER THE NUMBERS :-

NO 4 : -8

LIST CONTAINS :-

3 --> -2 --> 7 --> -8 --> NULL

DO YOU WANT TO ENTER MORE NUMBERS (y/n) : y

ENTER THE NUMBERS :-

NO 5 : 1

LIST CONTAINS :-

3 --> -2 --> 7 --> -8 --> 1 --> NULL

DO YOU WANT TO ENTER MORE NUMBERS (y/n) : n

WANT TO CONTINUE (y/n) : y

1. INSERT NUMBERS IN THE LIST

2. DELETE -VE NUMBERS

3. SEPARATING +VE AND -VE NUMBERS

4. MERGING TWO SORTED LISTS

CHOICE : 2

LIST BEFORE DELETING :-

LIST CONTAINS :-

3 --> -2 --> 7 --> -8 --> 1 --> NULL

LIST AFTER DELETING ALL -VE NUMBERS :-

LIST CONTAINS :-

3 --> 7 --> 1 --> NULL

WANT TO CONTINUE (y/n) : y

1. INSERT NUMBERS IN THE LIST

2. DELETE -VE NUMBERS

3. SEPARATING +VE AND -VE NUMBERS

4. MERGING TWO SORTED LISTS

CHOICE : 3

NO NEGATIVE NUMBERS ARE PRESENT!!!

LIST CONTAINS :-

3 --> 7 --> 1 --> NULL

WANT TO CONTINUE (y/n) : y

1. INSERT NUMBERS IN THE LIST

2. DELETE -VE NUMBERS

3. SEPARATING +VE AND -VE NUMBERS

4. MERGING TWO SORTED LISTS

CHOICE : 4

ENTER VALUES :-

1st LIST :

NO 1 : 1

LIST CONTAINS :-

1 --> NULL

WANT TO INSERT MORE NUMBERS? (y/n) : y

NO 2 : 2

LIST CONTAINS :-

1 --> 2 --> NULL

WANT TO INSERT MORE NUMBERS? (y/n) : y

NO 3 : 3

LIST CONTAINS :-

1 --> 2 --> 3 --> NULL

WANT TO INSERT MORE NUMBERS? (y/n) : n

2nd LIST :

NO 1 : 5

LIST CONTAINS :-

5 --> NULL

WANT TO INSERT MORE NUMBERS? (y/n) : y

NO 2 : 5

LIST CONTAINS :-

5 --> 5 --> NULL

WANT TO INSERT MORE NUMBERS? (y/n) : n

THE MERGED LIST :-

LIST CONTAINS :-

1 --> 2 --> 3 --> 5 --> 5 --> NULL

WANT TO CONTINUE (y/n) : y

1. INSERT NUMBERS IN THE LIST

2. DELETE -VE NUMBERS

3. SEPARATING +VE AND -VE NUMBERS

4. MERGING TWO SORTED LISTS