**Debre Markos University**

**Department of Information Technology**

**Data Structure and Algorithm Lab Manual**

**Lab 5: Implementation of doubly linked list**

**Tools used:** Quincy 2005v.1.3 editor

**Objective:**

Students should be able to know:

- What doubly linked lists are;

- How operations like insertion and deletion are implemented on doubly linked lists

#include<iostream.h>

struct Dlink

{

int data;

Dlink \*next;

Dlink \*prev;

};

Dlink \*current = NULL,\*t;

void creatDlink(int d)

{

Dlink \*temp;

temp = new Dlink;

temp-> data = d;

temp -> next = NULL;

temp->prev=NULL;

if(current = = NULL)

{

current = temp;

t=current;

}

else

{

t->next = temp;

temp->prev = t;

t=temp;

} }

void DDeleteFront()

{

Dlink \*temp;

if(current = = NULL)

cout<<"Empty list.\n";

else

{

temp = current;

current = current->next;

current->prev = NULL;

delete temp;

} }

void DDeleteEnd()

{

Dlink \*temp1,\*temp2;

if(current = = NULL)

cout<<"Empty list.\n";

else

{

temp1 = current;

while(temp1->next != NULL)

{

temp2 = temp1;

temp1 = temp1->next;

}

temp2->next = NULL;

delete temp1;

} }

void DinsertEnd(int item)

{

Dlink \*temp1,\*temp2;

temp1=current;

temp2=new Dlink;

temp2->data=item;

temp2->next=NULL;

while(temp1->next!=NULL)

{temp1=temp1->next;}

temp1->next=temp2;

temp2->prev=temp1;

}

void DinsertFront(int item)

{

Dlink \*temp;

temp=new Dlink;

temp->data=item;

temp->prev=NULL;

temp->next=current;

current->prev=temp;

current=temp;

}

void Ddisplay()

{

Dlink \*t;

if(current = = NULL)

cout<<"Empty list.\n";

else

{

t = current;

while(t != NULL)

{

cout<<t-> data<<" ";

t = t->next;

}}

cout<<endl;

}

int main()

{

int m, value;

cout<<"How many elements in the doubly linked list? ";

cin>>m;

for(int i=1;i<=m ; i ++)

{

cout<<"\tEnter element "<<i<<" :";

cin>>value;

creatDlink(value);

}

cout<<"\nThe doubly linked list elements are: \n\t";

Ddisplay();

cout<<"\nWhen FRONT element is DELETED, it becomes:\n\t";

DDeleteFront();

Ddisplay();

cout<<"\nEnter an element to INSERT at the END: ";

cin>>value;

cout<<"\nAfter inserting "<<value<<" at the end, it becomes: \n\t";

DinsertEnd(value);

Ddisplay();

cout<<"\nEnter an element to INSERT at FRONT: ";

cin>>value;

cout<<"\nAfter inserting "<<value<<" at front, it becomes: \n\t";

DinsertFront(value);

Ddisplay();

cout<<"\nAfter the END element is DELETED, it becomes: \n\t";

DDeleteEnd();

Ddisplay();

}