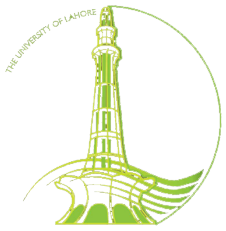

THE UNIVERSITY OF LAHORE
ISLAMABAD CAMPUS

Data Structure And Algorithum

Lab Report

Name:	Ch Hasnain Zafar
Registration #:	SEU-F16-104
Lab Report #:	06
Dated:	19-03-2018
Submitted To:	Mr. Usman Ahmed

The University of Lahore, Islamabad Campus
Department of Computer Science & Information Technology

Experiment # 1
Double Linked List

Objective
To understand and implement the Double Link List Problem.

Software Tool
1. DEV C++

1 Theory

Doubly linked list implementation. A doubly-linked list is a linked data structure that consists of a set of sequentially linked records called nodes. Each node contains two fields, called links, that are references to the previous and to the next node in the sequence of nodes.
A node can be added in four ways
1) At the front of the DLL
2) After a given node.
3) At the end of the DLL
4) Before a given node.

2 Task

2.1 Procedure: Task 1

```
#include<iostream>
#include<cstdio>
#include<cstdlib>

using namespace std;
struct node
{
```

```
    int info;
    struct node *next;
    struct node *prev;
}*start;


class double_llist
{
    public:
        void create_list(int value);
        void add_begin(int value);
        void add_after(int value, int position);
        void delete_element(int value);
        void search_element(int value);
        void display_dlist();
        void count();
        void reverse();
        double_llist()
        {
            start = NULL;
        }
};

int main()
{
    int choice, element, position;
    double_llist dl;
    while (1)
    {
        cout<<endl<<"----------------------------------"<<endl;
        cout<<endl<<"Operations on Doubly linked list"<<endl;
        cout<<endl<<"----------------------------------"<<endl;
        cout<<"1.Create Node"<<endl;
        cout<<"2.Add at begining"<<endl;
        cout<<"3.Add after position"<<endl;
        cout<<"4.Delete"<<endl;
        cout<<"5.Display"<<endl;
        cout<<"6.Count"<<endl;
        cout<<"7.Reverse"<<endl;
        cout<<"8.Quit"<<endl;
```

```
cout<<"Enter your choice : ";
cin>>choice;
switch ( choice )
{
case 1:
    cout<<"Enter the element: ";
    cin>>element;
    dl.create_list(element);
    cout<<endl;
    break;
case 2:
    cout<<"Enter the element: ";
    cin>>element;
    dl.add_begin(element);
    cout<<endl;
    break;
case 3:
    cout<<"Enter the element: ";
    cin>>element;
    cout<<"Insert Element after postion: ";
    cin>>position;
    dl.add_after(element, position);
    cout<<endl;
    break;
case 4:
    if (start == NULL)
    {
        cout<<"List empty, nothing to delete"<<endl;
        break;
    }
    cout<<"Enter the element for deletion: ";
    cin>>element;
    dl.delete_element(element);
    cout<<endl;
    break;
case 5:
    dl.display_dlist();
    cout<<endl;
    break;
case 6:
```

```
            dl.count();
            break;
        case 7:
            if (start == NULL)
            {
                cout<<"List empty, nothing to reverse"<<endl;
                break;
            }
            dl.reverse();
            cout<<endl;
            break;
        case 8:
            exit(1);
        default:
            cout<<"Wrong choice"<<endl;
        }
    }
    return 0;
}

void double_llist::create_list(int value)
{
    struct node *s, *temp;
    temp = new(struct node);
    temp->info = value;
    temp->next = NULL;
    if (start == NULL)
    {
        temp->prev = NULL;
        start = temp;
    }
    else
    {
        s = start;
        while (s->next != NULL)
            s = s->next;
        s->next = temp;
        temp->prev = s;
    }
}
```

```
void double_llist::add_begin(int value)
{
    if (start == NULL)
    {
        cout<<"First Create the list."<<endl;
        return;
    }
    struct node *temp;
    temp = new(struct node);
    temp->prev = NULL;
    temp->info = value;
    temp->next = start;
    start->prev = temp;
    start = temp;
    cout<<"Element Inserted"<<endl;
}

void double_llist::add_after(int value, int pos)
{
    if (start == NULL)
    {
        cout<<"First Create the list."<<endl;
        return;
    }
    struct node *tmp, *q;
    int i;
    q = start;
    for (i = 0;i < pos - 1;i++)
    {
        q = q->next;
        if (q == NULL)
        {
            cout<<"There are less than ";
            cout<<pos<<" elements."<<endl;
            return;
        }
    }
    tmp = new(struct node);
    tmp->info = value;
```

```
    if (q->next == NULL)
    {
        q->next = tmp;
        tmp->next = NULL;
        tmp->prev = q;
    }
    else
    {
        tmp->next = q->next;
        tmp->next->prev = tmp;
        q->next = tmp;
        tmp->prev = q;
    }
    cout<<"Element Inserted"<<endl;
}

void double_llist::delete_element(int value)
{
    struct node *tmp, *q;
    if (start->info == value)
    {
        tmp = start;
        start = start->next;
        start->prev = NULL;
        cout<<"Element Deleted"<<endl;
        free(tmp);
        return;
    }
    q = start;
    while (q->next->next != NULL)
    {
        if (q->next->info == value)
        {
            tmp = q->next;
            q->next = tmp->next;
            tmp->next->prev = q;
            cout<<"Element Deleted"<<endl;
            free(tmp);
            return;
        }
```

```
            q = q->next;
        }
        if (q->next->info == value)
        {
            tmp = q->next;
            free(tmp);
            q->next = NULL;
            cout<<"Element Deleted"<<endl;
            return;
        }
        cout<<"Element "<<value<<" not found"<<endl;
}

void double_llist::display_dlist()
{
        struct node *q;
        if (start == NULL)
        {
            cout<<"List empty,nothing to display"<<endl;
            return;
        }
        q = start;
        cout<<"The Doubly Link List is :"<<endl;
        while (q != NULL)
        {
            cout<<q->info<<" <-> ";
            q = q->next;
        }
        cout<<"NULL"<<endl;
}

void double_llist::count()
{
        struct node *q = start;
        int cnt = 0;
        while (q != NULL)
        {
            q = q->next;
            cnt++;
        }
```

```
    cout<<"Number of elements are: "<<cnt<<endl;
}

void double_llist::reverse()
{
    struct node *p1, *p2;
    p1 = start;
    p2 = p1->next;
    p1->next = NULL;
    p1->prev = p2;
    while (p2 != NULL)
    {
        p2->prev = p2->next;
        p2->next = p1;
        p1 = p2;
        p2 = p2->prev;
    }
    start = p1;
    cout<<"List Reversed"<<endl;
}
```