UCS301 Assignment 2

Name: Ananya Agarwal

Roll No.:102083036

Batch: 2CO14

Q1. Develop a menu driven program for the following operations of on a Singly Linked List.

(a) Insertion at the beginning.

(b) Insertion at the end.

(c) Insertion in between (before or after a node having a specific value, say 'Insert a new Node 35 before/after the Node 30').

(d) Deletion from the beginning.

(e) Deletion from the end.

(f) Deletion of a specific node, say 'Delete Node 60').

(g) Search for a node and display its position from head.

(h) Display all the node values.

Ans1.

```
#include<iostream>
using namespace std;
struct node
{
	int data;
	node *next;
};
class l_list
{
	node *head;
	public:
		l_list()
		{
			head=NULL;
```

```
                }
                void i_beg();
                void i_end();
                void i_before();
                void i_after();
                void d_beg();
                void d_end();
                void d_specific();
                void search();
                void display();
};
void l_list::i_beg()
{
        node *temp=new node;
        cout<<"Enter the value to be inserted";
        cin>>temp->data;
        temp->next=NULL;
        if(head == NULL)
        head=temp;
        else
        {
                node *temp1=head;
                while(temp1->next!=NULL)
                temp1=temp1->next;
                temp1->next=temp;
        }
}
void l_list::i_end()
{
node *temp=new node;
cout<<"Enter the value to insert : ";
cin>>temp->data;
```

```
temp->next=NULL;
if(head==NULL)
head=temp;
else
{
node *temp1=head;
while(temp1->next!=NULL)
temp1=temp1->next;
temp1->next=temp;
}
}
void l_list::i_before()
{
        if(head=NULL)
        {
                cout<<"list is empty!!!!\n";
        }
        else
        {
                int x;
                node *temp=new node;
                cout<<"Enter the value to be inserted:";
                cin>>temp->data;
                temp->next=NULL;
                cout<<"Enter the value before which the element should be inserted";
                cin>>x;
                node *temp1=head,*temp2=NULL;
                while(temp1!=NULL && temp1->data!=x)
                {
                        temp2=temp1;
                        temp1=temp1->next;
                }
```

```
                if(temp1==NULL)
                {
                        cout<<"not found in the list!! \n";
                        delete temp;
                }
                else if(temp1 ==head)
                {
                        temp->next=head;
                        head=temp;
                }
                else
                {
                        temp->next=temp1;
                        temp2->next=temp;
                }
        }
}
void l_list::i_after()
{
        if(head == NULL)
        {
                cout<<"List is empty !!!!!\n";
        }
        else
        {
                int x;
                node *temp=new node;
                cout<<"Enter the value to be inserted";
                cin>>temp->data;
                temp->next=NULL;
                cout<<"Enter the value before which the element should be inserted";
                cin>>x;
```

```
                node *temp1=head;
                while(temp1!=NULL && temp1->data!=x)
                temp1=temp1->next;
                if(temp1==NULL)
                {
                        cout<<"Element not found!!!\n ";
                        delete temp;
                }
                else
                {
                        temp->next=temp1->next;
                        temp1->next=temp;
                }
        }
}
void l_list::d_beg()
{
        if(head==NULL)
        cout<<"List is empty!!\n";
        else
        {
                node *temp=head;
                head=head->next;
                delete temp;
        }
}
void l_list::d_end()
{
        if(head==NULL)
        cout<<"List is empty!!\n";
        else
{
```

```
node *temp=head,*temp1=NULL;
while(temp->next!=NULL)
{
        temp1=temp;
        temp=temp->next;
        }
        if(temp==head)
        head=NULL;
        else
        temp1->next=NULL;
        delete temp;
}
}
void l_list::d_specific()
{
        if(head==NULL)
        cout<<"List is empty!!!\n";
        else
        {
                int x;
                cout<<"Enter the value of element to be deleted";
                cin>>x;
                node *temp=head,*temp1=NULL;
                while(temp!=NULL && temp->data!=x)
                {
                        temp1=temp;
                        temp=temp->next;
                }
                if(temp==NULL)
                cout<<"Element not found!!\n";
                else if(temp==head)
                {
```

```
                head=head->next;
                delete temp;
        }
        else
        {
                temp1->next=temp->next;
                delete temp;
        }
    }
}
void l_list::search()
{
    if(head==NULL)
    cout<<"List is empty!!\n";
    else
    {
        int x,count=0;
        node *temp=head;
        cout<<"Enter value of element to be searched";
        cin>>x;
        while(temp!=NULL && temp->data!=x)
        {
                temp=temp->next;
                count++;
        }
        if(temp==NULL)
        cout<<"Element not found!!\n";
        else
        {
                cout<<temp->data<<"found at position :"<<count<<endl;
        }
    }
```

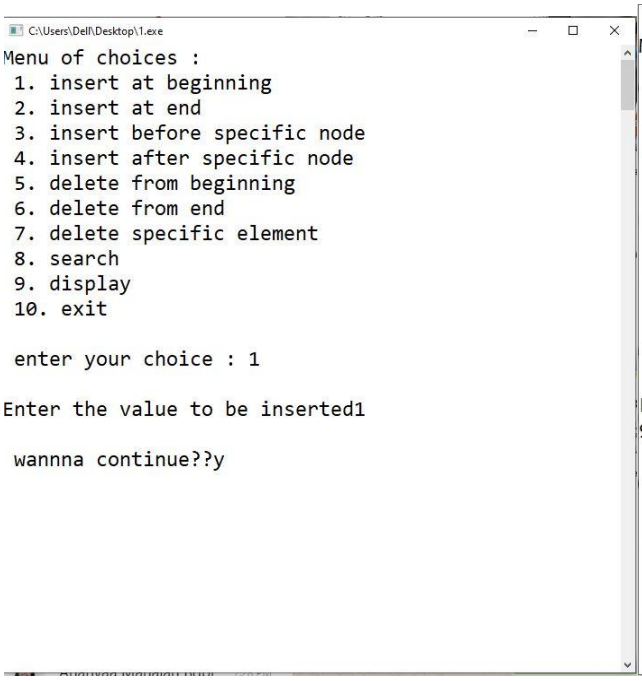
```
}
void l_list::display()
{
	if(head==NULL)
	cout<<"List is empty!!\n";
	else
	{
		cout<<"Linked list elements : \n";
		node *temp=head;
		while (temp!=NULL)
		{
			cout<<temp->data<<' ';
			temp=temp->next;
		}
		cout<<endl;
	}
}
int main()
{
	l_list a;
	char c;
	int ch;
	do
	{
		cout<<"Menu of choices :\n";
		cout<<" 1. insert at beginning \n";
		cout<<" 2. insert at end \n";
		cout<<" 3. insert before specific node \n";
		cout<<" 4. insert after specific node \n";
		cout<<" 5. delete from beginning \n";
		cout<<" 6. delete from end \n";
		cout<<" 7. delete specific element \n";
```

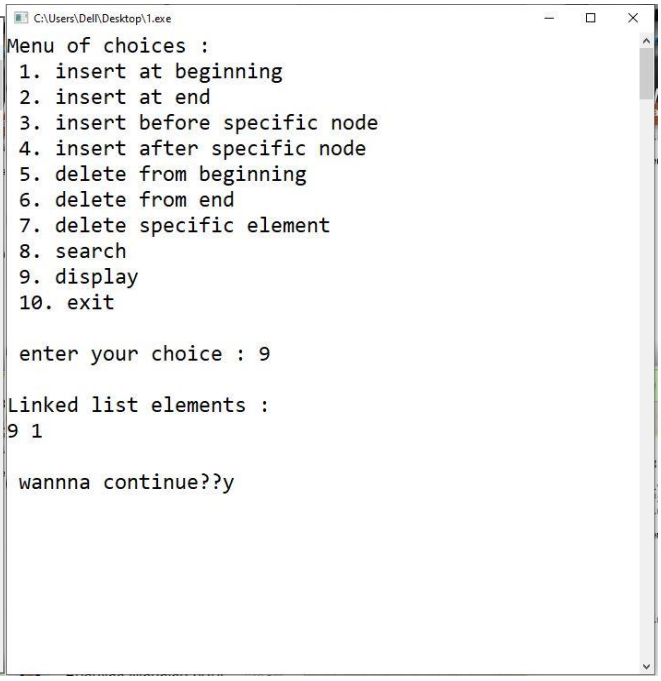
```
cout<<" 8. search \n";
cout<<" 9. display \n";
cout<<" 10. exit \n";
cout<<" \n enter your choice : ";
cin>>ch;
cout<<endl;
switch(ch)
{
        case 1:
                a.i_beg();
                break;
        case 2:
                a.i_end();
                break;
        case 3:
                a.i_before();
                break;
        case 4:
                a.i_after();
                break;
        case 5:
                a.d_beg();
                break;
        case 6:
                a.d_end();
                break;
        case 7:
                a.d_specific();
                break;
        case 8:
                a.search();
                break;
```

```
                case 9:
                        a.display();
                        break;
                case 10:
                        exit(0);
                default:
                cout<<"Invalid choice!!\n";
        }
        cout<<"\n wannna continue??";
        cin.ignore();
        cin>>c;
        system("cls");
}
while(c=='y'|| c=='Y');
}
```

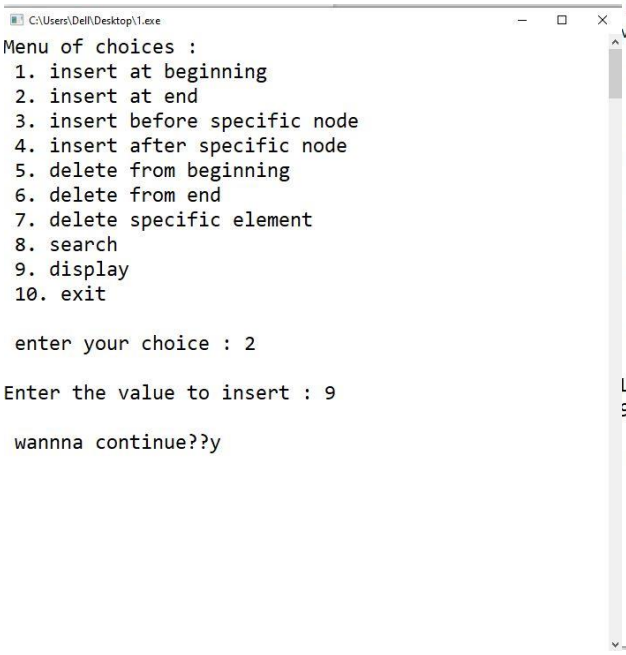
Outputs:

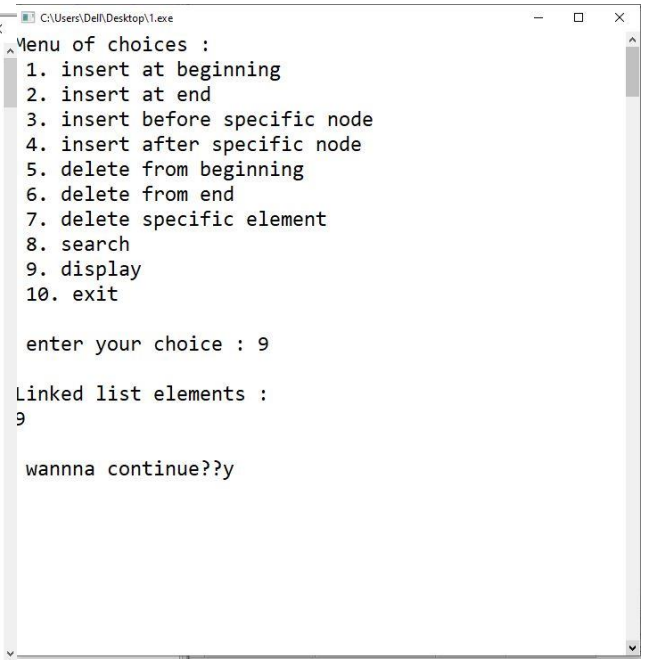
1. When Choice is 1



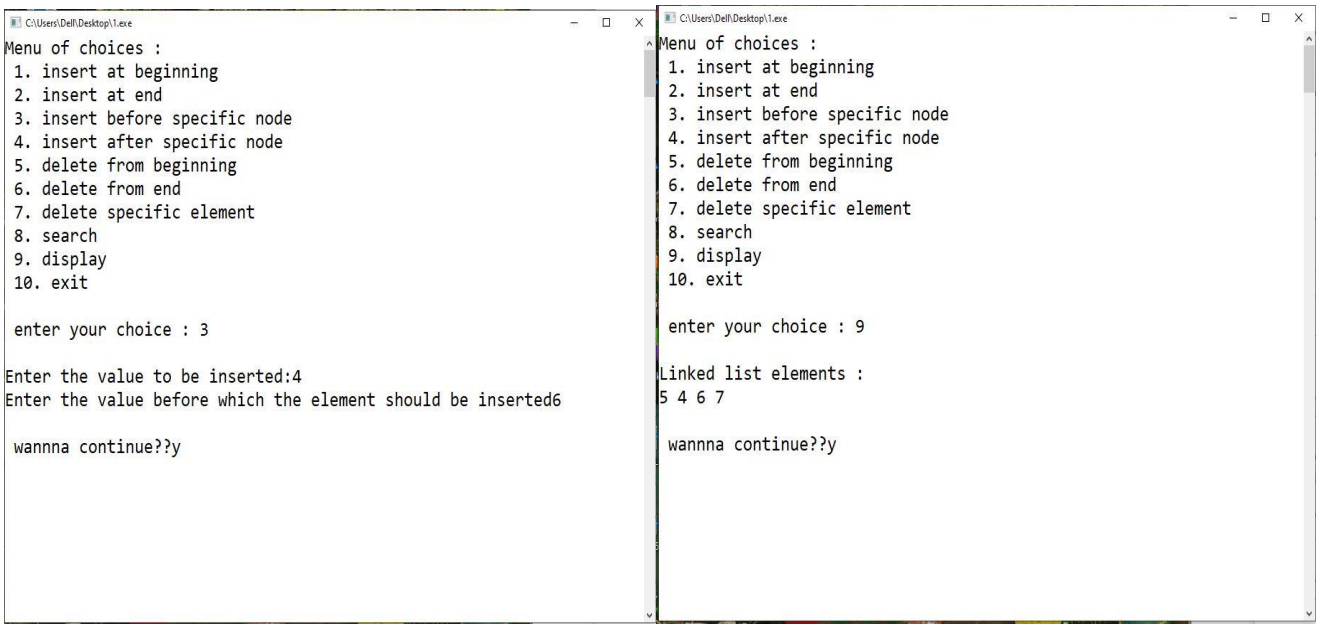


2. When Choice is 2

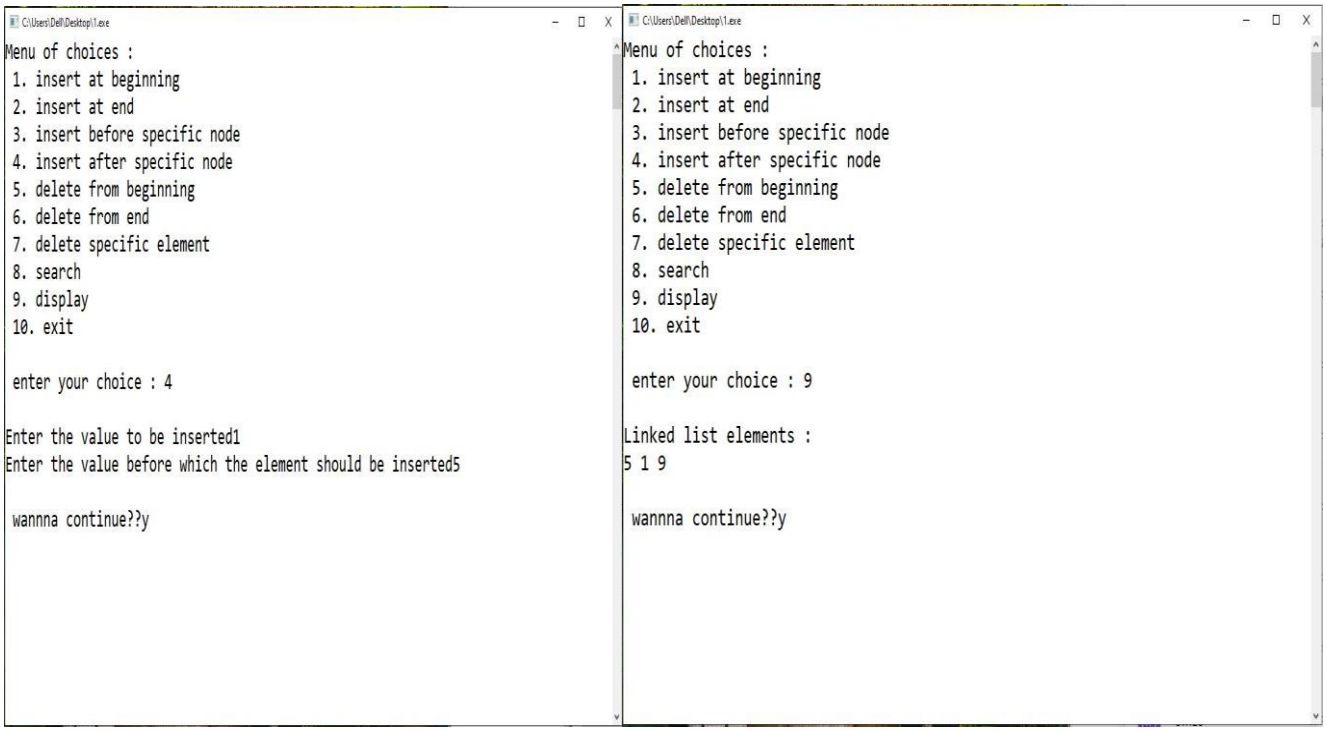




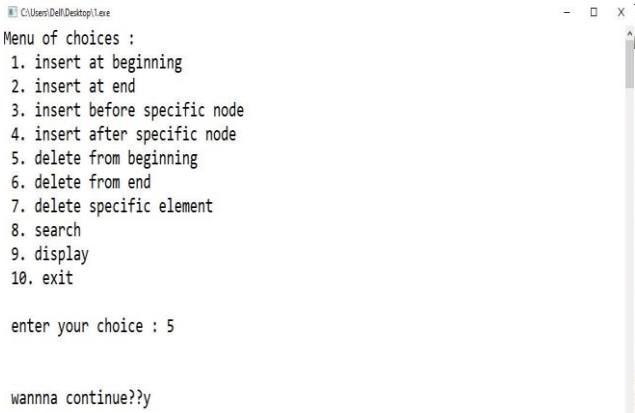
3. When Choice is 3

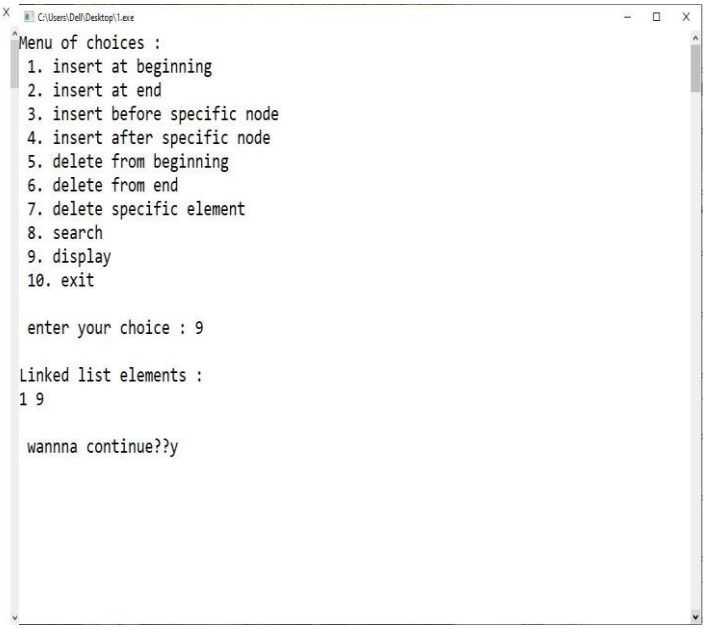


4. When Choice is 4

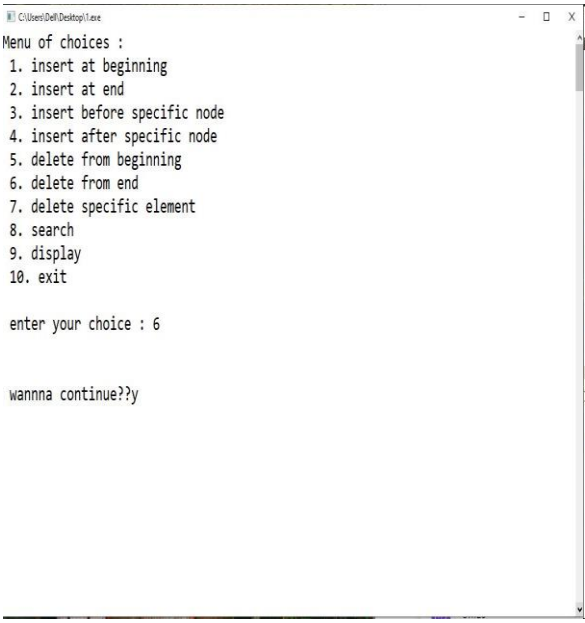


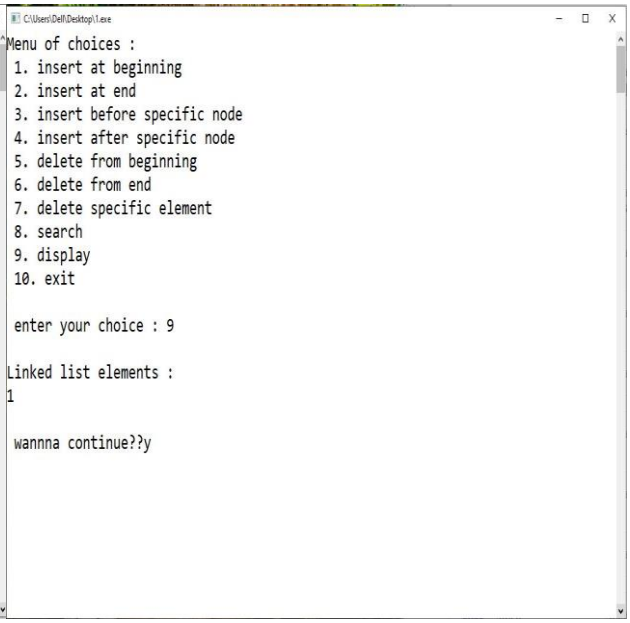
5. When Choice is 5



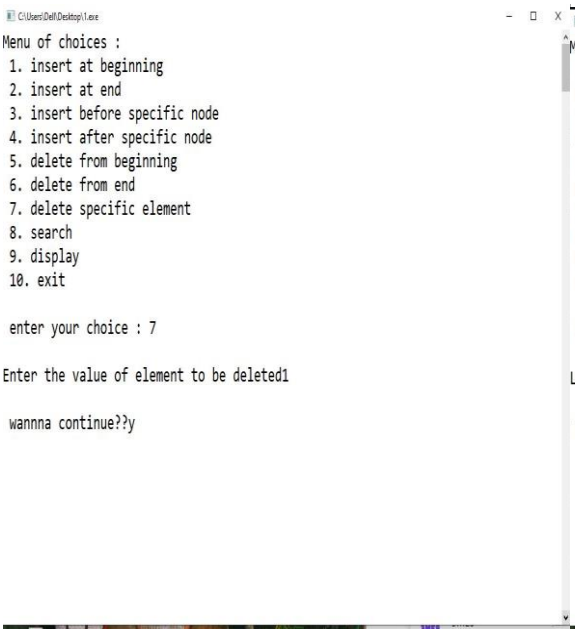


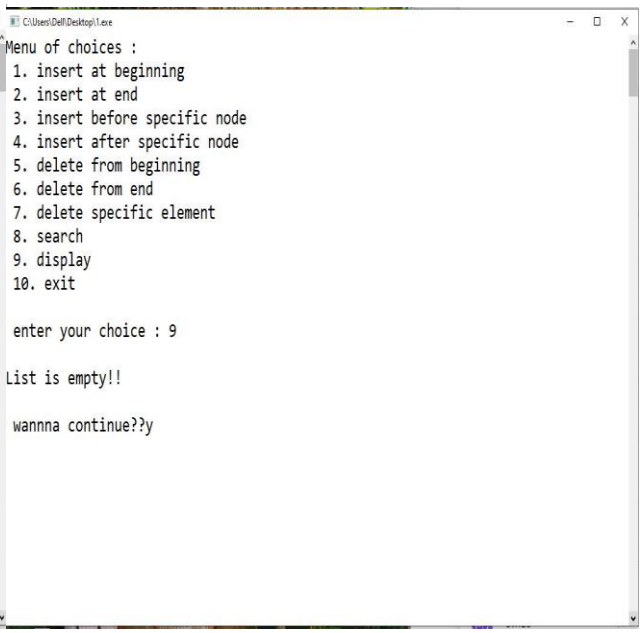
6. When Choice is 6



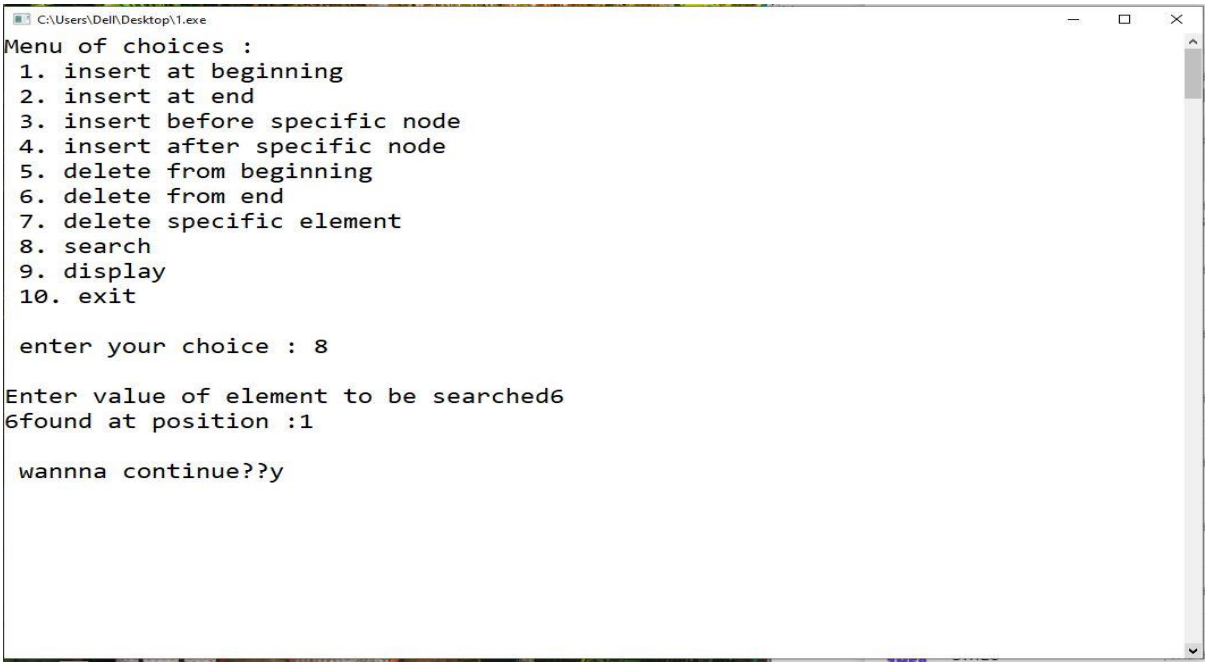


7. When Choice is 7

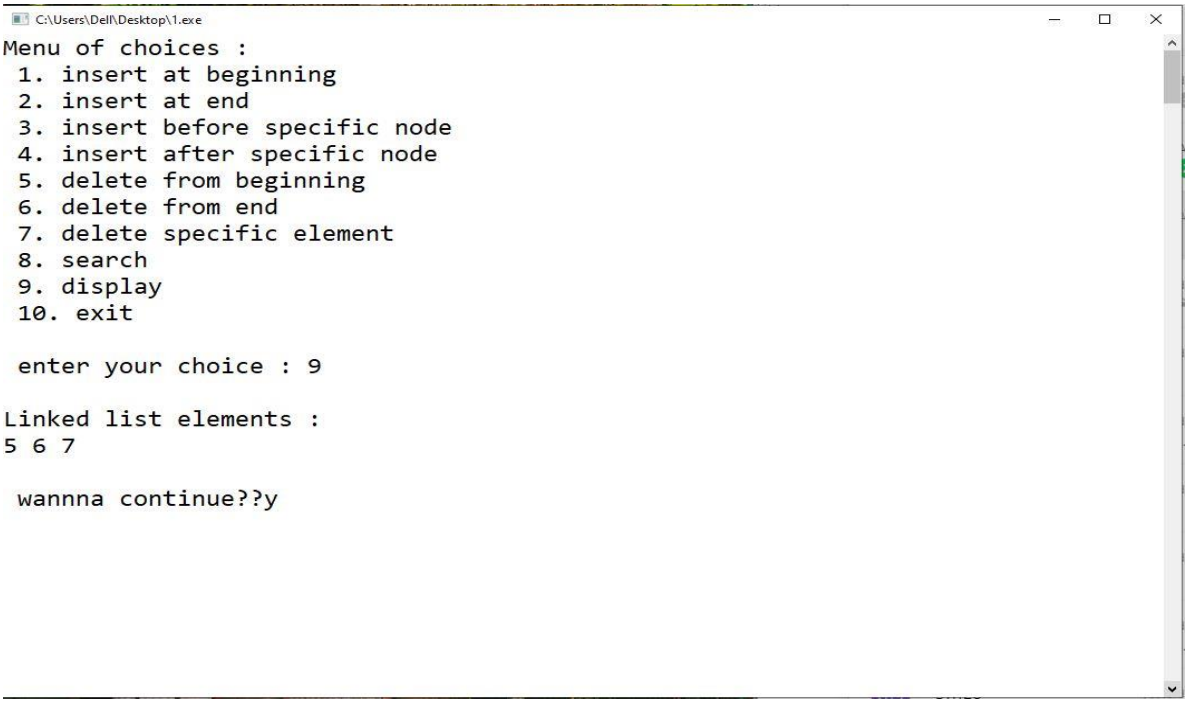




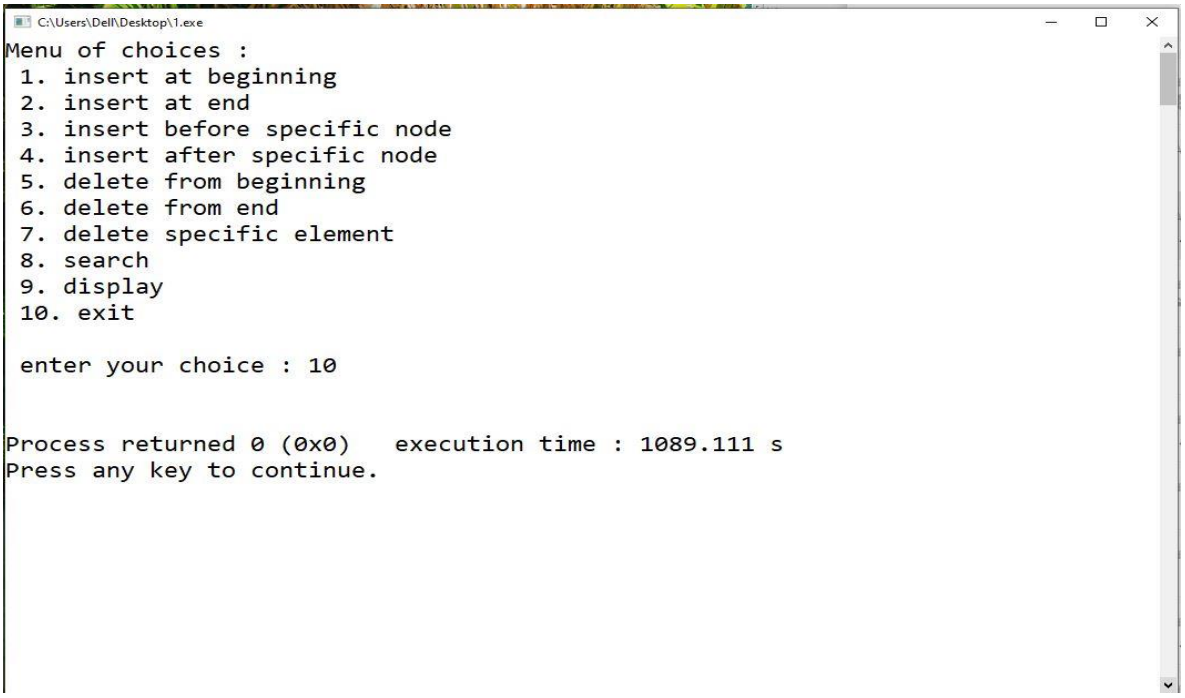
8. When Choice is 8



9. When Choice is 9



10. When Choice is 10



Q2. Write a program to count the number of occurrences of a given key in a singly linked list and then delete all the occurrences. For example, if given linked list is 1->2->1- >2->1->3->1 and given key is 1, then output should be 4. After deletion of all the occurrences of 1, the linked list is 2->2->3.

Ans2.

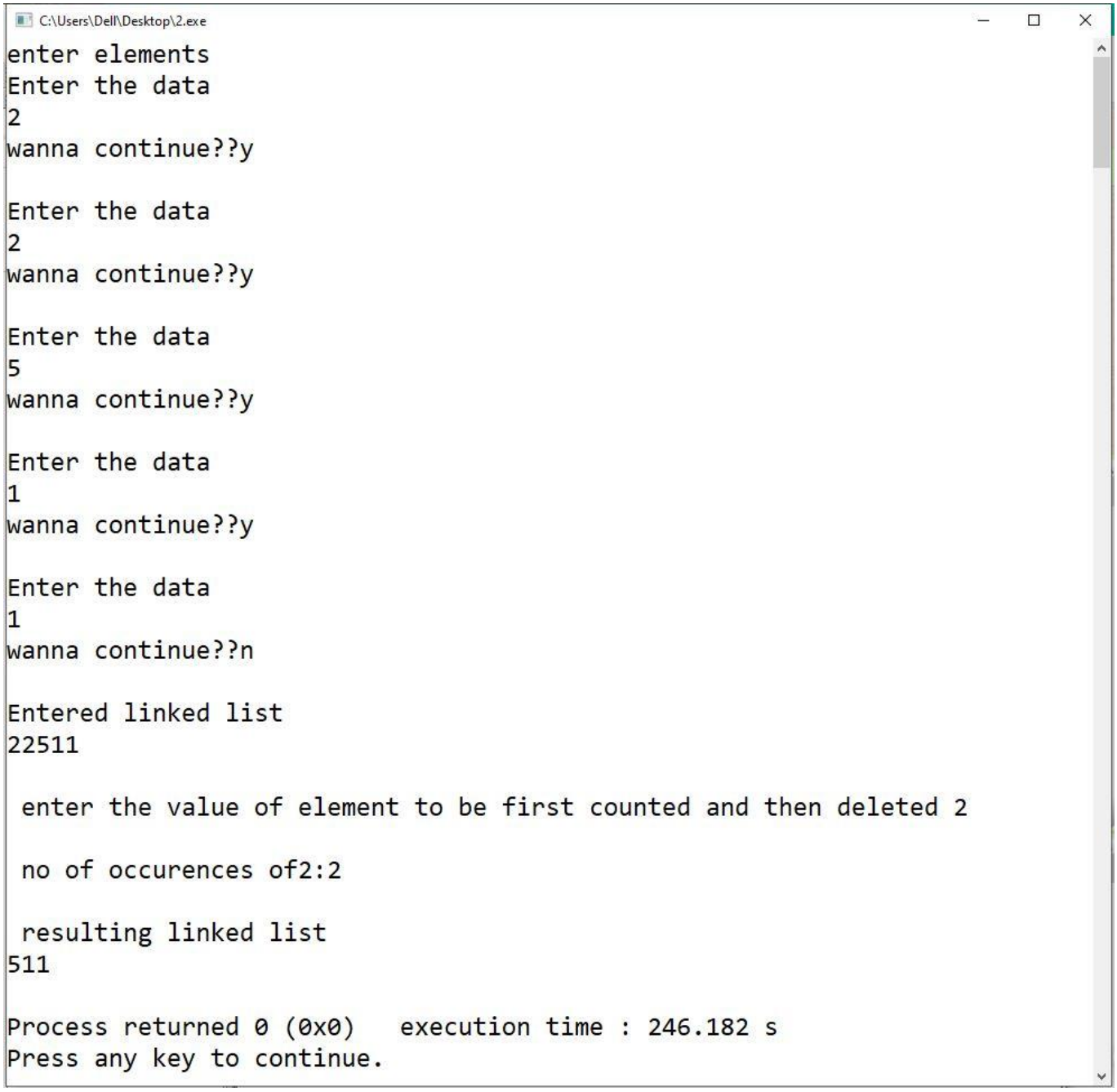
```
#include<iostream>
using namespace std;
struct node
{
        int data;
        node *next;
};
class l_list
{
        node *head;
        public:
        l_list()
        {
                head=NULL;
        }
        void create();
        void count_delete(int x);
        void display();
};
void l_list::create()
{
        char ch;
        do
        {
                node *temp=new node;
                cout<<"Enter the data\n";
                cin>>temp->data;
                temp->next=NULL;
                if(head == NULL)
                {
                        head=temp;
                }
                else
                {
                        node *temp1=head;
                        while(temp1->next!=NULL)
                        temp1=temp1->next;
                        temp1->next=temp;
                }
                cout<<"wanna continue??";
                cin>>ch;
                cout<<endl;
```

```
                }while(ch=='y' || ch=='Y');
        }
void l_list::count_delete(int x)
{
        if(head==NULL)
        cout<<"linked list is empty!!!\n";
        else
        {
                int f=0;
                node *temp=head,*pre=head;
                while(temp!=NULL)
                {
                        if(temp->data==x)
                        {

                                f++;
                                node *temp1=temp;
                                if(temp==head)
                                head=head->next;
                                else
                                pre->next=temp->next;
                                temp=temp->next;
                                delete temp1;
                        }
                        else
                        {
                                pre=temp;
                                temp=temp->next;
                        }
                }
                cout<<"\n no of occurences of"<<x<<":"<<f<<"\n";
        }
}
void l_list::display()
{
        if(head == NULL)
        cout<<"List is empty !!\n";
        else
        {
                node *temp=head;
                while(temp!=NULL)
                {
                        cout<<temp->data<<"";
                        temp=temp->next;
                }
                cout<<"\n";
        }
}
```

```
int main()
{
        int x;
        l_list a;
        cout<<"enter elements\n";
        a.create();
        cout<<"Entered linked list\n";
        a.display();
        cout<<"\n enter the value of element to be first counted and then deleted ";
        cin>>x;
        a.count_delete(x);
        cout<<"\n resulting linked list\n";
        a.display();
}
```

Outputs:



Q3. Write a program to find the middle of a linked list.

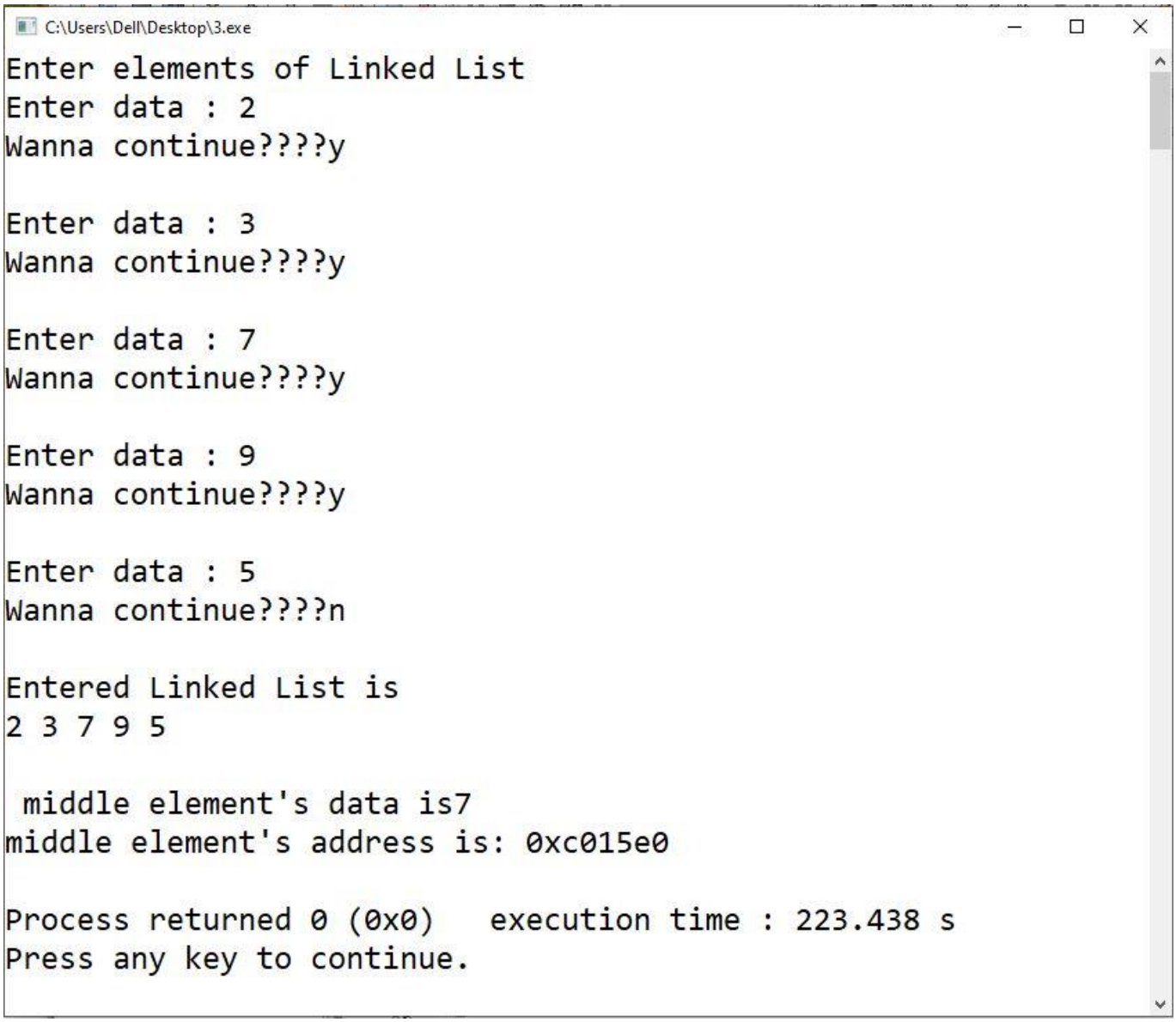
Ans3.

```
#include<iostream>
using namespace std;
struct node
{
int data;
node *next;
};
class L_List
{
node *head;
public:
L_List()
{
head=NULL;
}
void create();
void mid();
void display();
};
void L_List::create()
{
char ch;
do
{
node *temp=new node;
cout<<"Enter data : ";
cin>>temp->data;
temp->next=NULL;
if(head==NULL)
{
```

```
head=temp;
}
else
{
node *temp1=head;
while(temp1->next!=NULL)
temp1=temp1->next;
temp1->next=temp;
}
cout<<"Wanna continue????";
cin>>ch;
cout<<endl;
}while(ch=='y' || ch=='Y');
}
void L_List::mid()
{
if(head==NULL)
cout<<"List is empty!!!\n";
else
{
node *slow=head,*fast=head;
while(fast!=NULL && fast->next!=NULL)
{
slow=slow->next;
fast=(fast->next)->next;
}
cout<<"\n middle element's data is"<<slow->data<<endl;
cout<<"middle element's address is: "<<slow<<endl;
}
}
void L_List::display()
{
```

```
if(head==NULL)
cout<<"List is empty!!!!\n";
else
{
node *temp=head;
while(temp!=NULL)
{
cout<<temp->data<<" ";
temp=temp->next;
}
cout<<"\n";
}
}
int main()
{
L_List a;
cout<<"Enter elements of Linked List \n";
a.create();
cout<<"Entered Linked List is \n";
a.display();
a.mid();
return 0;
}
```

Outputs:


C:\Users\Dell\Desktop\3.exe
Enter elements of Linked List
Enter data : 2
Wanna continue????y
Enter data : 3
Wanna continue????y
Enter data : 7
Wanna continue????y
Enter data : 9
Wanna continue????y
Enter data : 5
Wanna continue????n
Entered Linked List is
2 3 7 9 5
middle element's data is7
middle element's address is: 0xc015e0
Process returned 0 (0x0) execution time : 223.438 s
Press any key to continue.

Q4. Write a program to reverse a linked list.

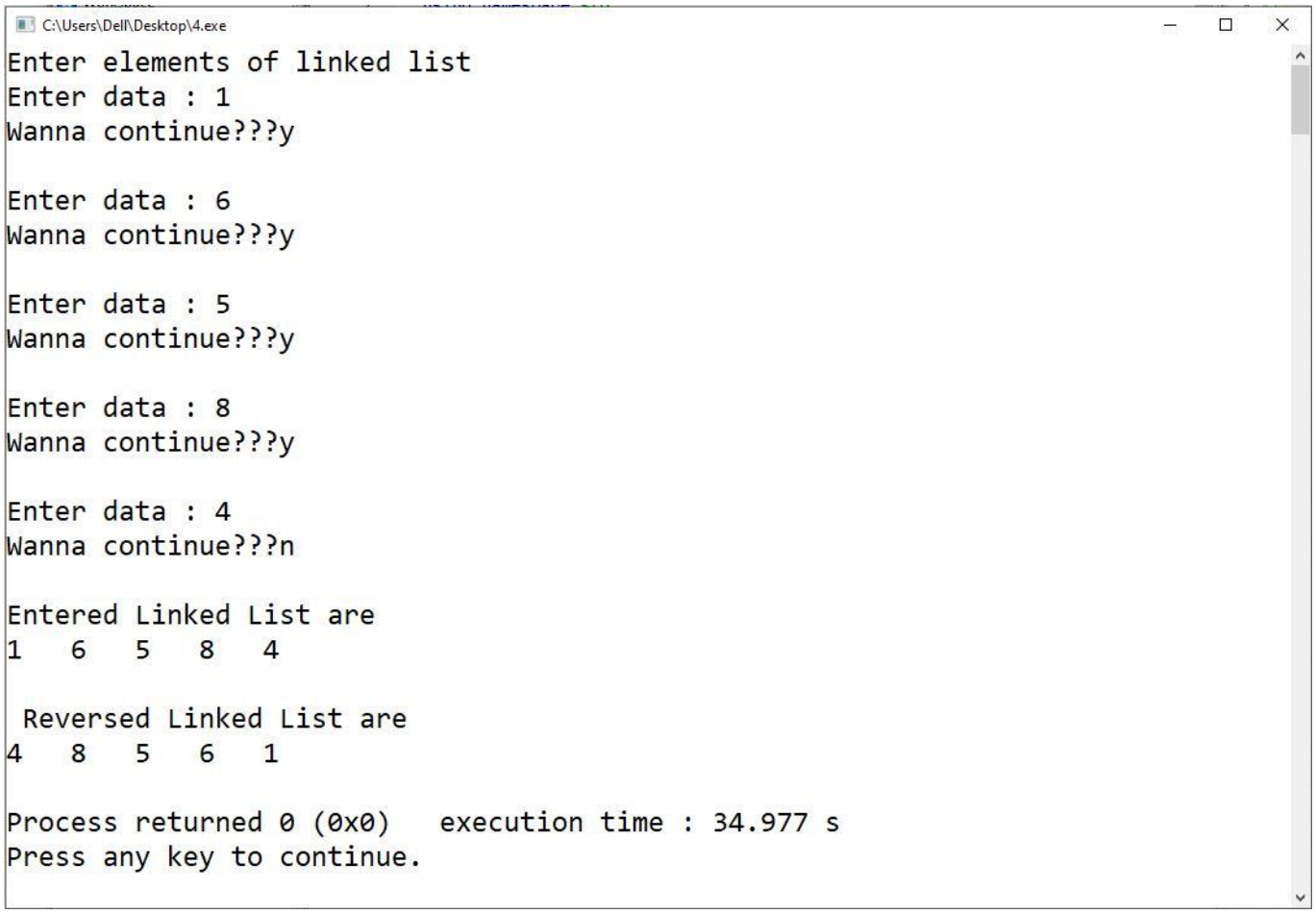
Ans4.

```
#include<iostream>
using namespace std;
struct node
{
int data;
node *next;
};
class L_List
{
node *head;
public:
L_List()
{
head=NULL;
}
void create();
void reverse();
void display();
};
void L_List::create()
{
char ch;
do
{
node *temp=new node;
cout<<"Enter data : ";
cin>>temp->data;
temp->next=NULL;
if(head==NULL)
{
```

```
head=temp;
}
else
{
node *temp1=head;
while(temp1->next!=NULL)
temp1=temp1->next;
temp1->next=temp;
}
cout<<"Wanna continue???";
cin>>ch;
cout<<endl;
}while(ch=='y' || ch=='Y');
}
void L_List::reverse()
{
if(head==NULL)
cout<<"List is empty!!!\n";
else
{
node *after=NULL,*curr=head,*prev=NULL;
while(curr!=NULL)
{
after=curr->next;
curr->next=prev;
prev=curr;
curr=after;
}
head=prev;
}
}
void L_List::display()
```

```
{
if(head==NULL)
cout<<"list is empty!!!!\n";
else
{
node *temp=head;
while(temp!=NULL)
{
cout<<temp->data<<"   ";
temp=temp->next;
}
cout<<"\n";
}
}
int main()
{
L_List a;
cout<<"Enter elements of linked list\n";
a.create();
cout<<"Entered Linked List are\n";
a.display();
a.reverse();
cout<<"\n Reversed Linked List are \n";
a.display();
return 0;
}
```

Outputs:


C:\Users\Dell\Desktop\4.exe
Enter elements of linked list
Enter data : 1
Wanna continue???y
Enter data : 6
Wanna continue???y
Enter data : 5
Wanna continue???y
Enter data : 8
Wanna continue???y
Enter data : 4
Wanna continue???n
Entered Linked List are
1 6 5 8 4
Reversed Linked List are
4 8 5 6 1
Process returned 0 (0x0) execution time : 34.977 s
Press any key to continue.