==========

Roll Number: SYCOC303 Division: C

PRN Number: 122B2B303 Batch: C4

Name: VINAYAK MADAN SHETE

==========

Problem Statement:

- **Write a C++ program to implement a doubly linked list and perform operations such as insert, delete, display and search element from it.**

==========

INPUT:

```
/*
 *     ====================================
 *            Program Name: DoublyLL.cpp
 *     Created on: November 30, 2022
 *      Author: Vinayak Shete
 *  ====================================
 */
#include<iostream>
using namespace std;
struct node
{
	int data;
	struct node* next;
	struct node* previous;
};
class DoublyLL
{
	struct node* head=NULL;
	public:
		void insertAtBeg();
		void insertAtEnd();
		void insertAtPos(int);
		void deleteAtBeg();
```

```
            void deleteAtEnd();
            void deleteAtPos();
            void searching();
            void display();
            void reverse();
};
//functions for insertion operation
//inserting at beginning
void DoublyLL::insertAtBeg()
{
      struct node* new_node=new node();
      cout<<"\n=================You are doing Insertion Operation at
Beginning====================";
      cout<<"\nEnter the number to be stored in the Linked List:";
      cin>>new_node->data;
      new_node->next=NULL;
      new_node->previous=NULL;
      //if entering for the first time
      if(head==NULL)
      {
            cout<<"\nYou are entering for the first
time..\n================================\n\n";
            head=new_node;
      }
      else
      {
            new_node->next=head;
            head->previous=new_node;
            head=new_node;
      }
}
//inserting at end
void DoublyLL::insertAtEnd()
{
      struct node* new_node=new node();
      cout<<"\n=================You are doing Insertion Operation at
Position====================";
      cout<<"\nEnter the number to be stored in the Linked List:";
```

```
        cin>>new_node->data;
        new_node->next=NULL;
        new_node->previous=NULL;
        //if entering for the first time
        if(head==NULL)
        {
                cout<<"\nYou are entering for the first
time..\n================================";
                head=new_node;
        }
        else
        {
                struct node *temp=head;
                while(temp->next!=NULL)
                {
                        temp=temp->next;
                }
                temp->next=new_node;
                new_node->previous=temp;
        }
}
//inserting at specific position
void DoublyLL::insertAtPos(int position)
{
        struct node* new_node=new node();
        cout<<"\n=================You are doing Insertion Operation at
Position====================";
        cout<<"\nEnter the number to be stored in the Linked List:";
        cin>>new_node->data;
        new_node->next=NULL;
        new_node->previous=NULL;
        //if entering for the first time
        if(head==NULL)
        {
                cout<<"\nYou are entering for the first
time..\n================================";
                head=new_node;
        }
```

```
        else
        {
                struct node *temp=head;
                for(int i=0;i<position-2;i++)
                {
                        temp=temp->next;
                }
                new_node->next=temp->next;
                temp->next=new_node;
                new_node->previous=temp;
        }
}
//functions for deletion operation
//deleting at beginning
void DoublyLL::deleteAtBeg()
{
        cout<<"\n=================You are doing Deletion Operation at
Beginning====================";
        if(head==NULL)
        {
                cout<<"\nList is empty!Deletion operation failed!";
        }
        else
        {
                struct node *temp=head;
                head=temp->next;
                head->previous=NULL;
                int del_data=temp->data;
                cout<<"\nElement "<<del_data<<" deleted successfully!\n";
                delete(temp);
        }
}
//deleting at beginning
void DoublyLL::deleteAtEnd()
{
        cout<<"\n=================You are doing Deletion Operation at
End====================";
```

```
	if(head==NULL)
	{
		cout<<"\nList is empty!Deletion operation failed!";
	}
	else
	{
		struct node *temp=head;
		struct node *temp1=head->next;
		while(temp1->next!=NULL)
		{
			temp=temp->next;
			temp1=temp1->next;
		}
		int del_data=temp1->data;
		temp->next=NULL;
		temp1->previous=NULL;
		cout<<"\nElement "<<del_data<<" deleted successfully!\n";
		delete(temp1);
	}
}
//deleting at specific location
void DoublyLL::deleteAtPos()
{
	int position;
	cout<<"\n=================You are doing Deletion Operation at
Position====================";
	if(head==NULL)
	{
		cout<<"\nList is empty!Deletion operation failed!";
	}
	else
	{
		cout<<"\nEnter the position you want to enter an element at:";
		cin>>position;
		struct node *temp=head;
		struct node *temp1=head->next;
		for(int i=0;i<position-2;i++)
```

```
		{
			temp=temp->next;
			temp1=temp1->next;
		}
		int del_data=temp1->data;
		temp->next=temp1->next;
		temp1->next->previous=temp;
		cout<<"\nElement "<<del_data<<" deleted successfully!\n";
		delete(temp1);
	}
}
//function for search an element into the list
void DoublyLL::searching()
{
	int ser_ele;
	int ser_pos=1;
	cout<<"\n=================You are doing Searching
Operation====================";
	if(head==NULL)
	{
		cout<<"\nList is empty!Searching operation failed!\n";
	}
	else
	{
		cout<<"\nEnter the element you want to search:";
		cin>>ser_ele;
		struct node *temp=head;
		while(temp->next!=NULL)
		{
			if(temp->data==ser_ele)
			{
				cout<<"\nElement found successfully at "<<ser_pos<<"
position.\n";
				return;
			}
			else
			{
```

```
                        ser_pos++;
                }


                temp=temp->next;
        }
        cout<<"\nElement is not present in the linked list..\n";
    }
}
//function for reversing the list
void DoublyLL::reverse()
{
    struct node* current=head;
    struct node* prev=NULL;
    struct node* next=NULL;
    cout<<"\n=================You are doing Reversing
Operation====================";
    if(head==NULL)
    {
        cout<<"\nList is empty!Reversing operation failed!\n";
    }
    else
    {
        while(current!=NULL)
        {
        next=current->next;
        current->next=prev;
        prev=current;
        current=next;
        }
        head=prev;
        cout<<"\nThe list is reversed successfully!\n";
    }
}
//function for displaying the list
void DoublyLL::display()
{
    struct node* temp=head;
```

```
	cout<<"\n====================Displaying the LINKED LIST====================";
	cout<<"\nLinkedList is:\tHEAD";
	while(temp!=NULL)
	{
		cout<<"<<-- -->>"<<temp->data;
		temp=temp->next;
	}
	cout<<"<<-- -->>NULL";
	cout<<"\n=================================================================\n\
n";
}
int main()
{
	DoublyLL obj;
	int ch,doch,insch,ins,pos,delch,del;
	int ser_ele,ser_i=0;
	cout<<"\n====================================Welcome=========================
==========\n";
	do
	{
		cout<<"1.Insert element into the List.\n2.Delete element from the
List.\n3.Search an element into the List.\n4.Display elements in the List.\n5.Reverse
the list.\n6.Exit.";
		cout<<"\nEnter your choice:";
		cin>>ch;
		switch(ch)
		{
			case 1:
				do{
					cout<<"\n============Insertion
Operations============";
					cout<<"\n1.Insert at beginning.\n2.Insert at specific
position.\n3.Insert at end.\nEnter your choice:";
					cin>>insch;
					switch(insch)
					{
						case 1:
							obj.insertAtBeg();
							cout<<"\nElement entered
successfully!\n";
```

```
                                        obj.display();
                                        break;
                                case 2:
                                        cout<<"\nEnter the position you want to
enter an element at:";
                                        cin>>pos;
                                        obj.insertAtPos(pos);
                                        cout<<"\nElement entered
successfully!\n";
                                        obj.display();
                                        break;
                                case 3:
                                        obj.insertAtEnd();
                                        cout<<"\nElement entered
successfully!\n";
                                        obj.display();
                                        break;
                                }
                                cout<<"\nDo you want to continue insertion
operation?[Press 1 for YES | Press 0 for NO}-->";
                                cin>>ins;
                        }while(ins==1);
                        break;
                case 2:
                        do{
                                cout<<"\n============Deletion
Operations============";
                                cout<<"\n1.Delete at beginning.\n2.Delete at specific
position.\n3.Delete at end.\nEnter your choice:";
                                cin>>insch;
                                switch(insch)
                                {
                                        case 1:
                                                obj.deleteAtBeg();
                                                obj.display();
                                                break;
                                        case 2:
                                                obj.deleteAtPos();
                                                obj.display();
```

```
                                        break;
                                case 3:
                                        obj.deleteAtEnd();
                                        obj.display();
                                        break;
                                }
                                cout<<"\nDo you want to continue deletion
operation?[Press 1 for YES | Press 0 for NO}-->";
                                cin>>del;
                        }while(del==1);
                        break;
                case 3:
                        obj.searching();
                        break;
                 case 4:
                        obj.display();
                        break;
                case 5:
                        obj.reverse();
                        obj.display();
                        break;
                case 6:
                        goto exit;
                        break;
            }
            cout<<"\nDo you want to continue with MAIN MENU?[Press 1 for YES | Press
0 for NO}-->";
            cin>>doch;
      }while(doch==1);
      exit:
            cout<<"\n==================================Thank
You!======================================";
}
```

==========

OUTPUT:

The insertion and deletion will contain the 3 sub-options as follows:

Insertion:

- ⇨ **Insertion at beginning**
- ⇨ **Insert at specific postion**
- ⇨ **Insertion ar end**

Deletion:

- ⇨ **Deletion at beginning**
- ⇨ **Deletion at specific position**
- ⇨ **Deletion at end**

Inserting elements into the Doubly Linked List:

```
====================================Welcome====================================
1.Insert element into the List.
2.Delete element from the List.
3.Search an element into the List.
4.Display elements in the List.
5.Reverse the list.
6.Exit.
Enter your choice:1

============Insertion Operations============
1.Insert at beginning.
2.Insert at specific position.
3.Insert at end.
Enter your choice:1

=================You are doing Insertion Operation at Beginning====================
Enter the number to be stored in the Linked List:10

You are entering for the first time..
================================


Element entered successfully!

=====================Displaying the LINKED LIST====================
LinkedList is:  HEAD<<-- -->>10<<-- -->>NULL
==================================================================


Do you want to continue insertion operation?[Press 1 for YES | Press 0 for NO}-->1

============Insertion Operations============
1.Insert at beginning.
2.Insert at specific position.
3.Insert at end.
Enter your choice:3

=================You are doing Insertion Operation at Position====================
Enter the number to be stored in the Linked List:60

Element entered successfully!
```

```
==================You are doing Insertion Operation at Position====================
Enter the number to be stored in the Linked List:20

Element entered successfully!

======================Displaying the LINKED LIST====================
LinkedList is:  HEAD<<-- -->>10<<-- -->>20<<-- -->>60<<-- -->>NULL
====================================================================


Do you want to continue insertion operation?[Press 1 for YES | Press 0 for NO}-->1

============Insertion Operations============
1.Insert at beginning.
2.Insert at specific position.
3.Insert at end.
Enter your choice:2

Enter the position you want to enter an element at:3

==================You are doing Insertion Operation at Position====================
Enter the number to be stored in the Linked List:30

Element entered successfully!

======================Displaying the LINKED LIST====================
LinkedList is:  HEAD<<-- -->>10<<-- -->>20<<-- -->>30<<-- -->>60<<-- -->>NULL
====================================================================


Do you want to continue insertion operation?[Press 1 for YES | Press 0 for NO}-->1

============Insertion Operations============
1.Insert at beginning.
2.Insert at specific position.
3.Insert at end.
Enter your choice:2

Enter the position you want to enter an element at:4

==================You are doing Insertion Operation at Position====================
Enter the number to be stored in the Linked List:40
```

```
Do you want to continue insertion operation?[Press 1 for YES | Press 0 for NO}-->1

============Insertion Operations============
1.Insert at beginning.
2.Insert at specific position.
3.Insert at end.
Enter your choice:2

Enter the position you want to enter an element at:5

==================You are doing Insertion Operation at Position====================
Enter the number to be stored in the Linked List:50

Element entered successfully!

======================Displaying the LINKED LIST====================
LinkedList is:  HEAD<<-- -->>10<<-- -->>20<<-- -->>30<<-- -->>40<<-- -->>50<<-- -->>60<<-- -->>NULL
====================================================================


Do you want to continue insertion operation?[Press 1 for YES | Press 0 for NO}-->
```

Deleting elements from the Doubly Linked List:

```
3.Delete at end.
Enter your choice:1

=================You are doing Deletion Operation at Beginning====================
Element 10 deleted successfully!

====================Displaying the LINKED LIST====================
LinkedList is:  HEAD<<-- -->>20<<-- -->>30<<-- -->>40<<-- -->>50<<-- -->>60<<-- -->>70<<-- -->>NULL
==================================================================


Do you want to continue deletion operation?[Press 1 for YES | Press 0 for NO}-->1

============Deletion Operations============
1.Delete at beginning.
2.Delete at specific position.
3.Delete at end.
Enter your choice:2

=================You are doing Deletion Operation at Position====================
Enter the position you want to enter an element at:4

Element 50 deleted successfully!

====================Displaying the LINKED LIST====================
LinkedList is:  HEAD<<-- -->>20<<-- -->>30<<-- -->>40<<-- -->>60<<-- -->>70<<-- -->>NULL
==================================================================


Do you want to continue deletion operation?[Press 1 for YES | Press 0 for NO}-->1

============Deletion Operations============
1.Delete at beginning.
2.Delete at specific position.
3.Delete at end.
Enter your choice:3

=================You are doing Deletion Operation at End====================
Element 70 deleted successfully!

====================Displaying the LINKED LIST====================
LinkedList is:  HEAD<<-- -->>20<<-- -->>30<<-- -->>40<<-- -->>60<<-- -->>NULL
==================================================================
```

Displaying the Doubly Linked List:

```
Enter your choice:4

======================Displaying the LINKED LIST======================
LinkedList is:  HEAD<<-- -->>20<<-- -->>30<<-- -->>40<<-- -->>60<<-- -->>NULL
======================================================================


Do you want to continue with MAIN MENU?[Press 1 for YES | Press 0 for NO}-->
```

Searching into Doubly Linked List:

```
Do you want to continue with MAIN MENU?[Press 1 for YES | Press 0 for NO}-->1
1.Insert element into the List.
2.Delete element from the List.
3.Search an element into the List.
4.Display elements in the List.
5.Reverse the list.
6.Exit.
Enter your choice:3

=================You are doing Searching Operation====================
Enter the element you want to search:40

Element found successfully at 3 position.

Do you want to continue with MAIN MENU?[Press 1 for YES | Press 0 for NO}-->1
1.Insert element into the List.
2.Delete element from the List.
3.Search an element into the List.
4.Display elements in the List.
5.Reverse the list.
6.Exit.
Enter your choice:3

=================You are doing Searching Operation====================
Enter the element you want to search:70

Element is not present in the linked list..

Do you want to continue with MAIN MENU?[Press 1 for YES | Press 0 for NO}-->
```

Reversing the Doubly Linked List:

```
Do you want to continue with MAIN MENU?[Press 1 for YES | Press 0 for NO}-->1
1.Insert element into the List.
2.Delete element from the List.
3.Search an element into the List.
4.Display elements in the List.
5.Reverse the list.
6.Exit.
Enter your choice:5

=================You are doing Reversing Operation====================
The list is reversed successfully!

====================Displaying the LINKED LIST====================
LinkedList is:  HEAD<<-- -->>60<<-- -->>40<<-- -->>30<<-- -->>20<<-- -->>NULL
==================================================================


Do you want to continue with MAIN MENU?[Press 1 for YES | Press 0 for NO}-->1
1.Insert element into the List.
2.Delete element from the List.
3.Search an element into the List.
4.Display elements in the List.
5.Reverse the list.
6.Exit.
Enter your choice:6

==================================Thank You!=======================================
```

==========