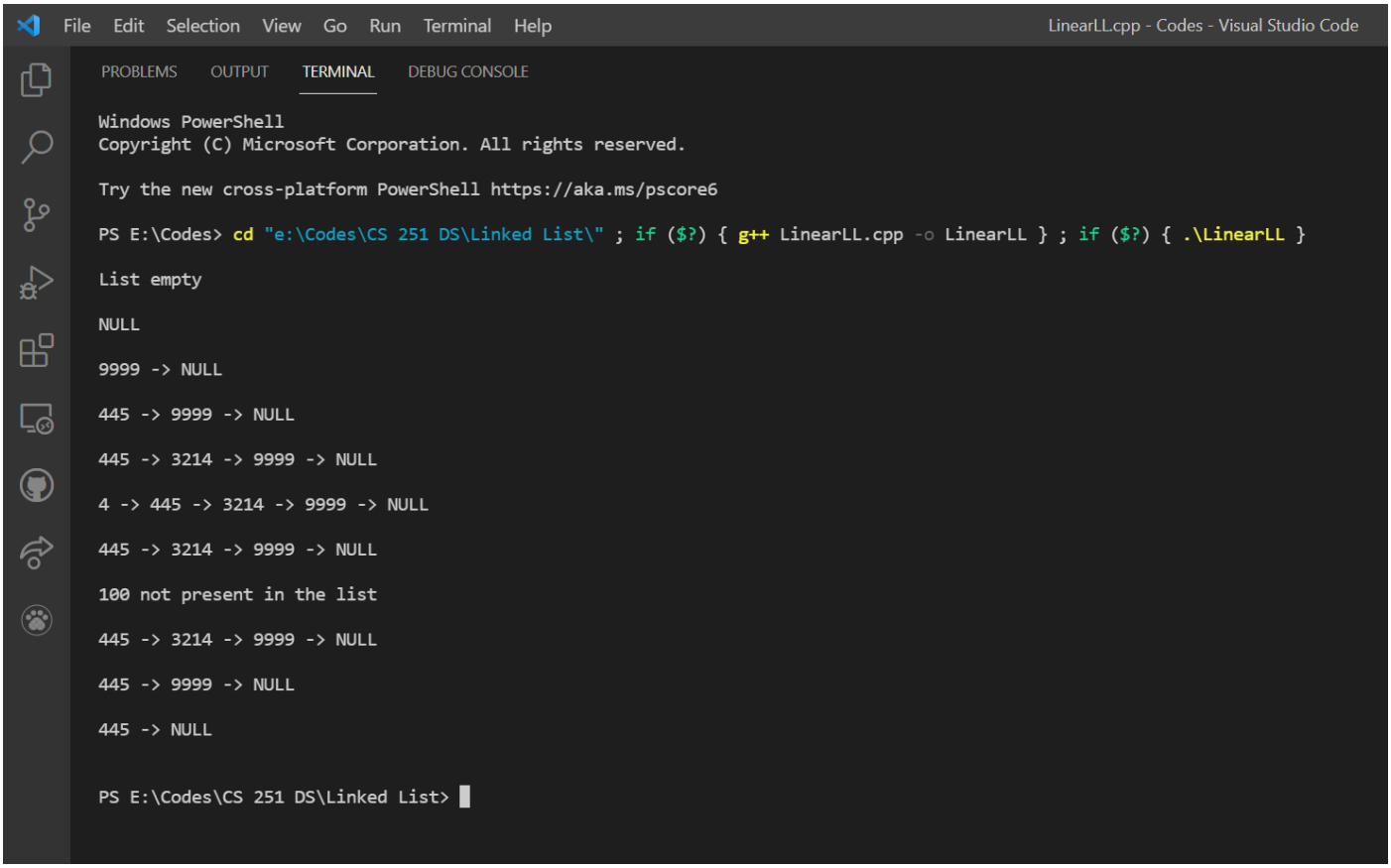
Linked List

```cpp
// Aneesh Panchal
// 2K20/MC/21

#include<bits/stdc++.h>
using namespace std;

class Node {
public:
    int data;
    Node* next=NULL;
};

class Linkedlist {
    Node* head;

    public:
        Linkedlist(){head = NULL;}

    void insert(int data_){
        Node* newNode = new Node();
        newNode->data = data_;

        Node* temp=head;

        if(head == NULL){
            head = newNode;
            return;
        }

        else if(head->data > data_){
            newNode->next = head;
            head = newNode;
            return;
        }

        while(temp->next != NULL && temp->next->data<data_){
            temp = temp->next;
        }
        newNode->next = temp->next;
        temp->next = newNode;
    }

    void print(){
        Node* temp=head;

        while(temp!=NULL){
            cout<< temp->data <<" -> ";
            temp = temp->next;
        }
        cout<<"NULL"<<endl<<endl;
    }
```

```
    void deleted(int data_){
        Node* temp=head;

        if(head==NULL){
            cout<<"List empty"<<endl<<endl;
            return;
        }

        else if(head->data == data_){
            head = temp->next;
            return;
        }

        while(temp->next->data!=data_){
            if(temp->next->next==NULL){
                cout<<data_<<" "<<"not present in the list"<<endl<<endl;
                return;
            }
            temp=temp->next;
        }
        Node* del = temp->next;
        temp->next = temp->next->next;
        delete del;
    }
};

int main()
{
    Linkedlist linkedL;
    cout<<endl;
    linkedL.deleted(4);
    linkedL.print();

    linkedL.insert(9999);
    linkedL.print();

    linkedL.insert(445);
    linkedL.print();

    linkedL.insert(3214);
    linkedL.print();

    linkedL.insert(4);
    linkedL.print();

    linkedL.deleted(4);
    linkedL.print();

    linkedL.deleted(100);
    linkedL.print();

    linkedL.deleted(3214);
    linkedL.print();
```

```
    linkedL.deleted(9999);
    linkedL.print();
    cout<<endl;

    return 0;
}
```



Doubly Ended Linked List

```cpp
// Aneesh Panchal
// 2K20/MC/21

#include<bits/stdc++.h>
using namespace std;

class Node {
public:
    int data;
    Node* next=NULL;
    Node* prev=NULL;
};

class DoublyEndedLinkedList {
    Node* head;

    public:
        DoublyEndedLinkedList(){head = NULL;}

    void insert(int data_){
        Node* newNode = new Node();
        newNode->data = data_;

        Node* temp=head;

        if(head == NULL){
            head = newNode;
            return;
        }

        else if(head->data > data_){
            newNode->prev=NULL;
            newNode->next = head;
            head->prev=newNode;
            head=newNode;
            return;
        }

        while(temp->next != NULL && temp->next->data<data_){
            temp = temp->next;
        }
        newNode->next = temp->next;
        newNode->prev = temp;
        temp->next = newNode;
        temp->next->prev = newNode;
    }

    void print(){
        Node* temp=head;

        while(temp!=NULL){
```

```
            cout<< temp->data <<" -> ";
            temp = temp->next;
        }
        cout<<"NULL"<<endl<<endl;
    }

    void deleted(int data_){
        Node* temp=head;

        if(head==NULL){
            cout<<"List empty"<<endl<<endl;
            return;
        }

        else if(head->data == data_){
            head = head->next;
            head->prev = NULL;
            return;
        }

        while(temp->next->data!=data_){
            if(temp->next->next==NULL){
                cout<<data_<<" "<<"not present in the list"<<endl<<endl;
                return;
            }
            temp=temp->next;
        }

        if(temp->next->next == NULL)
        {
            Node* del = temp->next->next;
            temp->next = NULL;
            delete del;
        }
        else
        {
            Node* del = temp->next;
            temp->next = del->next;
            del->next->prev = temp;
            delete del;
        }
    }
};

int main()
{
    DoublyEndedLinkedList DElinkedL;
    cout<<endl;
    DElinkedL.deleted(12);
    DElinkedL.print();

    DElinkedL.insert(2365);
    DElinkedL.print();
```

```
    DElinkedL.insert(440);
    DElinkedL.print();

    DElinkedL.insert(1021);
    DElinkedL.print();

    DElinkedL.insert(144);
    DElinkedL.print();

    DElinkedL.deleted(144);
    DElinkedL.print();

    DElinkedL.deleted(100);
    DElinkedL.print();

    DElinkedL.deleted(1021);
    DElinkedL.print();

    DElinkedL.deleted(2365);
    DElinkedL.print();
    cout<<endl;

    return 0;
}
```