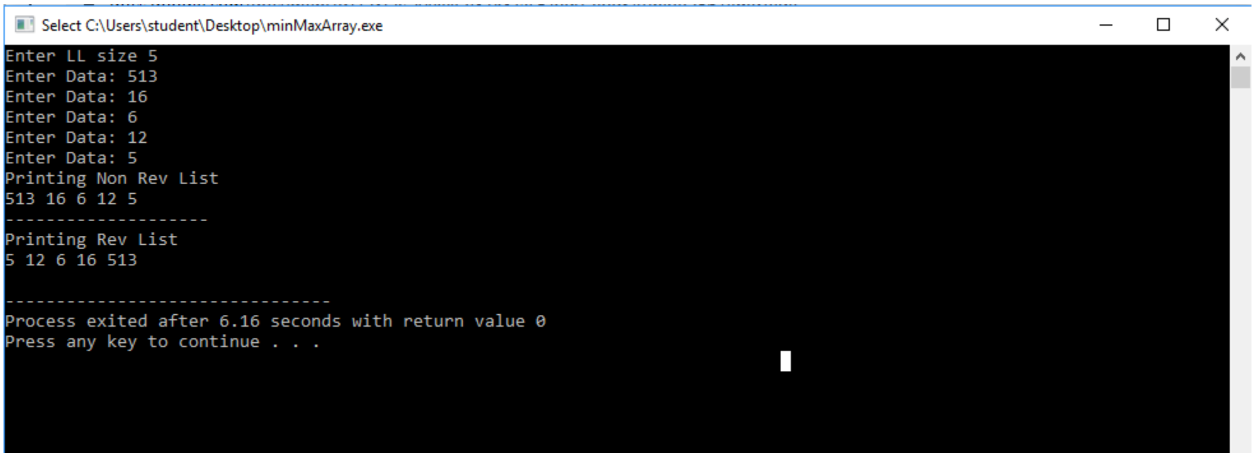
Q1 :

```
#include<iostream>
using namespace std;

struct Node
{
	int data;
	struct Node* next;
	Node(int data)
	{
		this->data = data;
		next = NULL;
	}
};

struct LinkList
{
	Node* head = NULL;

	void pb(int data)
	{
		Node *tmp = new Node(data);

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

		Node *smp = head;

		while(smp->next != NULL)
		smp=smp->next;

		smp->next = tmp;
	}

	void pr()
	{
		struct Node* temp = head;
		while(temp!=NULL)
		{
			cout<<temp->data<<" ";
```

```
                temp = temp->next;
            }
            cout<<endl;
        }

        void rev()
        {
            Node* cur = head;
            Node* prev = NULL, *next = NULL;

            while(cur != NULL)
            {
                next = cur->next;
                cur->next = prev;
                prev = cur;
                cur = next;
            }
            head = prev;
            pr();
        }

    };



    int main(){

        int llSize;
        cout << "Enter LL size ";
        cin >>  llSize;
        int z;

        LinkList lx;

        for (int itt=0;itt<llSize;itt++){
            cout<<"Enter Data: ";
            cin>>z;
            lx.pb(z);
        }
        cout<<"Printing Non Rev List"<<endl;
        lx.pr();
        cout<<"--------------------"<<endl;
        cout<<"Printing Rev List"<<endl;
        lx.rev();


}
```

Screenshot :

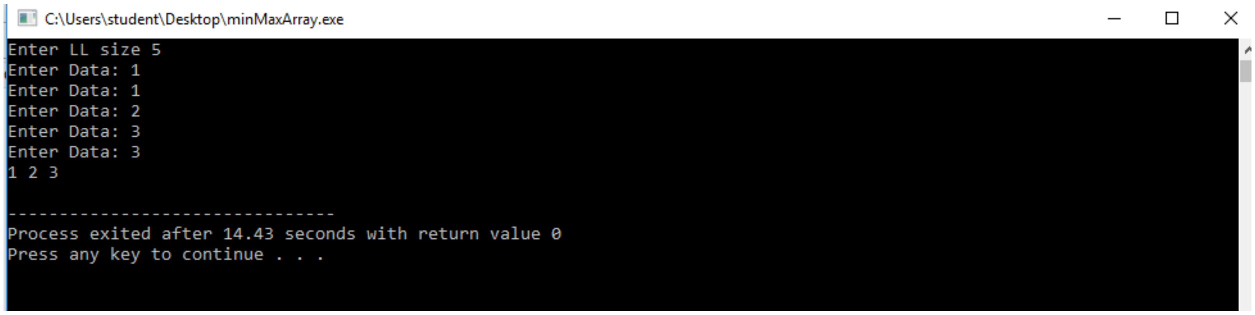


```
Enter LL size 5
Enter Data: 513
Enter Data: 16
Enter Data: 6
Enter Data: 12
Enter Data: 5
Printing Non Rev List
513 16 6 12 5
--------------------
Printing Rev List
5 12 6 16 513

--------------------------------
Process exited after 6.16 seconds with return value 0
Press any key to continue . . .
```

Q2 :

```
#include<iostream>
using namespace std;

struct Node
{
        int data;
        struct Node* next;
        Node(int data)
        {
                this->data = data;
                next = NULL;
        }
};

struct LinkList
{
        Node* head = NULL;

        void pb(int data)
        {
                Node *tmp = new Node(data);

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

                Node *smp = head;

                while(smp->next != NULL)
                smp=smp->next;

                smp->next = tmp;
        }

        void pr()
        {
                struct Node* temp = head;
                while(temp!=NULL)
                {
                        cout<<temp->data<<" ";
                        temp = temp->next;
                }
                cout<<endl;
        }

void delDup() {

   Node* current = head;
   Node* dtx;

   if (current == NULL)
   return;

   while (current->next != NULL)
   {

   if (current->data == current->next->data)
   {
      dtx = current->next->next;
      current->next = dtx;
   }
   else
        current = current->next;

   }

}

};



int main(){

        int llSize;
        cout << "Enter LL size ";
        cin >>  llSize;
        int z;

        LinkList lx;

        for (int itt=0;itt<llSize;itt++){
                cout<<"Enter Data: ";
                cin>>z;
                lx.pb(z);
        }



        lx.delDup();
        lx.pr();

}
```

O/P :

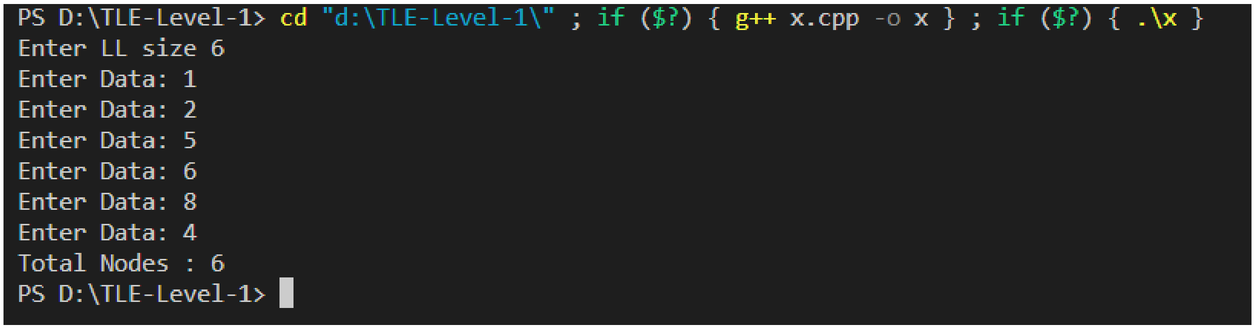


```
Enter LL size 5
Enter Data: 1
Enter Data: 1
Enter Data: 2
Enter Data: 3
Enter Data: 3
1 2 3

--------------------------------
Process exited after 14.43 seconds with return value 0
Press any key to continue . . .
```

Q3:

```
#include <iostream>
using namespace std;

struct Node
{
        int data;
        struct Node *next;
};

struct LinkList
{
        int count = 0;
        struct Node *Head = NULL;
        struct Node *Tail = NULL;

        void pb(int data)
        {

        struct Node *newNode = (struct Node *)malloc(sizeof(struct Node));

        newNode->data = data;
        if (Head == NULL)
        {
        Head = newNode;
        Tail = newNode;
        newNode->next = Head;
        }
        else
        {
        Tail->next = newNode;
        Tail = newNode;
        Tail->next = Head;
        }
        }

        void countNodes()
        {
        struct Node *Current = Head;
        do
        {
        count++;
        Current = Current->next;
        } while (Current != Head);
        }
};

int main()
{

        int llSize;
        cout << "Enter LL size ";
        cin >> llSize;
        int z;

        LinkList lx;

        for (int itt = 0; itt < llSize; itt++)
        {
        cout << "Enter Data: ";
        cin >> z;
        lx.pb(z);
        }
        lx.countNodes();
        cout << "Total Nodes : " << lx.count;
}
```

O/P:



```
PS D:\TLE-Level-1> cd "d:\TLE-Level-1\" ; if ($?) { g++ x.cpp -o x } ; if ($?) { .\x }
Enter LL size 6
Enter Data: 1
Enter Data: 2
Enter Data: 5
Enter Data: 6
Enter Data: 8
Enter Data: 4
Total Nodes : 6
PS D:\TLE-Level-1>
```

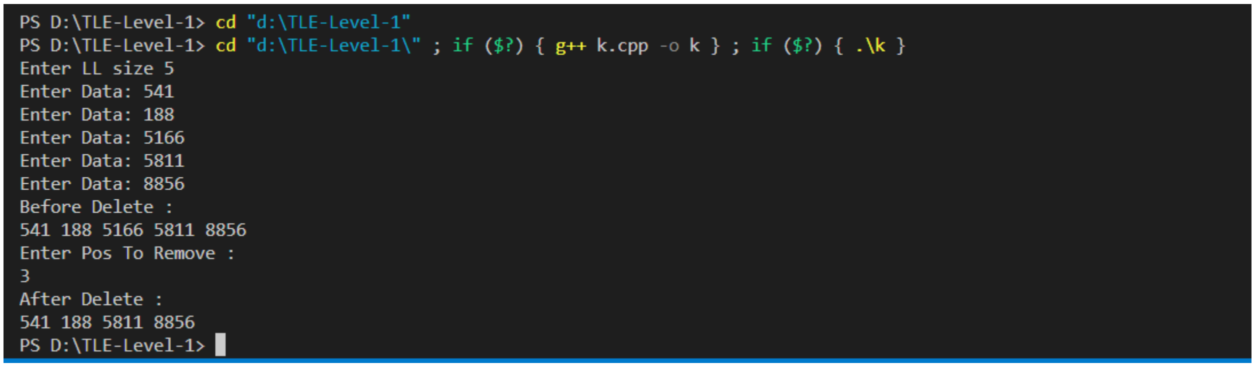
Q4:

```
#include <iostream>
using namespace std;

struct Node {
        int data;
        Node* next;
        Node* prev;
};

struct LinkList {
        Node* head;
        LinkList(){
        head = NULL;
        }

        void pb(int newElement) {
        Node* newNode = new Node();
        newNode->data = newElement;
        newNode->next = NULL;
        newNode->prev = NULL;
        if(head == NULL) {
        head = newNode;
        } else {
        Node* temp = head;
        while(temp->next != NULL)
        temp = temp->next;
        temp->next = newNode;
        newNode->prev = temp;
        }
        }

        void rm(int position) {
        if(position < 1) {
        cout<<"Position Error";
        } else if (position == 1 && head != NULL) {
        Node* nodeToDelete = head;
        head = head->next;
        free(nodeToDelete);
        if(head != NULL)
        head->prev = NULL;
        } else {
        Node* temp = head;
```

```
    }

    for(int i = 1; i < position-1; i++) {
    if(temp != NULL) {
    temp = temp->next;
    }
    }
    if(temp != NULL && temp->next != NULL) {
    Node* nodeToDelete = temp->next;
    temp->next = temp->next->next;
    if(temp->next->next != NULL)
    temp->next->next->prev = temp->next;
    free(nodeToDelete);
    } else {
    cout<<"Already Null";
    }
    }
    }

    void pr() {
    Node* temp = head;
    if(temp != NULL) {
    while(temp != NULL) {
    cout<<temp->data<<" ";
    temp = temp->next;
    }
    cout<<endl;
    }
    }
};

int main() {

    int llSize,posRem;
    cout << "Enter LL size ";
    cin >> llSize;
    int z;

    LinkList lx;

    for (int itt = 0; itt < llSize; itt++)
    {
    cout << "Enter Data: ";
    cin >> z;
    lx.pb(z);
    }
```

```
          cout<<"Before Delete : "<<endl;
          lx.pr();

          cout << "Enter Pos To Remove : "<<endl;
          cin >> posRem;

          lx.rm(posRem);
          cout<<"After Delete : "<<endl;
          lx.pr();

   return 0;
}
```

O/P :



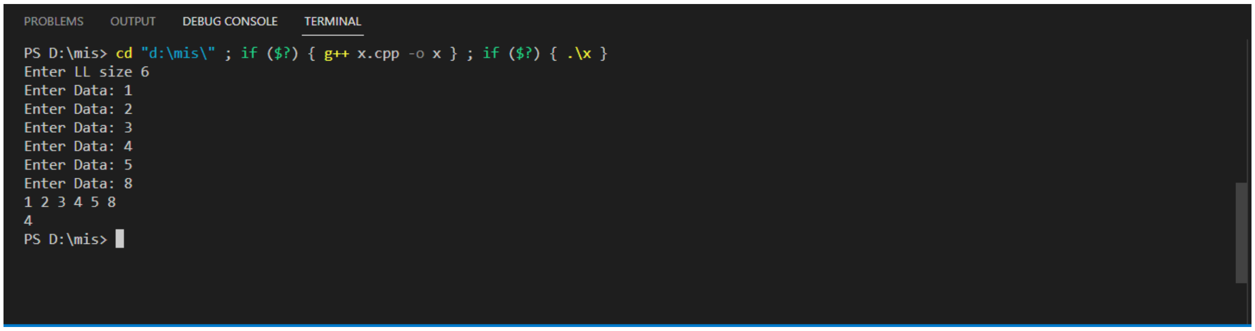
```
PS D:\TLE-Level-1> cd "d:\TLE-Level-1"
PS D:\TLE-Level-1> cd "d:\TLE-Level-1\" ; if ($?) { g++ k.cpp -o k } ; if ($?) { .\k }
Enter LL size 5
Enter Data: 541
Enter Data: 188
Enter Data: 5166
Enter Data: 5811
Enter Data: 8856
Before Delete :
541 188 5166 5811 8856
Enter Pos To Remove :
3
After Delete :
541 188 5811 8856
PS D:\TLE-Level-1> 
```

Q5:

```
#include<iostream>
using namespace std;
#define endl '\n'

struct Node
{
        int data;
        struct Node *next;
        Node(int data)
        {
        this->data = data;
        next = NULL;
        }
};

struct LinkList
{
        Node *head = NULL;

        void pb(int data)
        {
        Node *tmp = new Node(data);

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

        Node *smp = head;

        while (smp->next != NULL)
        smp = smp->next;

        smp->next = tmp;
        }

        void pr()
        {
        struct Node *temp = head;
        while (temp != NULL)
        {
```

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

    int getLen()
    {
    int len = 0;
    class Node *temp = head;
    while (temp)
    {
    len++;
    temp = temp->next;
    }
    return len;
    }

    void pm()
    {

    if (head)
    {
    int len = getLen();
    class Node *temp = head;

    int midIdx = len / 2;
    while (midIdx--)
    {
        temp = temp->next;
    }
    cout << temp->data << endl;
    }
    }
};

int main()
{

    int llSize;
    cout << "Enter LL size ";
    cin >> llSize;
    int z;
    LinkList lx;

    for (int itt = 0; itt < llSize; itt++)
    {
    cout << "Enter Data: ";
    cin >> z;
    lx.pb(z);
    }

    lx.pr();
    lx.pm();
}
```

O/P :



```
PS D:\mis> cd "d:\mis\" ; if ($?) { g++ x.cpp -o x } ; if ($?) { .\x }
Enter LL size 6
Enter Data: 1
Enter Data: 2
Enter Data: 3
Enter Data: 4
Enter Data: 5
Enter Data: 8
1 2 3 4 5 8
4
PS D:\mis> 
```