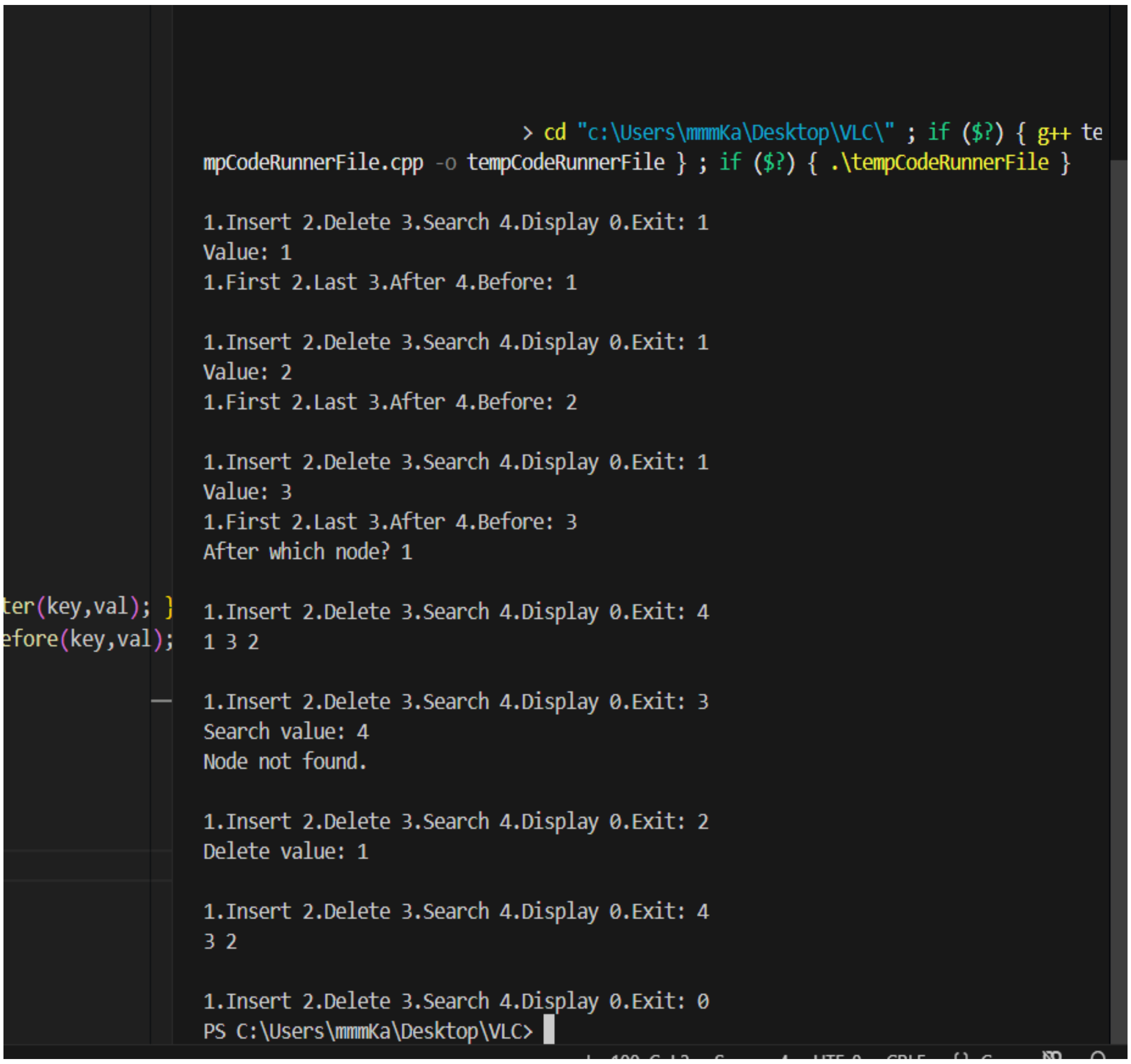
ASSIGMENT 6

Question 1

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

struct DNode {
    int data;
    DNode *prev, *next;
};

class DoublyLinkedList {
private:
    DNode* head;
public:
    DoublyLinkedList() : head(nullptr) {}

    void insertFirst(int val) {
        DNode* n = new DNode{val, nullptr, head};
        if (head) head->prev = n;
        head = n;
    }

    void insertLast(int val) {
        DNode* n = new DNode{val, nullptr, nullptr};
        if (!head) { head = n; return; }
        DNode* temp = head;
        while (temp->next) temp = temp->next;
        temp->next = n;
        n->prev = temp;
```

```
}

void insertAfter(int key, int val) {
    DNode* temp = head;
    while (temp && temp->data != key) temp = temp->next;
    if (!temp) { cout << "Node not found.\n"; return; }
    DNode* n = new DNode{val, temp, temp->next};
    if (temp->next) temp->next->prev = n;
    temp->next = n;
}

void insertBefore(int key, int val) {
    if (!head) return;
    if (head->data == key) { insertFirst(val); return; }
    DNode* temp = head;
    while (temp && temp->data != key) temp = temp->next;
    if (!temp) { cout << "Node not found.\n"; return; }
    DNode* n = new DNode{val, temp->prev, temp};
    temp->prev->next = n;
    temp->prev = n;
}

void deleteNode(int key) {
    if (!head) return;
    DNode* temp = head;
    while (temp && temp->data != key) temp = temp->next;
    if (!temp) { cout << "Node not found.\n"; return; }
    if (temp->prev) temp->prev->next = temp->next;
    else head = temp->next;
    if (temp->next) temp->next->prev = temp->prev;
    delete temp;
    }

    void search(int key) {
        DNode* temp = head;
        int pos = 1;
        while (temp) {
            if (temp->data == key) { cout << "Node found at position " << pos << endl; return; }
            temp = temp->next; pos++;
        }
        cout << "Node not found.\n";
    }

    void display() {
        DNode* temp = head;
        while (temp) { cout << temp->data << " "; temp = temp->next; }
        cout << endl;
    }
};

int main() {
    DoublyLinkedList dll;
    int choice, val, key;

    while (true) {
        cout << "\n1.Insert 2.Delete 3.Search 4.Display 0.Exit: ";
        cin >> choice;
        if (choice == 0) break;
        switch(choice) {
            case 1:
                cout << "Value: "; cin >> val;
                cout << "1.First 2.Last 3.After 4.Before: "; cin >> key;
            if(key==1) dll.insertFirst(val);
            else if(key==2) dll.insertLast(val);
            else if(key==3) { cout << "After which node? "; cin >> key; dll.insertAfter(key,val); }
            else if(key==4) { cout << "Before which node? "; cin >> key; dll.insertBefore(key,val); }
            break;
        case 2: cout << "Delete value: "; cin >> val; dll.deleteNode(val); break;
        case 3: cout << "Search value: "; cin >> val; dll.search(val); break;
        case 4: dll.display(); break;
      }
    }
}
```



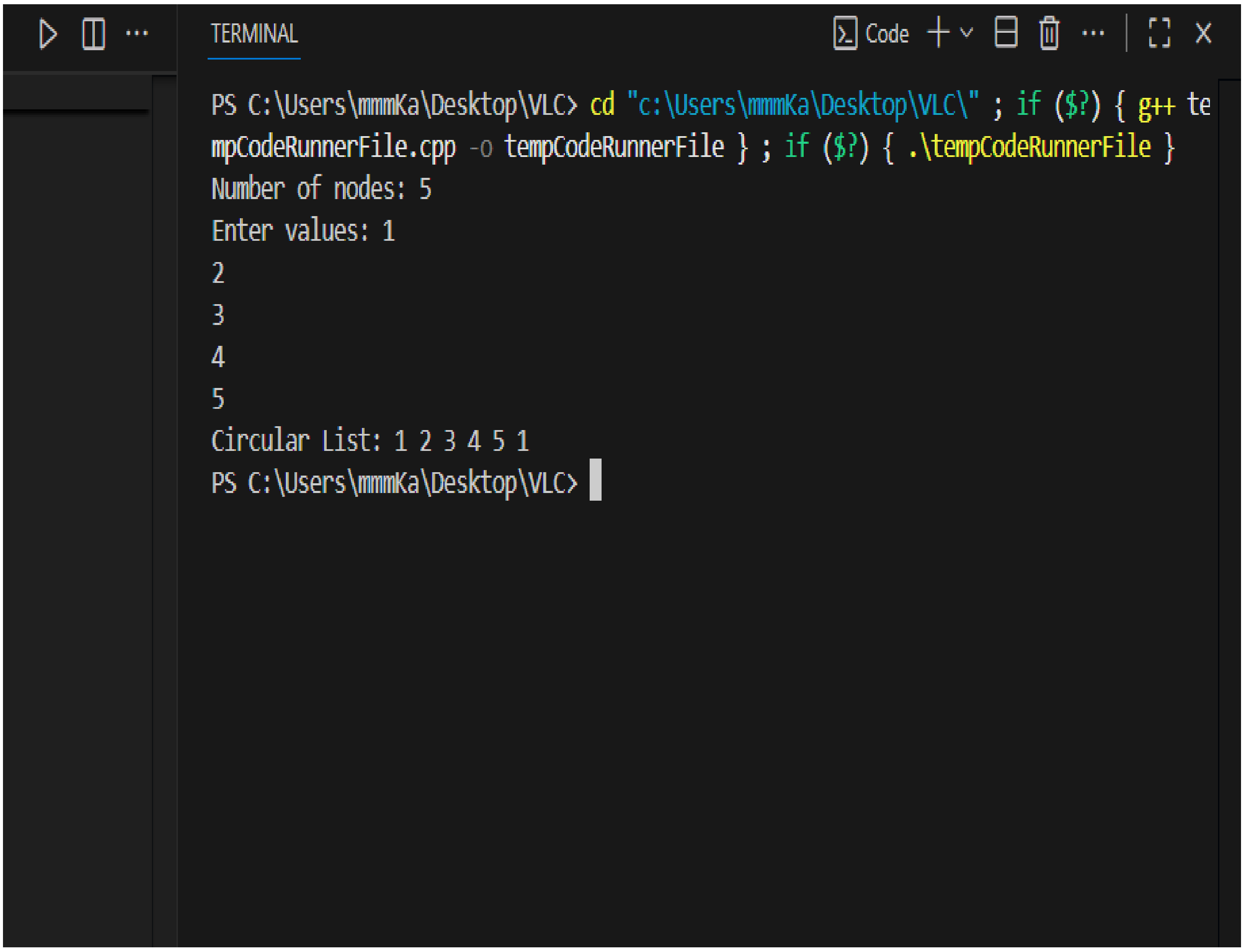
```
ter(key,val); }
efore(key,val);
```

```
> cd "c:\Users\mmmKa\Desktop\VLC\" ; if ($?) { g++ te
mpCodeRunnerFile.cpp -o tempCodeRunnerFile } ; if ($?) { .\tempCodeRunnerFile }

1.Insert 2.Delete 3.Search 4.Display 0.Exit: 1
Value: 1
1.First 2.Last 3.After 4.Before: 1

1.Insert 2.Delete 3.Search 4.Display 0.Exit: 1
Value: 2
1.First 2.Last 3.After 4.Before: 2

1.Insert 2.Delete 3.Search 4.Display 0.Exit: 1
Value: 3
1.First 2.Last 3.After 4.Before: 3
After which node? 1

1.Insert 2.Delete 3.Search 4.Display 0.Exit: 4
1 3 2

1.Insert 2.Delete 3.Search 4.Display 0.Exit: 3
Search value: 4
Node not found.

1.Insert 2.Delete 3.Search 4.Display 0.Exit: 2
Delete value: 1

1.Insert 2.Delete 3.Search 4.Display 0.Exit: 4
3 2

1.Insert 2.Delete 3.Search 4.Display 0.Exit: 0
PS C:\Users\mmmKa\Desktop\VLC>
```

Question 2

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


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


class CircularLinkedList {
private:
    CNode* head;
public:
    CircularLinkedList() : head(nullptr) {}


    void insertLast(int val) {
        CNode* n = new CNode{val,nullptr};
        if(!head) { head=n; n->next=head; return; }
        CNode* temp=head;
        while(temp->next!=head) temp=temp->next;
        temp->next=n;
        n->next=head;
    }


    void display() {
        if(!head) return;
        CNode* temp=head;
        do { cout << temp->data << " "; temp=temp->next; } while(temp!=head);
        cout << head->data << endl;
    }
};
```

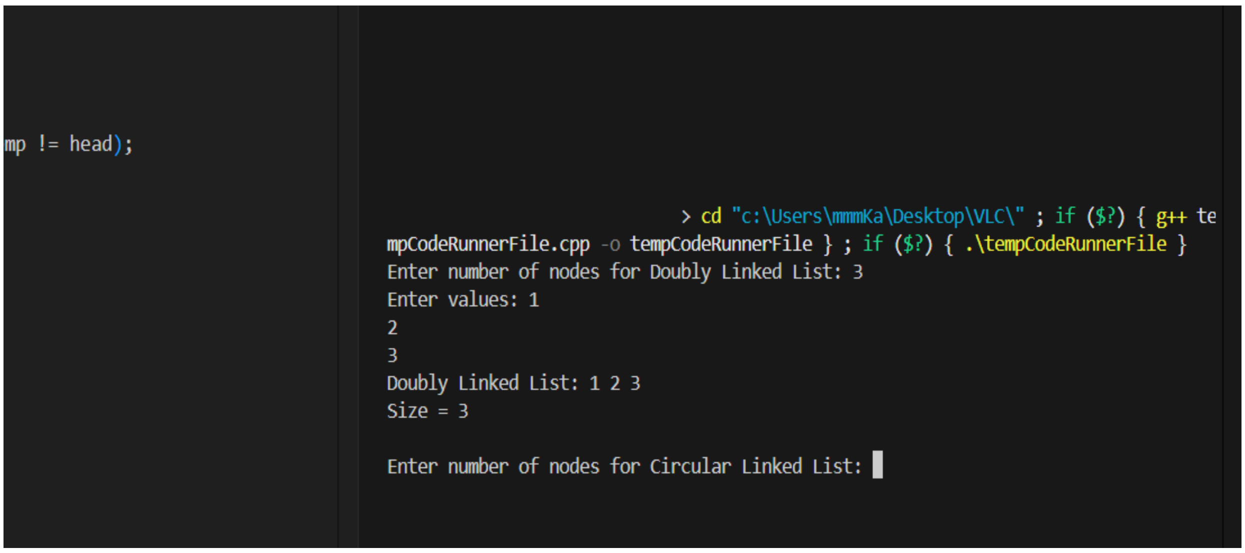
```cpp
int main() {
    CircularLinkedList cll;
    int n,val;
    cout << "Number of nodes: "; cin >> n;
    cout << "Enter values: ";
    for(int i=0;i<n;i++){ cin >> val; cll.insertLast(val); }
    cout << "Circular List: ";
    cll.display();
}
```



```
PS C:\Users\mmmKa\Desktop\VLC> cd "c:\Users\mmmKa\Desktop\VLC\" ; if ($?) { g++ tempCodeRunnerFile.cpp -o tempCodeRunnerFile } ; if ($?) { .\tempCodeRunnerFile }
Number of nodes: 5
Enter values: 1
2
3
4
5
Circular List: 1 2 3 4 5 1
PS C:\Users\mmmKa\Desktop\VLC>
```

Question 3

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

// Doubly Linked List Node
struct DNode {
    int data;
    DNode* prev;
    DNode* next;
};

// Doubly Linked List
class DoublyLinkedList {
private:
    DNode* head;
public:
    DoublyLinkedList() : head(nullptr) {}

    void insertLast(int val) {
        DNode* n = new DNode{val, nullptr, nullptr};
        if (!head) { head = n; return; }
        DNode* temp = head;
        while (temp->next) temp = temp->next;
        temp->next = n;
        n->prev = temp;
    }

    int size() {
        int count = 0;
        DNode* temp = head;
        while (temp) { count++; temp = temp->next; }
        return count;
    }

    void display() {
        DNode* temp = head;
        while (temp) { cout << temp->data << " "; temp = temp->next; }
        cout << endl;
    }
};

// Circular Linked List Node
struct CNode {
    int data;
    CNode* next;
};

// Circular Linked List
class CircularLinkedList {
private:
    CNode* head;
public:
    CircularLinkedList() : head(nullptr) {}

    void insertLast(int val) {
        CNode* n = new CNode{val, nullptr};
        if (!head) { head = n; n->next = head; return; }
        CNode* temp = head;
        while (temp->next != head) temp = temp->next;
        temp->next = n;
        n->next = head;
    }
    int size() {
        if (!head) return 0;
        int count = 0;
        CNode* temp = head;
        do { count++; temp = temp->next; } while (temp != head);
        return count;
    }

    void display() {
        if (!head) return;
        CNode* temp = head;
        do { cout << temp->data << " "; temp = temp->next; } while (temp != head);
        cout << endl;
    }
};

int main() {
    DoublyLinkedList dll;
    CircularLinkedList cll;
    int n, val;

    cout << "Enter number of nodes for Doubly Linked List: ";
    cin >> n;
    cout << "Enter values: ";
    for (int i = 0; i < n; i++) { cin >> val; dll.insertLast(val); }

    cout << "Doubly Linked List: "; dll.display();
    cout << "Size = " << dll.size() << endl;

    cout << "\nEnter number of nodes for Circular Linked List: ";
    cin >> n;

    cout << "Enter values: ";

    for (int i = 0; i < n; i++) { cin >> val; cll.insertLast(val); }


    cout << "Circular Linked List: "; cll.display();

    cout << "Size = " << cll.size() << endl;

}
```

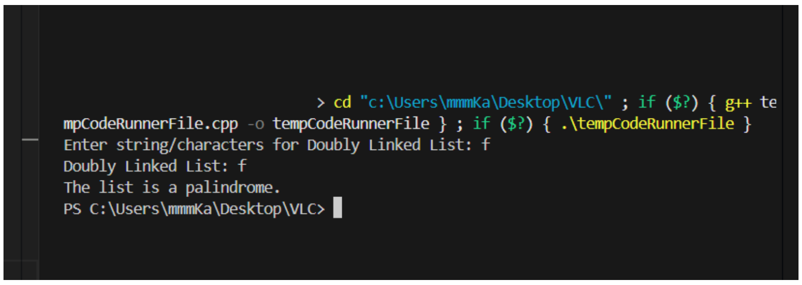


```
mp != head);

                                  > cd "c:\Users\mmmKa\Desktop\VLC\" ; if ($?) { g++ te
mpCodeRunnerFile.cpp -o tempCodeRunnerFile } ; if ($?) { .\tempCodeRunnerFile }
Enter number of nodes for Doubly Linked List: 3
Enter values: 1
2
3
Doubly Linked List: 1 2 3
Size = 3

Enter number of nodes for Circular Linked List: 
```

Question 4

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

struct DNode {
   char data;
   DNode* prev;
   DNode* next;
};

class DoublyLinkedList {
private:
   DNode* head;
public:
   DoublyLinkedList() : head(nullptr) {}

   void insertLast(char val) {
      DNode* n = new DNode{val, nullptr, nullptr};
      if (!head) { head = n; return; }
      DNode* temp = head;
      while (temp->next) temp = temp->next;
      temp->next = n;
      n->prev = temp;
   }

   bool isPalindrome() {
      if (!head) return true;
      DNode* left = head;
      DNode* right = head;
      while (right->next) right = right->next;
```

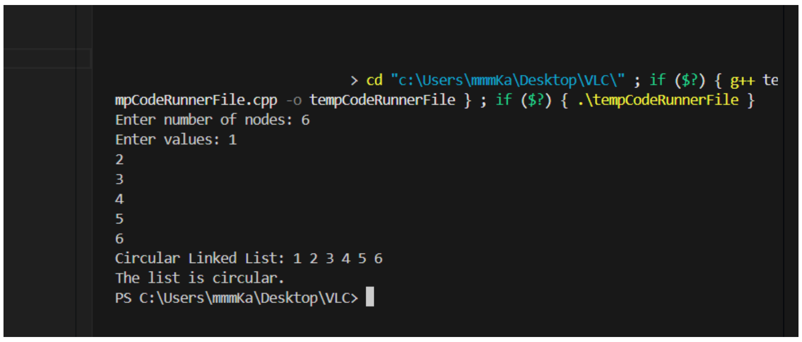
```
        while (left != right && right->next != left) {
            if (left->data != right->data) return false;
            left = left->next;
            right = right->prev;
        }
        return true;
    }

    void display() {
        DNode* temp = head;
        while (temp) { cout << temp->data << " "; temp = temp->next; }
        cout << endl;
    }
};

int main() {
    DoublyLinkedList dll;
    string s;
    cout << "Enter string/characters for Doubly Linked List: ";
    cin >> s;

    for (char c : s) dll.insertLast(c);

    cout << "Doubly Linked List: "; dll.display();
    if (dll.isPalindrome()) cout << "The list is a palindrome.\n";
    else cout << "The list is not a palindrome.\n";
}
```



```
                         > cd "c:\Users\mmmKa\Desktop\VLC\" ; if ($?) { g++ te
mpCodeRunnerFile.cpp -o tempCodeRunnerFile } ; if ($?) { .\tempCodeRunnerFile }
Enter string/characters for Doubly Linked List: f
Doubly Linked List: f
The list is a palindrome.
PS C:\Users\mmmKa\Desktop\VLC>
```

Question 5

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

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

class CircularLinkedList {
private:
    CNode* head;
public:
    CircularLinkedList() : head(nullptr) {}

    void insertLast(int val) {
        CNode* n = new CNode{val, nullptr};
        if (!head) { head = n; n->next = head; return; }
        CNode* temp = head;
        while (temp->next != head) temp = temp->next;
        temp->next = n;
        n->next = head;
    }

    void display() {
        if (!head) return;
        CNode* temp = head;
        do { cout << temp->data << " "; temp = temp->next; } while (temp != head);
        cout << endl;
    }
    bool isCircular() {

        if (!head) return false;

        CNode* temp = head->next;

        while (temp && temp != head) temp = temp->next;

        return temp == head;

    }

};


int main() {

    CircularLinkedList cll;

    int n, val;

    cout << "Enter number of nodes: ";

    cin >> n;


    for (int i = 0; i < n; i++) { cin >> val; cll.insertLast(val); }


    cout << "Circular Linked List: "; cll.display();


    if (cll.isCircular()) cout << "The list is circular.\n";

    else cout << "The list is not circular.\n";

}
```



```
                              > cd "c:\Users\mmmKa\Desktop\VLC\" ; if ($?) { g++ te
mpCodeRunnerFile.cpp -o tempCodeRunnerFile } ; if ($?) { .\tempCodeRunnerFile }
Enter number of nodes: 6
Enter values: 1
2
3
4
5
6
Circular Linked List: 1 2 3 4 5 6
The list is circular.
PS C:\Users\mmmKa\Desktop\VLC>
```