DATA STRUCTURES

Division	**CS(AIML) -A**
Batch	**2**
GR-no	**12311493**
Roll no	**54**
Name	**Atharva Kangralkar**

Assignment 6:

Write a program to implement SLL, DLL and CLL operations

Code:-

1)SLL:-

```
#include <iostream>
using namespace std;
struct SLLNode {
 int data;
 SLLNode* next;
 SLLNode(int val) : data(val), next(nullptr) {}
};
class SLL {
public:
 SLLNode* head;
 SLL() : head(nullptr) {}
 void insert(int val) {
 SLLNode* newNode = new SLLNode(val);
 if (!head) head = newNode;
 else {
 SLLNode* temp = head;
 while (temp->next) temp = temp->next;
 temp->next = newNode;
 }
 }
 void remove(int val) {
 if (!head) return;
 if (head->data == val) {
 SLLNode* temp = head;
 head = head->next;
 delete temp;
 return;
 }
```

```
 SLLNode* temp = head;
 while (temp->next && temp->next->data != val) temp = temp->next;
 if (temp->next) {
 SLLNode* delNode = temp->next;
 temp->next = delNode->next;
 delete delNode;
 }
 }
 void display() {
 SLLNode* temp = head;
 while (temp) {
 cout << temp->data << " -> ";
 temp = temp->next;
 }
 cout << "NULL\n";
 }
};
int main() {
 SLL sll;
 cout << "Singly Linked List:\n";
 sll.insert(1);
 sll.insert(2);
 sll.insert(3);
 sll.display();
 sll.remove(2);
 sll.display();
 return 0;
}
```

2)DLL:-

```
#include <iostream>
using namespace std;
struct DLLNode {
 int data;
 DLLNode* next;
 DLLNode* prev;
 DLLNode(int val) : data(val), next(nullptr), prev(nullptr) {}
};
class DLL {
public:
 DLLNode* head;
 DLL() : head(nullptr) {}
 void insert(int val) {
 DLLNode* newNode = new DLLNode(val);
 if (!head) head = newNode;
 else {
 DLLNode* temp = head;
 while (temp->next) temp = temp->next;
 temp->next = newNode;
 newNode->prev = temp;
 }
 }
 void remove(int val) {
 if (!head) return;
 if (head->data == val) {
 DLLNode* temp = head;
 head = head->next;
 if (head) head->prev = nullptr;
 delete temp;
```

```
return;
}
DLLNode* temp = head;
while (temp && temp->data != val) temp = temp->next;
if (temp) {
if (temp->prev) temp->prev->next = temp->next;
if (temp->next) temp->next->prev = temp->prev;
delete temp;
}
}
void display() {
DLLNode* temp = head;
while (temp) {
cout << temp->data << " <-> ";
temp = temp->next;
}
cout << "NULL\n";
}
};
int main() {
DLL dll;
cout << "Doubly Linked List:\n";
dll.insert(10);
dll.insert(20);
dll.insert(30);
dll.display();
dll.remove(20);
dll.display();
return 0;
```

```
}
```

3)CLL:-

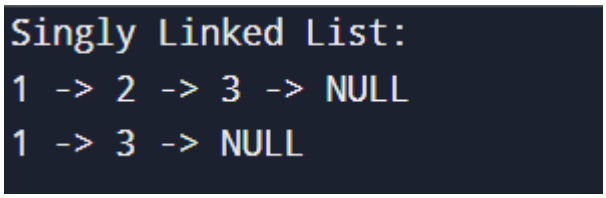
```
#include <iostream>
using namespace std;
struct CLLNode {
 int data;
 CLLNode* next;
 CLLNode(int val) : data(val), next(nullptr) {}
};
class CLL {
public:
 CLLNode* head;
 CLL() : head(nullptr) {}
 void insert(int val) {
 CLLNode* newNode = new CLLNode(val);
 if (!head) {
 head = newNode;
 head->next = head;
 } else {
 CLLNode* temp = head;
 while (temp->next != head) temp = temp->next;
 temp->next = newNode;
 newNode->next = head;
 }
 }
 void remove(int val) {
 if (!head) return;
 if (head->data == val && head->next == head) {
 delete head;
```

```
head = nullptr;
return;
}
CLLNode *temp = head, *prev = nullptr;
while (temp->next != head && temp->data != val) {
prev = temp;
temp = temp->next;
}
if (temp->data == val) {
if (prev) prev->next = temp->next;
if (temp == head) head = head->next;
delete temp;
}
}
void display() {
if (!head) return;
CLLNode* temp = head;
do {
cout << temp->data << " -> ";
temp = temp->next;
} while (temp != head);
cout << "(HEAD)\n";
}
};
int main() {
CLL cll;
cout << "Circular Linked List:\n";
cll.insert(100);
cll.insert(200);
```

```
cll.insert(300);
cll.display();
cll.remove(200);
cll.display();
return 0;
}
```

Output:-

1)SLL:-



2)DLL:-

3)CLL:-