```
/* DSL – EXPERIMENT 7 – C19 */

#include<iostream>
#include<string.h>
using namespace std;

class Member {
    public:
        int prn;
        char name[20];
        Member *next;
};

class Club {
    Member *root;
    Member *curr;
    Member *prev;

    public:
        Club() {
            root = NULL;
            curr = NULL;
            prev = NULL;
        }

        Member *GetRoot() {
            return (root);
        }

        void AddMembers() {

            while (true) {
                curr = new Member();
                curr->next = NULL;
                cout << "\nEnter PRN No: ";
                cin >> curr->prn;

                if(curr->prn == 0) break;

                cout << "\nEnter Member Name: ";
                cin >> curr->name;

                if (root == NULL) {
                    root = curr;
                    prev = curr;
                }
                else {
                    prev->next = curr;
                    prev = curr;
                }
            }
```

```
}

void DisplayMembers() {
    curr = root;

    while (curr ≠ NULL) {
        printf("(%d) %s ⟶ ", curr→prn, curr→name);
        curr = curr→next;
    }
    cout<<"NULL";
}

void CountMembers() {
    int c = 0;
    curr = root;

    while(curr ≠ NULL) {
        c++;
        curr = curr→next;
    }
    cout<<c;
}

void DisplayMembersReverse(Member *r) {
    if(r == NULL) return;
    DisplayMembersReverse(r→next);
    printf("(%d) %s ⟶ ", r→prn, r→name);
}

void DeleteMembers() {
    int prn;
    cout << "\nEnter PRN No. To Delete: ";
    cin >> prn;
    curr = root;
    prev = NULL;

    while (curr ≠ NULL) {
        if(prn == curr→prn) {

            // Root node
            if (curr == root){
                root = root→next;
                delete curr;
                break;
            }

            // Last node
            else if (curr→next==NULL) {
                prev→next = NULL;
                delete curr;
                break;
```

```
            }

            // Other nodes
            else {
                prev->next= curr->next;
                delete curr;
                break;
            }
        }
        prev = curr;
        curr = curr->next;
    }
}

Club operator + (Club c2) {
    Member *r1;
    Member *r2;
    Member *r3;
    Club c3;

    r1 = root;
    r2 = c2.root;

    while (r1≠NULL) {
        c3.curr = new Member();
        c3.curr->prn = r1->prn;
        strcpy(c3.curr->name,r1->name);
        c3.curr->next = NULL;
        if(c3.root == NULL) {
            c3.root = c3.curr;
            c3.prev = c3.curr;
        }
        else {
            c3.prev->next = c3.curr;
            c3.prev = c3.curr;
        }

        r1 = r1->next;
    }

    while (r2≠NULL) {
        c3.curr = new Member();
        c3.curr->prn = r2->prn;
        strcpy(c3.curr->name,r2->name);
        c3.curr->next = NULL;
        if(c3.root == NULL) {
            c3.root = c3.curr;
            c3.prev = c3.curr;
        }
        else {
            c3.prev->next = c3.curr;
```

```
                c3.prev = c3.curr;
            }
            r2 = r2->next;
        }

        return (c3);
    }
};

int main() {
    Club c1,c2,c3;
    int op = -1;

    while(op!=0) {
        cout << "\n\n1.Add Members";
        cout << "\n2.Display Members";
        cout << "\n3.Display Reverse";
        cout << "\n4.Count Members";
        cout << "\n5.Delete Members";
        cout << "\n6.Concatenate List Using Operator Overloading";
        cout << "\n0.Exit";
        cout << "\n\n*** Select Option *** -> ";
        cin >> op;

        switch(op) {
            case 1:
                cout << "\n\n*** Add To Club 1 ***\n";
                c1.AddMembers();
                cout << "\n\n*** Add To Club 2 ***\n";
                c2.AddMembers();
                break;

            case 2:
                cout << "\n\n*** Display Members - Club 1 ***\n\n";
                c1.DisplayMembers();
                cout<<"\n";
                cout << "\n*** Display Members - Club 2 ***\n\n";
                c2.DisplayMembers();
                cout<<"\n";
                break;

            case 3:
                cout << "\n\n*** Display Members Reverse - Club 1 ***\n\n";
                c1.DisplayMembersReverse(c1.GetRoot());
                cout << "\n\n*** Display Members Reverse - Club 2 ***\n\n";
                c2.DisplayMembersReverse(c2.GetRoot());
                cout<<"\n";
                break;

            case 4:
                cout<<"\n\nTotal Members Of Club 1 Are ";//<<c1.CountMembers();
```

```
            c1.CountMembers();
            cout<<"\n\nTotal Members Of Club 2 Are ";//<<c2.CountMembers();
            c2.CountMembers();
            cout<<"\n";
            break;

        case 5:
            cout << "\n\n*** Delete Members - Club 1 ***\n";
            c1.DeleteMembers();
            cout << "\n\n*** Delete Members - Club 2 ***\n";
            c2.DeleteMembers();
            cout<<"\n";
            break;

        case 6:
            cout << "\n\n*** Concatenating - Club 1 & Club 2***\n";
            c3 = c1 + c2;
            cout << "\n*** Display Members - Club 3 ***\n\n";
            c3.DisplayMembers();
            cout<<"\n";
            break;

        case 0:
            break;
    }
  }

  return 0;
}
```

```
/*

──────────────────── OUTPUT ──────────────────────


1.Add Members
2.Display Members
3.Display Reverse
4.Count Members
5.Delete Members
6.Concatenate List Using Operator Overloading
0.Exit

*** Select Option *** → 1


*** Add To Club 1 ***

Enter PRN No: 123

Enter Member Name: Pratik

Enter PRN No: 456

Enter Member Name: Alex

Enter PRN No: 789

Enter Member Name: Jojo

Enter PRN No: 0


*** Add To Club 2 ***

Enter PRN No: 147

Enter Member Name: Shama

Enter PRN No: 258

Enter Member Name: Riya

Enter PRN No: 369

Enter Member Name: Yuki

Enter PRN No: 0
```

```
1.Add Members
2.Display Members
3.Display Reverse
4.Count Members
5.Delete Members
6.Concatenate List Using Operator Overloading
0.Exit

** Select Option ** → 2


** Display Members - Club 1 **

(123) Pratik ⟶ (456) Alex ⟶ (789) Jojo ⟶ NULL

** Display Members - Club 2 **

(147) Shama ⟶ (258) Riya ⟶ (369) Yuki ⟶ NULL


1.Add Members
2.Display Members
3.Display Reverse
4.Count Members
5.Delete Members
6.Concatenate List Using Operator Overloading
0.Exit

** Select Option ** → 3


** Display Members Reverse - Club 1 **

(789) Jojo ⟶ (456) Alex ⟶ (123) Pratik ⟶

** Display Members Reverse - Club 2 **

(369) Yuki ⟶ (258) Riya ⟶ (147) Shama ⟶


1.Add Members
2.Display Members
3.Display Reverse
4.Count Members
5.Delete Members
6.Concatenate List Using Operator Overloading
0.Exit

** Select Option ** → 4
```

```
Total Members Of Club 1 Are 3

Total Members Of Club 2 Are 3


1.Add Members
2.Display Members
3.Display Reverse
4.Count Members
5.Delete Members
6.Concatenate List Using Operator Overloading
0.Exit

*** Select Option *** → 5


*** Delete Members - Club 1 ***

Enter PRN No. To Delete: 456


*** Delete Members - Club 2 ***

Enter PRN No. To Delete: 369


1.Add Members
2.Display Members
3.Display Reverse
4.Count Members
5.Delete Members
6.Concatenate List Using Operator Overloading
0.Exit

*** Select Option *** → 2


*** Display Members - Club 1 ***

(123) Pratik ⟶ (789) Jojo ⟶ NULL

*** Display Members - Club 2 ***

(147) Shama ⟶ (258) Riya ⟶ NULL


1.Add Members
2.Display Members
3.Display Reverse
4.Count Members
5.Delete Members
```

```
6.Concatenate List Using Operator Overloading
0.Exit

** Select Option ** → 4


Total Members Of Club 1 Are 2

Total Members Of Club 2 Are 2


1.Add Members
2.Display Members
3.Display Reverse
4.Count Members
5.Delete Members
6.Concatenate List Using Operator Overloading
0.Exit

** Select Option ** → 6


** Concatenating - Club 1 & Club 2**

** Display Members - Club 3 **

(123) Pratik ⟶ (789) Jojo ⟶ (147) Shama ⟶ (258) Riya ⟶ NULL


1.Add Members
2.Display Members
3.Display Reverse
4.Count Members
5.Delete Members
6.Concatenate List Using Operator Overloading
0.Exit

** Select Option ** → 0

*/
```