

```

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

class SLL;
class dnode
{
    int div;
    int prn;
    char name[20];
    dnode *next;

    friend SLL;

public:
    dnode()
    {
        next = NULL;
        div = prn = 0;
    }
    dnode(int d, int p, char Name[])
    {
        div = d;
        prn = p;
        next = NULL;
        strcpy(name, Name);
    }
};

class SLL
{
public:
    dnode *head;
    dnode *insert;
    dnode *end;

    void create();
    void print();
    void insertMember();
    void deleteMember();
    void count();
    void mergeList();
    void recursionPrint(dnode *temp);

    SLL()
    {
        head = NULL;
        insert = NULL;
        end = NULL;
    }
} list1;

void SLL::create()
{
    int n, d, p;
    char Name[20];

    cout << "Enter the number of students: ";

```

```

cin >> n;

cout << "\nEnter the name, division and PRN of President: \n";
cin >> Name >> d >> p;

head = new dnode(d, p, Name);
end = head;

cout << "\nEnter the name, division and PRN of members: \n";
for (int i = 1; i < n - 1; i++)
{
    cin >> Name >> d >> p;
    end->next = new dnode(d, p, Name);
    end = end->next;
}
insert = end;

cout << "\nEnter the name, division and PRN of SECRETARY: \n";
cin >> Name >> d >> p;
end->next = new dnode(d, p, Name);
end = end->next;
end->next = NULL;
}

void SLL::insertMember()
{
    int d, p;
    char Name[20];
    cout << "\nEnter the name, division and PRN of member to INSERT: \n";
    cin >> Name >> d >> p;
    insert->next = new dnode(d, p, Name);
    insert = insert->next;
    insert->next = end;
}

void SLL::deleteMember()
{
    int searchPRN;
    dnode *remove;
    cout << "Enter the PRN of member to delete: ";
    cin >> searchPRN;
    if (searchPRN == head->prn)
    {
        remove = head;
        head = head->next;
        delete remove;
    }
    else
    {
        for (dnode *temp = head; temp != insert; temp = temp->next)
        {
            if (temp->next->prn == searchPRN)
            {
                if (temp->next == insert)
                    insert = temp;
                remove = temp->next;
                temp->next = temp->next->next;
                delete remove;
            }
        }
    }
}

```

```

        if (temp == insert)
            break;
    }
}
if (searchPRN == end->prn)
{
    remove = end;
    insert->next = NULL;
    end = insert;
    delete remove;
}

void SLL::count()
{
    int count = 0;
    for (dnode *temp = head; temp != NULL; temp = temp->next)
        count++;
    cout << "\nTotal no. of students : " << count;
}

void SLL::print()
{
    cout << "\n\nNAME\tDIV\tPRN\n";
    for (dnode *temp = head; temp != NULL; temp = temp->next)
        cout << temp->name << "\t" << temp->div << "\t" << temp->prn <<
"\n";
    cout << "\nPresident is: " << head->name;
    cout << "\nSecretary is: " << end->name << "\n";
}

void SLL::mergeList()
{
    SLL list2;
    cout << "\nEnter the contents for second list: \n";
    list2.create();
    list1.end->next = list2.head;
    list1.end = list2.end;
}

void SLL::recursionPrint(dnode *temp)
{
    if (temp == NULL)
        return;
    else
        recursionPrint(temp->next);
    cout << temp->name << "\t" << temp->div << "\t" << temp->prn << "\n";
}

int main()
{
    int choice;
    do
    {
        cout << "\n1. Create List. \n2. Insert Member \n3. Delete Member
\n4. Print List \n5. Merge List \n6. Reverse List \n7. Count \n8. Exit";

```

```

    cout << "\n\nEnter your choice: ";
    cin >> choice;
    switch (choice)
    {
    case 1:
        list1.create();
        break;

    case 2:
        list1.insertMember();
        break;

    case 3:
        list1.deleteMember();
        break;

    case 4:
        list1.print();
        break;

    case 5:
        list1.mergeList();
        break;

    case 6:
        cout << "\n\nNAME\tDIV\tPRN\n";
        list1.recursionPrint(list1.head);
        break;

    case 7:
        list1.count();
        break;

    case 8:
        return 0;

    default:
        cout << "Invalid Choice !";
        break;
    }
} while (choice != 0);
return 0;
}

```