**Modern Education Society’s**

|  |  |
| --- | --- |
| **NAME OF STUDENT:** Prathamesh Kalyan Sable | **CLASS:** SE Comp 1 |
| **SEMESTER/YEAR:** Sem-3 / 2022-23 | **ROLL NO:** 015 |
| **DATE OF PERFORMANCE:**  / /2022 | **DATE OF SUBMISSION:** / /2022 |
| **EXAMINED BY:** Prof. N.R. Mhaske | **EXPERIMENT NO: DSL C-19** |

**College of Engineering, Pune**

**TITLE : TO PERFORM VARIOUS OPERATIONS ON LINK LIST .**

**PROBLEM STATEMENT :**Department of Computer Engineering has student's club named 'pinnacle club' . Students of second ,third and final year of department can be granted membership on request . Similarly on may cancel the membership of club .First node is reserved for the president of the club and last node is reserved for the secretary of club .write C++ program to maintain club members information using singly linked list .store student PRN and Name .Write function to:

a) Add and Delete the members as well as the president or even secretary .

b)Compute total numbers of members of club .

c)Display members .

d)Display list in reverse order using recursion .

e)Two linked lists exists for two divisions .Concatinate two lists .

**OBJECTIVES:**

1. To understand structure of singly linked list .

2. To understand how to Create,Display and perform various operations on singly linked list .

**OUTCOMES:**

1.To analyze the problems to apply suitable algorithm and data structure .

2.To discriminate the usage of various data structures in approaching the problem solution .

3.To understand concepts of Linear Data Structure (singly linked lists ) .

**PRE-REQUISITES:**

1.Knowledge of C++ programming .

2.Knowledge of singly linked lists .

**APPARATUS:**

Working Computer with g++ compiler installed.

**QUESTIONS:**

1. What is a linked list ? State it's types .

2. Write applications of SLL.

**SOURCE CODE:**

#include <iostream>

using namespace std;

// node defination

class node

{

public:

    string prn;

    string name;

    node \*next;

    // constructors

    node()

    {

        prn = "";

        name = "";

        next = NULL;

    }

    node(string prn, string name)

    {

        this->prn = prn;

        this->name = name;

        this->next = NULL;

    }

};

// predeclearation of function count

int member\_count(node \*);

// function to display data

void display\_list(node \*head)

{

    node \*temp = head;

    if (member\_count(head) >= 1)

    {

        int count = 1;

        while (temp != NULL)

        {

            cout << count << ". PRN: " << temp->prn;

            cout << " NAME: " << temp->name << endl;

            temp = temp->next;

            count++;

        }

    }

    else

    {

        cout << "Empty List" << endl;

    }

}

// fucntion to add president to list

void add\_president(node \*&head, string prn, string name)

{

    node \*new\_president = new node(prn, name);

    new\_president->next = head;

    head = new\_president;

}

// fucntion to add secretory to list

void add\_secretary(node \*head, string prn, string name)

{

    node \*new\_secretary = new node(prn, name);

    node \*temp = head;

    while (temp->next != NULL)

    {

        temp = temp->next;

    }

    temp->next = new\_secretary;

}

// fucntion to add member to list

void add\_member(node \*head, string prn, string name)

{

    node \*new\_member = new node(prn, name);

    node \*temp = head;

    while (temp->next->next != NULL)

    {

        temp = temp->next;

    }

    new\_member->next = temp->next;

    temp->next = new\_member;

}

// fucntion to count number of members

int member\_count(node \*head)

{

    node \*temp = head;

    // temp = temp->next;

    int count = 0;

    while (temp != NULL)

    {

        temp = temp->next;

        count++;

    }

    return count;

}

// recursive function to display list in reverse order

void display\_reverse(node \*cur)

{

    if (cur != NULL)

    {

        display\_reverse(cur->next);

        cout << "PRN : " << cur->prn;

        cout << " NAME : " << cur->name << endl;

    }

}

// function to concatinate two lists ie 2 to 1 and 2 will be empty

void concatinate(node \*head1, node \*&head2)

{

    node \*temp = new node;

    temp = head1;

    while (temp->next != NULL)

    {

        temp = temp->next;

    }

    temp->next = head2;

    head2 = NULL;

}

// function to delete president

void delete\_president(node \*&head)

{

    node \*tofree = head;

    head = head->next;

    delete tofree;

}

// function to delete secretory

void delete\_secretory(node \*head)

{

    node \*temp = new node;

    temp = head;

    while (temp->next->next != NULL)

    {

        temp = temp->next;

    }

    node \*tofree = temp->next;

    temp->next = NULL;

    delete tofree;

}

// function to delete member

void delete\_member(node \*head, string prn)

{

    node \*temp = new node;

    temp = head;

    while (temp->next != NULL)

    {

        if (temp->next->prn == prn)

        {

            node \*tofree = temp->next;

            temp->next = temp->next->next;

            delete tofree;

        }

        temp = temp->next;

    }

}

// function to swap current list

void swap(node \*&l1, node \*&l2)

{

    node \*temp = l1;

    l1 = l2;

    l2 = temp;

}

// fucntion to initialize the list with president and secretory

node \*start(char cur)

{

    string prn, name;

    cout << "Enter Details for Class " << cur << endl;

    cout << "Enter the PRN of President:";

    cin >> prn;

    cout << "Enter the Name of President:";

    cin.ignore();

    getline(cin, name);

    node \*list = new node(prn, name);

    cout << "Enter the PRN of Secretory:";

    cin >> prn;

    cout << "Enter the Name of Secretory:";

    cin.ignore();

    getline(cin, name);

    add\_secretary(list, prn, name);

    return list;

}

// main function run

int main()

{

    // variables for run control

    char current = 'A';

    string prn, name;

    int choice, choice2;

    bool while\_var = true;

    // creating lists

    node \*list1 = start('A');

    node \*list2 = start('B');

    // loop

    while (while\_var)

    {

        cout << "\n----------------------------------" << endl;

        cout << "\tPINNACLE CLUB" << endl;

        cout << " - Current list is Class " << current << endl;

        cout << "1. Display Current List" << endl;

        cout << "2. Add Member" << endl;

        cout << "3. Delete Member" << endl;

        cout << "4. Count Member" << endl;

        cout << "5. Display list in Reverse Order" << endl;

        cout << "6. Switch List" << endl;

        cout << "7. Merge two Lists" << endl;

        cout << "8. Exit" << endl;

        cout << "Enter your choice :";

        cin >> choice;

        switch (choice)

        {

        case 1:

            cout << "\nThe Members of Class " << current << " are as: " << endl;

            display\_list(list1);

            break;

        case 2:

            cout << "\n1. Add President" << endl;

            cout << "2. Add Member" << endl;

            cout << "3. Add Secretory" << endl;

            cout << "Enter your choice:";

            cin >> choice2;

            switch (choice2)

            {

            case 1:

                cout << "Enter the PRN of President:";

                cin >> prn;

                cout << "Enter the Name of President:";

                cin.ignore();

                getline(cin, name);

                add\_president(list1, prn, name);

                cout << "President added sucessfully to Class " << current << endl;

                break;

            case 2:

                cout << "Enter the PRN:";

                cin >> prn;

                cout << "Enter the Name :";

                cin.ignore();

                getline(cin, name);

                add\_member(list1, prn, name);

                cout << "Member added sucessfully to Class " << current << endl;

                break;

            case 3:

                cout << "Enter the PRN of Secretory:";

                cin >> prn;

                cout << "Enter the Name of Secretory:";

                cin.ignore();

                getline(cin, name);

                add\_secretary(list1, prn, name);

                cout << "Secretory added sucessfully to Class " << current << endl;

                break;

            default:

                cout << "Enter a valid choice" << endl;

                break;

            }

            break;

        case 3:

            cout << "\n1. Delete President" << endl;

            cout << "2. Delete Member" << endl;

            cout << "3. Delete Secretory" << endl;

            cout << "Enter your choice:";

            cin >> choice2;

            switch (choice2)

            {

            case 1:

                delete\_president(list1);

                cout << "President deleted sucessfully from Class " << current << endl;

                break;

            case 2:

                cout << "Enter PRN to delete:";

                cin >> prn;

                delete\_member(list1, prn);

                cout << "Member deleted sucessfully from Class " << current << endl;

                break;

            case 3:

                delete\_secretory(list1);

                cout << "Secretory deleted sucessfully from Class " << current << endl;

                break;

            default:

                cout << "Enter a valid choice" << endl;

                break;

            }

            break;

        case 4:

            cout << "\nMembers in list " << current << " are " << member\_count(list1) << endl;

            break;

        case 5:

            cout << "\nThe Members of Class " << current << " in reverse order are as: " << endl;

            display\_reverse(list1);

            break;

        case 6:

            swap(list1, list2);

            if (current == 'A')

                current = 'B';

            else

                current = 'A';

            cout << "\nThe current list is changed sucessfully to class " << current << endl;

            break;

        case 7:

            concatinate(list1, list2);

            cout << "\nThe Members added to Class " << current << " Sucessfully " << endl;

            delete list2;

            list2 = start('B');

            break;

        case 8:

            cout << "\nThank you. " << endl;

            while\_var = false;

            break;

        default:

            cout << "Enter a valid choice..." << endl;

            break;

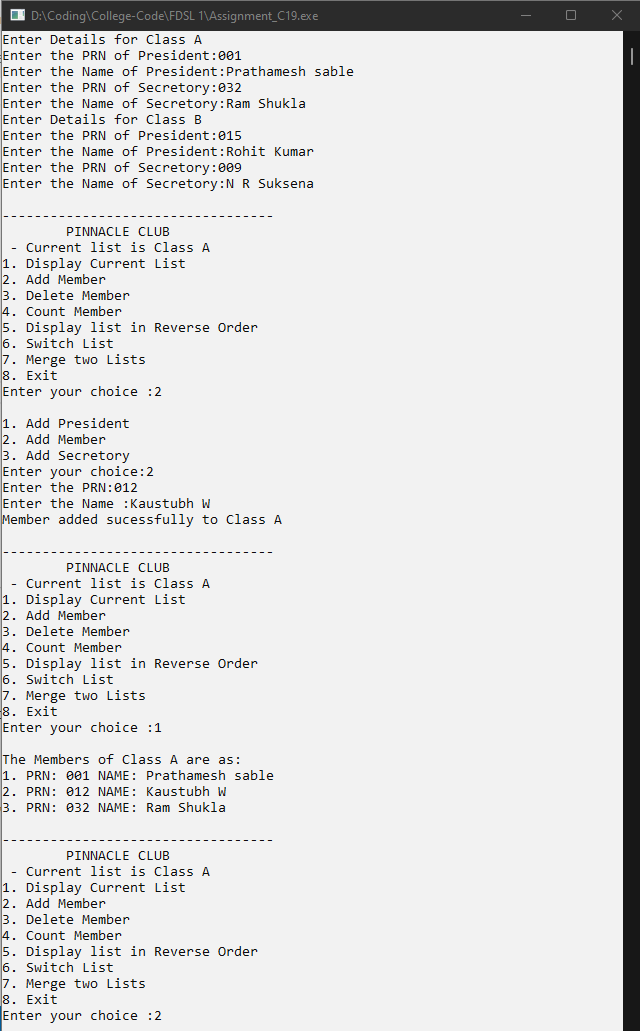
        }

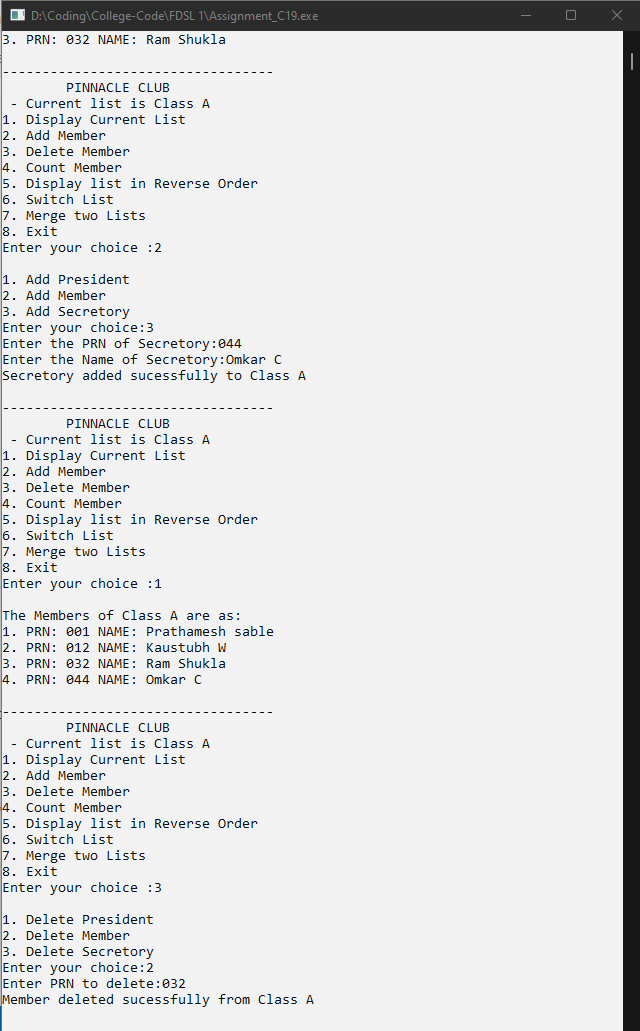
    }

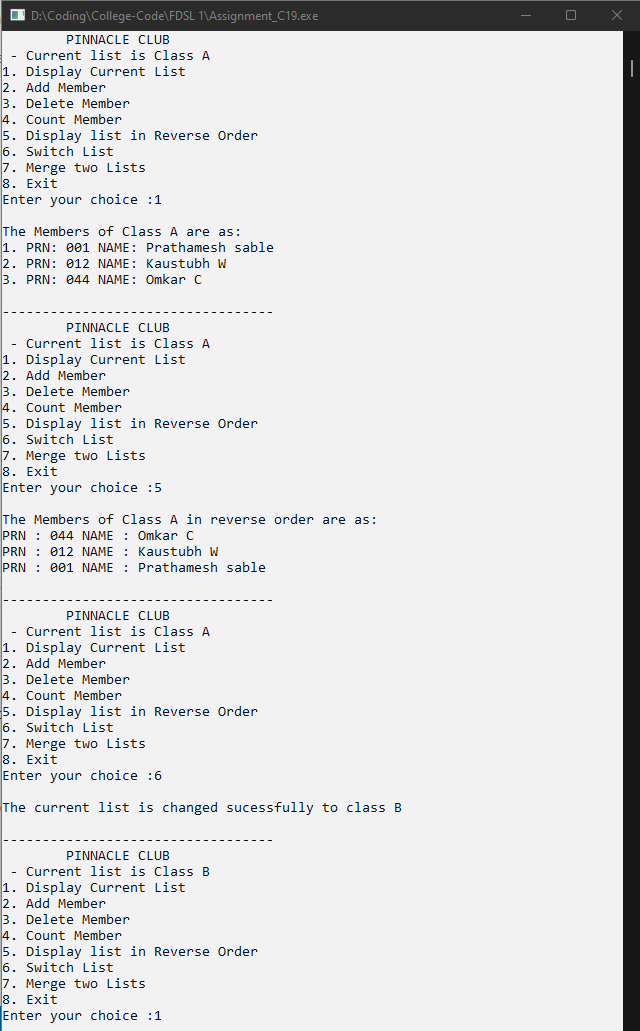
    return 0;

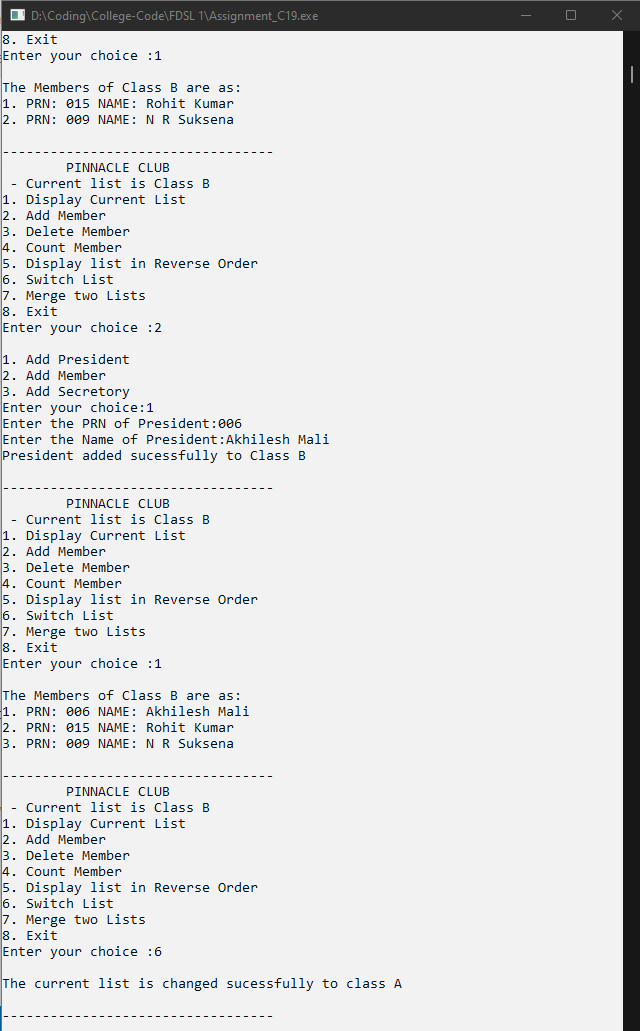
}

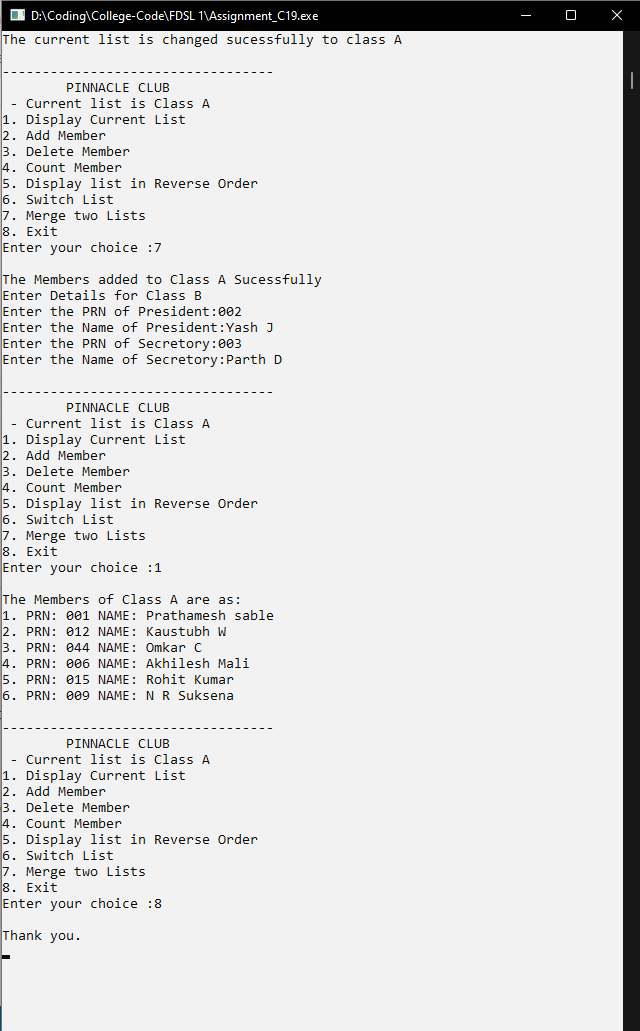
**OUTPUT:**

****

****

****

****

****