#include <iostream>

using namespace std;

struct Node {

    int data;

    Node\* next;

};

class LinkedList {

public:

    Node\* head;

    LinkedList() {

        head = nullptr;

    }

    ~LinkedList() {

        Node\* current = head;

        while (current != nullptr) {

            Node\* nextNode = current->next;

            delete current;

            current = nextNode;

        }

    }

    void insert(int val) {

        Node\* newNode = new Node();

        newNode->data = val;

        newNode->next = head;

        head = newNode;

    }

    bool search(int val) {

        Node\* temp = head;

        while (temp != nullptr) {

            if (temp->data == val)

                return true;

            temp = temp->next;

        }

        return false;

    }

    void display() {

        Node\* temp = head;

        while (temp != nullptr) {

            cout << temp->data << " ";

            temp = temp->next;

        }

        cout << endl;

    }

    void intersection(LinkedList& set2) {

        Node\* temp = head;

        cout << "Set of students who like both vanilla and butterscotch: ";

        while (temp != nullptr) {

            if (set2.search(temp->data)) {

                cout << temp->data << " ";

            }

            temp = temp->next;

        }

        cout << endl;

    }

    void unionSet(LinkedList& set2) {

        Node\* temp = head;

        cout << "Set of students who like either vanilla or butterscotch but not both: ";

        while (temp != nullptr) {

            if (!set2.search(temp->data)) {

                cout << temp->data << " ";

            }

            temp = temp->next;

        }

        temp = set2.head;

        while (temp != nullptr) {

            if (!search(temp->data)) {

                cout << temp->data << " ";

            }

            temp = temp->next;

        }

        cout << endl;

    }

};

int main() {

    LinkedList vanilla, butterscotch;

    int totalStudents, vanillaCount, butterscotchCount, neitherCount = 0, student;

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

    cin >> totalStudents;

    cout << "Enter number of students who like vanilla: ";

    cin >> vanillaCount;

    cout << "Enter the roll numbers of students who like vanilla: ";

    for (int i = 0; i < vanillaCount; i++) {

        cin >> student;

        vanilla.insert(student);

    }

    cout << "Enter number of students who like butterscotch: ";

    cin >> butterscotchCount;

    cout << "Enter the roll numbers of students who like butterscotch: ";

    for (int i = 0; i < butterscotchCount; i++) {

        cin >> student;

        butterscotch.insert(student);

    }

    cout << "\nVanilla set: ";

    vanilla.display();

    cout << "Butterscotch set: ";

    butterscotch.display();

    vanilla.intersection(butterscotch);

    vanilla.unionSet(butterscotch);

    cout << "Set of students who like neither vanilla nor butterscotch: ";

    for (int i = 1; i <= totalStudents; i++) {

        if (!vanilla.search(i) && !butterscotch.search(i)) {

            cout << i << " ";

            neitherCount++;

         }

    }

    cout << endl

         << "Number of students who like neither vanilla nor butterscotch: "

         << neitherCount

         << endl;

    return 0;

}

OUTPUT:-

Enter total number of students: 5

Enter number of students who like vanilla: 2

Enter the roll numbers of students who like vanilla: 2

5

Enter number of students who like butterscotch: 4

Enter the roll numbers of students who like butterscotch: 2

1

3

5

Vanilla set: 5 2

Butterscotch set: 5 3 1 2

Set of students who like both vanilla and butterscotch: 5 2

Set of students who like either vanilla or butterscotch but not both: 3 1

Set of students who like neither vanilla nor butterscotch: 4

Number of students who like neither vanilla nor butterscotch: 1