```
//============================================================================
// Name        : 21465_Pract8.cpp
// Author      : Chaitanya Paraskar
// Roll No.    : 21465
// Aim         : Second year Computer Engineering class, set A of students like
Vanilla Icecream and set B
//               of students like butterscotch ice-cream. Write C++ program to
store two sets using linked list.
//               compute and display - a) Set of students who like both vanilla
and butterscotch
//                                     b) Set of students who like either
vanilla or butterscotch or not both
//                                     c) Number of students who like neither
vanilla nor butterscotch
//============================================================================

#include "iostream"
using namespace std;

class Student
{
public:
    int rno;
    Student *next;
    Student(int rno)
    {
        this->rno = rno;
    }
};

class Set
{
    Student *head;
    int count = 0;

    void add(int rno)
    {
        Student *s = new Student(rno);
        s->next = this->head;
        head = s;

        count++;
    }

    friend Set *initUniversal();

public:
    Set() {}
    Set(Set *R)
    {
        int n;

        cout << "No. of Students : ";
        cin >> n;
```

```
    int i = 0;

    while (i < n)
    {
        int rno;

        cout << "Enter Roll No. of " << i + 1 << "th student : ";
        cin >> rno;

        if (this->not_in(rno) && R->in(rno))
        {
            this->add(rno);
            i++;
        }
        else
        {
            cout << "Already Present in Set !!" << endl;
        }
    }
}

void display()
{
    cout << "Set : [";

    Student *ptr = this->head;

    while (ptr != NULL)
    {
        cout << ptr->rno;

        if (ptr->next != NULL)
            cout << ", ";

        ptr = ptr->next;
    }

    cout << "]" << endl;
}

bool in(int n)
{
    Student *ptr = this->head;

    while (ptr != NULL)
    {
        if (ptr->rno == n)
            return true;
        ptr = ptr->next;
    }

    return false;
}
```

```
    bool not_in(int n)
    {
        Student *ptr = this->head;

        while (ptr != NULL)
        {
            if (ptr->rno == n)
                return false;
            ptr = ptr->next;
        }

        return true;
    }

    int getCount()
    {
        return this->count;
    }

    static Set *And(Set *A, Set *B);
    static Set *Or(Set *A, Set *B);
    static Set *Difference(Set *A, Set *B);
};

Set *Set::And(Set *A, Set *B)
{
    Set *R = new Set();

    Student *a = A->head;

    while (a != NULL)
    {
        if (B->in(a->rno))
            R->add(a->rno);

        a = a->next;
    }

    return R;
}

Set *Set::Or(Set *A, Set *B)
{
    Set *R = new Set();

    Student *a = A->head;

    while (a != NULL)
    {
        if (R->not_in(a->rno))
            R->add(a->rno);
        a = a->next;
    }
```

```
    Student *b = B->head;

    while (b != NULL)
    {
        if (R->not_in(b->rno))
            R->add(b->rno);
        b = b->next;
    }

    return R;
}

Set *Set::Difference(Set *A, Set *B)
{
    Set *R = new Set();

    Student *a = A->head;

    while (a != NULL)
    {
        if (B->not_in(a->rno))
            R->add(a->rno);
        a = a->next;
    }

    return R;
}

Set *initUniversal()
{

    cout << "Enter Universal Set : " << endl;
    Set *U = new Set();
    int n;

    cout << "No. of Students : ";
    cin >> n;

    int i = 0;

    while (i < n)
    {
        int rno;

        cout << "Enter Roll No. of " << i + 1 << "th student : ";
        cin >> rno;

        if (U->not_in(rno))
        {
            U->add(rno);
            i++;
        }
        else
```

```
        {
            cout << "Already Present in Set !!" << endl;
        }
    }

    return U;
}

int main()
{

    Set *U = initUniversal();

    cout << "Enter Set of Roll No. of Students who like Vanilla : " << endl;
    Set *Vanilla = new Set(U);
    cout << "Enter Set of Roll No. of Students who like Butter Scotch : " <<
endl;
    Set *ButterScotch = new Set(U);

    cout << "Roll No. of Students who Like Vanilla : " << endl;
    Vanilla->display();

    cout << "Roll No. of Students who Like Butter Scotch : " << endl;
    ButterScotch->display();

    cout << "Students who like both Vanilla and Butter Scotch : " << endl;
    Set *a = Set::And(Vanilla, ButterScotch);
    a->display();

    cout << "Students who like Either Vanilla or Butter Scotch but not both : "
<< endl;
    Set *x = Set::Or(Vanilla, ButterScotch);
    Set *y = Set::And(Vanilla, ButterScotch);
    Set *b = Set::Difference(x, y);
    b->display();

    Set *c = Set::Difference(U, x);
    int count = c->getCount();
    cout << "No. of Students who like neither Vanilla, nor Butter Scotch : " <<
count << endl;

    return 0;
}

/*

Output:

$ g++ Pract8.cpp -o out && ./out
Enter Universal Set :
No. of Students : 4
Enter Roll No. of 1th student : 111
Enter Roll No. of 2th student : 222
Enter Roll No. of 3th student : 333
```

```
Enter Roll No. of 4th student : 444
Enter Set of Roll No. of Students who like Vanilla :
No. of Students : 2
Enter Roll No. of 1th student : 111
Enter Roll No. of 2th student : 222
Enter Set of Roll No. of Students who like Butter Scotch :
No. of Students : 2
Enter Roll No. of 1th student : 111
Enter Roll No. of 2th student : 333
Roll No. of Students who Like Vanilla :
Set : [222, 111]
Roll No. of Students who Like Butter Scotch :
Set : [333, 111]
Students who like both Vanilla and Butter Scotch :
Set : [111]
Students who like Either Vanilla or Butter Scotch but not both :
Set : [222, 333]
No. of Students who like neither Vanilla, nor Butter Scotch : 1

*/
```