นาย อรุชา คงสุวรรณ 6330580021

Day Cover

Code:

```cpp
#include <iostream>
#include <vector>
#include <algorithm>
#include <limits>

using namespace std;

bool fully(vector<bool> days)
{
    bool full = true;
    for (auto a : days)
    {
        if (!a)
            full = false;
    }
    return full;
}

int number_of_student(vector<bool> students)
{
    int n = 0;
    for (int i = 0; i < students.size(); i++)
    {
        if (students[i])
            n++;
    }
    return n;
}

int day_cover(
    vector<vector<int>> A,
    vector<bool> &student_used,
    vector<bool> &day_used,
    int pos)
{
    if (fully(day_used))
    {
        return number_of_student(student_used);
    }
    if (pos >= student_used.size())
    {
        return INT32_MAX;
    }
    else
    {
        int ans1 = day_cover(A, student_used, day_used, pos + 1);

        student_used[pos] = true;
        for (int i = 0; i < A[pos].size(); i++)
            day_used[A[pos][i] - 1] = true;
        int ans2 = day_cover(A, student_used, day_used, pos + 1);
        return min(ans1, ans2);
    }
}

int main()
{
    int n, m;
    cin >> n;
    cin >> m;

    vector<vector<int>> A;

    for (int i = 0; i < m; i++)
    {
        int days_each_student;
        cin >> days_each_student;
        vector<int> vector_days;
        // vector_days.push_back(days_each_student);
        int day_each_student;

        for (int j = 0; j < days_each_student; j++)
        {
            cin >> day_each_student;
            vector_days.push_back(day_each_student);
        }
        A.push_back(vector_days);
    }

    vector<bool> day_used(n);
    vector<bool> student_used(m);

    int min_student = day_cover(A, student_used, day_used, 0);
    cout << "min student : " << min_student << endl;
}
```

นาย อรุชา คงสุวรรณ 6330580021

Explain:

1. รับ Input: n, mตามลำดับ และวันที่ว่างของนักเรียนแต่ละคน

   Ex 2 3

   1 1

   1 2

   2 1 2

2. นำวันที่นิสิตแต่ละคนสามารถมาจัดงานได้ เพิ่มไปใน vector<vector<int>> A เพิ่มนำไปใช้

   Ex {{1}, {2}, {1, 2}}

3. สร้าง vector<bool> ขนาด n และ m ชื่อว่า day_used, student_used เพื่อเช็คว่าวันไหนมีคนทำแล้ว และมีนิสิตคนไหนที่ถูกเลือกมาจัดงาน
4. bool fully(vector<bool> days): เพื่อเช็ค day_used ว่า วันที่จัดงาน มีนิสิตมาจัดงานครบ หรือยัง ถ้าครบแล้วจะ return true ถ้ายังจะ return false
5. int number_of_student(vector<bool> students): เพื่อเช็คจำนวนนิสิตที่ถูกเรียกใช้งานว่า ถูกลือกไปกี่คน
6. int day_cover(vector<vector<int>> A, vector<bool> &student_used, vector<bool> &day_used, int pos): เพื่อ return จำนวนนิสิตที่น้อยสุด ในขณะที่มีคนทำงานครบทุกวัน
7. โดยการใช้วิธี recursive โดย pos คือ นักเรียนคนที่เราจะเลือก หรือไม่เลือก โดยเริ่มจาก 0 และ คนสุดท้าย คือคนที่ m-1
8. เราจะทำการ recursive โดย ไม่เลือกนิสิตคนที่ pos คือ ans1 และ เลือกนิสิตคนที่ pos คือ ans2
9. การเลือก นิสิตคนที่ pos จะต้องแก้ student_used[pos] เป็น true และ day_used วันที่ นิสิตคนที่ pos สามารถมาทำงานได้ ซึ่งอยู่ใน code บรรทัดที่ 48 – 51
10. ทำการเลือก คำตอบที่น้อยที่สุด ระหว่าง ans1 และ ans2
11. ถ้าทำการ recursive ไปจนสุด จะมี 2 กรณี คือ
    a. วันที่ทำงานครบแล้ว (day_used) ก็จะไม่ทำการ recursive ต่อ และทำการ return number_of_student(student_used) หรือ จำนวนนักนิสิตที่ถูกเลือก
    b. pos เกิน m - 1 เนื่องจาก ถ้า pos มีขนาดมากกว่า m – 1 มากกว่าจำนวนนิสิตทั้งหมด เพราะนิสิตมีจำนวนตั้งแต่ 0 จนถึง m – 1 ซึ่งจะ return INT32_MAX เนื่องจาก ถ้า pos เกิน day_used จะไม่ครบอยู่แล้ว เราจึง return ค่าที่มากๆ
12. จากการ recursive ด้านบน จะสรุปได้ว่าทำการเลือก และไม่เลือกนิสิตทั้งหมดที่เป็นไปได้ และเลือกจำนวนนิสิตที่น้อยสุด