**算法分析与设计实验报告**

**第 3 次实验**

**10043 Bottle taps**

|  |
| --- |
| **Problem description** |
| Programmer Petrov has a hobby to collect beer-bottle taps. There’s nothing unusual — he knows hundreds of programmers that like beer. And they collect taps, too. Not everyone, but some of them. Frankly speaking, he has bought a part of his collection. But unfortunately he hasn’t got some rare taps to complete his collection. He has found some programmers over the Internet that are ready to sell him these taps. Some of the programmers sell the taps in sets with big discounts. It’s left to find an optimal offer. Petrov can explain to his wife why he is to store the taps but he won’t be able to prove why he is to spend money for the collection. So he is to buy the taps as cheap as possible. Petrov has written down all the variants and has started thinking. There’s no way to find out the solution of the problem without a program! |
| **Input** |
| The first line contains an integer N — an amount of available taps (1 <= N <= 20). The following N lines contain prices of bottles with the taps if one buys them in stores. The next line contains an integer M (0 <= M <= 100),an amount of offers to sell the taps. The following M lines describe the sets. The first number of each line is the price of the set and the second one is the amount of taps in the set. Then there are numbers of the taps in the set (each number lies in the range from 1 to N). The numbers in a set are unique. All the prices are positive integers and do not exceed 1000. The last line begins with the amount of taps that Petrov plans to buy. Then their numbers follow separated by spaces. These numbers are unique, too. |
| **Output** |
| The minimal sum of money that Petrov should spend on obtaining the necessary taps. |
| **Sample Input** |
| 4  10  11  12  13  3  17 2 1 3  25 3 2 3 4  15 2 3 4  3 1 3 4  0 |
| **Sample Output** |
| 25 |

**附录：完整代码**

#include <stdio.h>

#include <algorithm>

#define eps 1e-8

#define M 1000100

#define LL \_\_int64

#define INF 0x3f3f3f

#define PI 3.1415926535898

using namespace std;

const int maxn = 1010;

int dp[1 << 22];

int num[maxn];

int f[maxn];

int st[maxn];

int n;

void dfs(int x, int y, int z)

{

    if (x == n)

    {

        dp[y] = z;

        return;

    }

    dfs(x + 1, y, z);

    dfs(x + 1, y | (1 << x), z + num[x]);

}

int main()

{

    while (1)

    {

        scanf("%d", &n);

        if (!n)

            break;

        for (int i = 0; i < (1 << n); i++)

            dp[i] = INF;

        for (int i = 0; i < n; i++)

            scanf("%d", &num[i]);

        dfs(0, 0, 0);

        int m;

        scanf("%d", &m);

        for (int i = 0; i < m; i++)

        {

            scanf("%d", &f[i]);

            int x;

            scanf("%d", &x);

            int sum = 0;

            int tmx;

            for (int j = 0; j < x; j++)

            {

                scanf("%d", &tmx);

                sum |= (1 << (tmx - 1));

            }

            st[i] = sum;

        }

        int kk;

        scanf("%d", &kk);

        int ans = 0;

        int p;

        for (int i = 0; i < kk; i++)

        {

            scanf("%d", &p);

            ans |= (1 << (p - 1));

        }

        int Min = INF;

        for (int i = 0; i < (1 << n); i++)

        {

            if ((i & ans) == ans)

                Min = min(Min, dp[i]);

            for (int j = 0; j < m; j++)

                dp[i | st[j]] = min(dp[i | st[j]], dp[i] + f[j]);

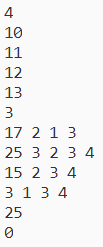
        }

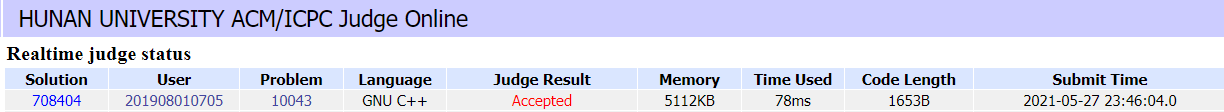
        printf("%d\n", Min);

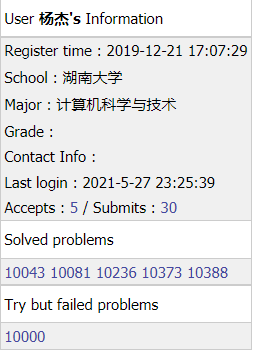
    }

    return 0;

}



****

****