IUT 2017 - Paying Bills

You are given N coins of value C_1, C_2, … , C_N and was told to pay some pending bills on M different stores. Bill of i-th store is B_i. You have to find out the minimum number of coins needed to pay all the bills.

There will be T (≤1000) test cases. Each case starts with a positive integer N (≤15) denoting the number of coins. Then comes the value of each coin C_i ($1 \le C_i \le 1000$). Next, there will be an integer M (≤N) denoting the number of stores followed by the bill of each store B_i ($1 \le B_i \le 1000$).

Print case number and the minimum number of coins needed. Print -1 if it is not possible to pay all the bills with given coins.

Sample Input **Sample Output**

```
2
6
1 2 3 4 5 6              Case 1: 3
2
9 3
6
1 2 3 4 5 6              Case 2: -1
3
4 4 4
```

****

Sample Code(TIME-1.3S):

```
int n,m,coin[20],bill[20],dp[16][(1<<15)+5];;
vector<int>taka[1009];

int cnt(int mask)
{
    int one=0;
    for(int i=0; i<n; i++){
        if(mask&(1<<i)) one++;
    }
    return one;
}
int take(int mask)
{
    int sum=0;
    for(int i=0; i<n; i++){
        if(mask & (1<<i))sum += coin[i];
    }
    return sum;
}
```

```
int fun(int shop,int mask)
{
    if(shop==m) return cnt(mask);

    int &ret=dp[shop][mask];
    if(ret!=-1) return ret;

    ret=n+12;
    int sz=taka[bill[shop]].size();
    for(int i=0; i<sz; i++)
    {
        int need = taka[bill[shop]][i];
        if((mask & need)==0){
            ret=min(ret,fun(shop+1, mask|need));
        }
    }
    return ret;
}

int main()
{
    int t; scanf("%d", &t);
    for(int ks=1; ks<=t; ks++)
    {
        scanf("%d", &n);
        for(int i=0; i<n; i++)scanf("%d", &coin[i]);
        scanf("%d", &m);
        for(int i=0; i<m; i++)scanf("%d", &bill[i]);

        for(int i=1; i<(1<<n); i++)
        {
            int ta = take(i);
            if(ta<=1000)taka[ta].push_back(i);
        }
        memset(dp,-1,sizeof(dp));

        int ans=fun(0,0);

        if(ans>n)ans=-1;
        printf("Case %d: %d\n",ks,ans);

        for(int i=0; i<1003; i++)taka[i].clear();
    }
    return 0;
}
```

Another Solution(TIME-0.1S):

```
#include <bits/stdc++.h>
using namespace std;

map < int, map < int, map<int,int> > > dp;
map < int, map < int, map<int,int> > > vs;
//map < int, map < int, int > > dp[17];
//map < int, map < int, int > > vs[17];

int coin[17],bill[17], n, m;

int solve(int shop, int mask, int val)
{
    if(shop==m)
    {
        int cnt=0;
        for (int i=0; i<n; i++){
            if ((mask & (1<<i))) cnt++;
        }
        return cnt;
    }

    if(mask == ((1<<n)-1)) return n+12;

    if(vs[shop][mask][val] != 0) return dp[shop][mask][val];
    vs[shop][mask][val] = 1;

    int ret = n+12;

    for(int i=0; i<n; i++)
    {
        if(((mask & (1<<i))==0) && (coin[i]<=val))
        {
            int rem = val - coin[i];
            if (rem == 0)
                ret = min (ret, solve(shop+1, mask | (1<<i), bill[shop+1]));
            else
                ret = min (ret, solve(shop, mask | (1<<i), rem));
        }
    }

    return dp[shop][mask][val] = ret;
}
```

```
int main()
{
    int t; scanf("%d", &t);
    for(int ks=1; ks<=t; ks++)
    {
        scanf("%d",&n);
        for(int i=0; i<n; i++)scanf("%d",&coin[i]);
        scanf("%d",&m);
        for(int i=0; i<m; i++)scanf("%d",&bill[i]);

        dp.clear();
        vs.clear();

     /*
        for(int i=0; i<17; i++){
            dp[i].clear();
            vs[i].clear();
        }
     */
        int ans = solve(0, 0, bill[0]);

        if(ans>n) ans = -1;
        printf("Case %d: %d\n", ks, ans);
    }

    return 0;
}
```