不设找零

一. 考察内容：

动态规划 状压DP

二. 题目分析：

[题目大意]

用n(n<=16)枚硬币按照顺序买一些商品，可以一次买多个商品，但是购买商品剩余的钱不能找零，求最后最多剩余多少钱。

[写题思路]

很显然，看到n的数据范围，就可以想到这是用状态压缩来写，设f[i]为使用集合i的硬币时最多能买到前多少个物品，转移也很显然，从i中找出一枚硬币j，f[i]=f[i-j]+j从f[i-j]+1开始能够买到的东西个数，由于购买的物品与物品的价值成正比，所以我们可以对于剩下可以购买的物品进行二分查找，找到j最多能买到的东西，然后转移即可。

对于每一个状态，转移完之后顺便更新一下答案，如果该状态能买到所有的物品，则更新答案。

三. 代码实现：

```
#define _CRT_SECURE_NO_DEPRECATE
/*************************
*创建时间：2018 09 18
*文件类型：源代码文件
*题目来源：COGS
*当前状态：已通过
*备忘录：动态规划 状压DP
*作者：HtBest
*************************/
#include <stdio.h>
#include <iostream>
#include <string>
#include <stdlib.h>
#include <string.h>
#include <algorithm>
#include <queue>
#include <bitset>
// #include <sys/wait.h>
// #include <sys/types.h>
// #include <unistd.h>
using namespace std;
int n,m,a[20],f[1<<20];
long long b[100001];
namespace IO
{
    #define SIZE 100000
    char s[SIZE+1];
    int T=0,end=0;
    int in()
    {
        T=0;
        return end=fread(s,1,SIZE,stdin);
    }
    int read(int &a)
    {
        int i=a=0;
        while(1)
        {
            if(T==end&&!in())return i;
            if(s[T]==' '||s[T]=='\n'){if(i)return i;}
```

```
            else {a=a*10+s[T]-'0';++i;}
            ++T;
        }
    }
    int read(long long &a)
    {
        int i=0;
        while(1)
        {
            if(T==end&&!in())return i;
            if(s[T]==' '||s[T]=='\n'){if(i)return i;}
            else {a=a*10+s[T]-'0';++i;}
            ++T;
        }
    }
}
/* Variable explain:

*/
void read()
{
    using IO::read;
    read(n);
    read(m);
    for(int i=0;i<n;++i)read(a[i]);
    for(int i=1;i<=m;++i)read(b[i]),b[i]+=b[i-1];
    return;
}
int find(int x,int start)
{
    int l=start,r=m,ans=0;
    while(l<=r)
    {
        int mid=(l+r)>>1;
        if(b[mid]-b[start-1]<=x)ans=mid,l=mid+1;
        else r=mid-1;
    }
    return ans;
}
long long value(int x)
{
    long long ans=0;
    for(int i=0;i<n;++i)if(x&(1<<i))ans+=a[i];
    return ans;
}
void dp()
{
    int U=(1<<n)-1;
    long long ans=-1;
    for(int i=1;i<=U;++i)
    {
        for(int j=0;j<n;++j)
        {
            if((i&(1<<j))==0)continue;
            int now=find(a[j],f[i^(1<<j)]+1);
            f[i]=max(f[i],now);
        }
        // printf("%d\n",f[i]);
        if(f[i]==m)ans=max(ans,value(U)-value(i));
    }
    printf("%lld",ans);
}
int main()
{
    freopen("nochange.in","r",stdin);
    freopen("nochange.out","w",stdout);
    read();
    dp();
    return 0;
}
```

<题目跳转> <查看代码>