**★★★☆☆**

**題組：Problem Set Archive** **with Online Judge**

**題號：10130:SuperSale**

**解題者：莊弼堯**

**解題日期：2013年5月9日**

**題意：**

Background描述:
有一家人到拍賣會買東西，每人對同一種商品最多僅可拿取一次， 每個人都有自己可承擔的負重，希望能有最大利潤 (即是0/1 Knapsack problem)
Input描述:
共有T筆測資數(1~1000)，每組測資皆有N種商品種類(1~1000)，而每種商品各自有自己本身的利潤(P=1~100)和重量(W=1~30)，
且家庭成員數一共有G人(1~100)，每個成員都有自己最大可拿取負重 (MW=1~30) ，求該總成員數最大可獲得的總利潤為多少？
**題意範例：**

**Input / Output**

 **2**

** 3 3種商品**

 **72 17**

 **44 23利潤/重量**

** 31 24**

 **1成員數 1人**

 **26 負重**

** 6 6種商品**

 **64 26**

 **85 22**

** 52 4利潤/重量**

 **99 18**

 **39 13**

 **54 9**

 **4成員數 4人**

 **23**

 **20負重**

 **20**

 **26**

**解法：**

**思考方向**:在題目的測資中每位成員的最大負重最多為30，所以我們僅須把1~30的每種重量所能夠得到的最大利潤求出來，即可做出相對應的選擇。
利用Dynamic Program，先建立一個cost table，尋找每一種重量最大可以拿取

的最大利潤，每一次都檢查每種重量是否可以拿取該商品，
若無法拿取該商品，則保持原來所擁有的利潤值。
若可以拿取該商品，則拿取商品，再將目前所能夠負載的重量扣除商品的重量(剩餘的重量)，從cost table找尋目前該重量所能夠得到的最大cost，兩個利潤值相加，之後再進行比較

關係式:
最後所選擇的最大利潤=Max{目前該重量所擁有的最大利潤，
 目前選進來的商品利潤+剩餘重量的最大利潤}

 一件一件商品逐一按照上述步驟，最終即可得到每一種重量所能夠得到的最大利潤。

 最後再從每位成員所能夠負載的最大重量，從cost table找尋所相對應的重量所能夠得到的最大利潤，相加得到總合，即可得到總合為最大值。

**解法範例：**

以前面的第二組測資當例子

**6種商品:**

****

**4位成員:**

****

****

**第一步驟 i=1: p=64,w=26**

****

****

**第二步驟 i=2: p=85,w=22**

****

****

**第三步驟 i=3: p=52,w=4**

****

****

**第四步驟 i=4: p=99,w=18**

****

****

**第五步驟 i=5: p=39,w=13**

****

****

**第六步驟 i=6: p=54,w=9**

****

****

****

最後再根據cost Table中的值，就可以找出每一種重量可拿取的最大利潤值

**討論：**

本題題目限制條件：

1.MW(每位成員可承擔負重)=1~30

(1)**演算法方面** :

上述的使用方法為演算法中的DP(dynamin program:動態分配) ，

使用DP特點:
(1)找尋所有可能的結果
(2)相同答案不重複計算，以空間換取時間。
在找尋完所有可能後即可回答所有種重量所能夠得到的最大利潤。
(2)**時間複雜度方面**:

=>建立cost table的過程中，使用DP會將所有種商品逐一加進cost table 且每次所加進來的比較次數為商品的重量~30(成員最大可承擔重量) 取平均值為15次，所以比較次數並不會超時

=>O(商品種類數目\*(成員最大可承擔負重/2) )

=>O(N\*MW/2)

=>所以時間複雜度為**O(m\*n)**

**程式：**

#include <stdio.h>

#define N 1005

#define W 35

int profit[N], weight[N];

int main(void){

 int t,n,ground,sum;

 int i,j;

 scanf("%d",&t);

 while(t--){

 int cost[W]={0};

 scanf("%d",&n);

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

 scanf("%d%d",&profit[i],&weight[i]);

/\*\_\_\_\_\_\_\_\_\_\_\_建立cost table(By Dynamic program)\_\_\_\_\_\_\_\_\_\_\_\_\_\*/

 for(i=1;i<=n;i++){

 for(j=30;j>=weight[i];j--){

 if(cost[j]<cost[j-weight[i]]+profit[i])

 cost[j]=cost[j-weight[i]]+profit[i];

 }

 }

/\*依照成員每位成員的最大可承受負重 從cost table中找尋答案\*/

 scanf("%d",&ground);

 for(i=1,sum=0;i<=ground;i++){

 scanf("%d",&j);

 sum+=cost[j];

 }

 printf("%d\n",sum);

 }

 return 0;

}