背包1.0

01背包，完全背包和有依赖的背包问题

题目描述(p1064)

题目描述 金明今天很开心，家里购置的新房就要领钥匙了，新房里有一间金明自己专用的很宽敞的房间。更让他高兴的是，妈妈昨天对他说：“你的房间需要购买哪些物品，怎么布置，你说了算，只要不超过NN元钱就行”。今天一早，金明就开始做预算了，他把想买的物品分为两类：主件与附件，附件是从属于某个主件的，下表就是一些主件与附件的例子：

主件	附件
电脑	打印机，扫描仪
书柜	图书
书桌	台灯，文具
工作椅	无

如果要买归类为附件的物品，必须先买该附件所属的主件。每个主件可以有0个、1个或2个附件。附件不再有从属于自己的附件。金明想买的东西很多，肯定会超过妈妈限定的N元。于是，他把每件物品规定了一个重要度，分为5等：用整数1-5表示，第5等最重要。他还从因特网上查到了每件物品的价格（都是10元的整数倍）。他希望在不超过N元（可以等于N元）的前提下，使每件物品的价格与重要度的乘积的总和最大。

设第j件物品的价格为v[j]，重要度为w[j]，共选中了k件物品，编号依次为j1,j2,...jk，则所求的总和为： v[j1]×w[j1]+v [j2]×w[j2]+…+v[jk]×w[jk]。

请你帮助金明设计一个满足要求的购物单。

输入格式

第1行，为两个正整数，用一个空格隔开：

N m其中(N<32000)表示总钱数,(m<60)为希望购买物品的个数。从第2行到第m+1行，第j行给出了编号为j−1的物品的基本数据，每行有3个非负整数

vpq （其中v表示该物品的价格（v<10000），p表示该物品的重要度（1−5），q表示该物品是主件还是附件。如果q=0，表示该物品为主件，如果q>0，表示该物品为附件,q是所属主件的编号）

> 输出格式 一个正整数，为不超过总钱数的物品的价格与重要度乘积的总和的最大值（<200000）。

```
输入                输出 2200
1000 5
800 2 0
400 5 1
300 5 1
400 3 0
500 2 0
```

思路：

1. 01背包是所有元素有且只有一个
2. 完全背包是可用元素有无穷多个

本题是01背包的有多个条件的背包问题，简而言之就是递推的dp

值得注意的是：j>=u[i]不能是j>=0,要保证剩下的钱做购买东西。

注意两层循环中，第二层一定是倒着减的，正着减是完全背包问题

code

```cpp
#include <cstdio>
#include<iostream>
#include <cstring>
using namespace std;
int main(){
    int N,m;
    int u[65];
    int p[65];
    int q[65];
    int q1[65];//附件1
    int q2[65];//附件2
    int money[35000];
    memset(q,0,sizeof(q));
    memset(q1,0,sizeof(q1));
    memset(q2,0,sizeof(q2));
    memset(money,0,sizeof(money));
    memset(p,0,sizeof(p));
    cin>>N>>m;
    for(int i=1;i<=m;i++){
        cin>>u[i]>>p[i]>>q[i];
        if(q[i]!=0){
```

```
            if(q1[q[i]]==0) q1[q[i]]=i;//绑定附件1
            else if(q2[q[i]]==0) q2[q[i]]=i;//绑定附件2
        }
    }
    for(int i=1;i<=m;i++){   //背包模版
        for(int j=N;j>=u[i];j--){ //01背包
        //写成for(int j=u[i];j<=N;j++)便是完全背包
            if(q[i]==0){
                money[j]=max(money[j],money[j-u[i]]+u[i]*p[i]);
                //判断是否买主件
                if(q1[i]!=0){
                    if(j-u[i]-u[q1[i]]>=0)//买主件和附件1
                    {
                     money[j]=max(money[j],
                     money[j-u[i]-u[q1[i]]]+u[i]*p[i]+u[q1[i]]*p[q1[i]]);
                }
                }
                if(q2[i]!=0){
                    if(j-u[q2[i]]-u[i]>=0)  {//买主件和附件2
                    money[j]=max(money[j],
                    money[j-u[i]-u[q2[i]]]+u[i]*p[i]+u[q2[i]]*p[q2[i]]);
                    }
                    if(j-u[q2[i]]-u[i]-u[q1[i]]>=0) {//买主件，附件1，附件2
                    money[j]=max(money[j],
                    money[j-u[i]-u[q2[i]]-u[q1[i]]]
                    +u[i]*p[i]+u[q2[i]]*p[q2[i]]+u[q1[i]]*p[q1[i]]);
                    }
                }
            }
        }
    }
    cout<<money[N];
    return 0;
}
```