题意：买东西，每个东西有三个特征值，p代表价格，q代表你手中钱必须不低于q才能买这个物品，v代表得到的价值。

这道题和01背包很类似，唯一不同的就是买每件物品时多了一个限制条件，p，Q，v，需要钱数大于Q才能买，所以买东西需要有先后顺序。以前的01背包不用排序，dp[j] = max(dp[j],dp[j-w[i]]+v[i])表达的是这第i个物品要不要买，但同时也在判断第i个物品要不要先买，如果先买剩下的空间随便放，看是否得到的价值会更大。

        A物品  p1,q1,v1;

        B物品  p2,q2,v2

        若先买A，至少需要p1+q2 (p1 == q1情况下）若现买B，至少需要p2+q1;

        若A<B,应先买A，即 q1-p1 > q2-p2;即差值大的先买，所以先买的应排到后面

#include<iostream>

#include<cstdio>

#include<algorithm>

#include<cstring>

#include<iomanip>

#include<cstring>

#include<cstdlib>

#include<cstring>

#define ll long long

using namespace std;

struct node

{

int q,p,v;

}a[510];

bool cmp(node a,node b)

{

return (a.q-a.p)<(b.q-b.p);//按理说应该是从大到小排序，但因为dp是倒着的，所以先拿差值大的

}

int dp[5010];

int main()

{

//freopen("input.txt","r",stdin);

int n,m;

while(cin>>n>>m)

{

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

cin>>a[i].p>>a[i].q>>a[i].v;

sort(a,a+n,cmp);

memset(dp,0,sizeof(dp));

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

for(int j=m;j>=a[i].q;j--)

dp[j]=max(dp[j],dp[j-a[i].p]+a[i].v);//因为这是倒着，所以先拿差值大的

cout<<dp[m]<<"\n";

}

return 0;

}