# Cdq

//3842

#include<iostream>

#include<string>

#include<algorithm>

using namespace std;

const int M = 223456;

long long dp[M];

long long P[M],R[M],G[M],D[M];

int head,tail,n,d;

struct pos{

int num;

int val;

pos(int num=0,int val=0):num(num),val(val){}

}p[M];

long long getDP(int i,int j){

long long t1=G[j];t1=t1\*(D[i]-D[j]-1);

return (dp[j]-P[i]+R[j]+t1);

}

long long getUP(int j,int k){ //yj-yk的部分{

long long t1=-G[j];t1=t1\*D[j];

long long t2=+G[k];t2=t2\*D[k];

return (dp[j]+R[j]-G[j]-(dp[k]+R[k]-G[k]))+t1+t2;

}

int getDOWN(int j,int k){ //xj-xk的部分

return (G[j]-G[k]);

}

int cmp(pos A,pos B){

return A.val<B.val;

}

int q[M];

int check(int i,int tail){

double t1=getUP(p[i].num,q[tail-1])\*1.0;

t1=t1\*getDOWN(q[tail-1],q[tail-2]);

double t2=getUP(q[tail-1],q[tail-2])\*1.0;

t2=t2\*getDOWN(p[i].num,q[tail-1]);

return t1>=t2;

}

int cdq(int l,int r){

if(l==r){return 0;}

int mid=(l+r)>>1;

cdq(l,mid);

int sz=0;

for(int i=l;i<=mid;i++){

sz++;

p[sz].num=i,p[sz].val=G[i];

}

sort(1+p,1+p+sz,cmp);

head=tail=0;

q[tail++]=0;

for(int i=1;i<=sz;i++){

if(dp[p[i].num]<0) continue;

while(head+1<tail && check(i,tail)){

tail--;

}

q[tail++]=p[i].num;

}

for(int i=mid+1;i<=r;i++){

while(head+1<tail&& getDP(i,q[head])<=getDP(i,q[head+1])){

head++;

}

dp[i]=max(dp[i],getDP(i,q[head]));

}

cdq(mid+1,r);

return 0;

}

int t[M];

int init();

int main(){

int cas=0;

//freopen("data.in","r",stdin);

while(cin>>n>>dp[0]>>d){

cas++;

if(n==0) break;

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

scanf("%d%d%d%d",&D[i],&P[i],&R[i],&G[i]);

p[i].num=i;

p[i].val=D[i];

}

n++;

dp[n]=0.0;

D[n]=(d+1);P[n]=0;R[n]=0;G[n]=0;

p[n].val=D[n];

p[n].num=n;

init();

cdq(1,n);

printf("Case %d: ",cas);

printf("%lld\n",dp[n]);

}

return 0;

}

int init(){

sort(1+p,1+p+n,cmp);

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

t[i]=D[i];

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

D[i]=t[p[i].num];

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

t[i]=P[i];

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

P[i]=t[p[i].num];

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

t[i]=R[i];

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

R[i]=t[p[i].num];

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

t[i]=G[i];

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

G[i]=t[p[i].num];

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

dp[i]=dp[0]-P[i];

}

dp[0]=0;

return 0;

}