```cpp
// Problem: Machine Works
// Contest: Virtual Judge - UVA

// x*m1+c1 = x*m2+c2
// x(m1-m2) = c2-c1
// x=(c2-c1)/(m1-m2)

double intersect(const pii &l1, const pii &l2){
    return double(l2.second-l1.second)/double(l1.first-l2.first);
}

bool usefull(const pii &l1, const pii &l2, const pii &l3){
    double d12 = intersect(l1, l2);
    double d23 = intersect(l2, l3);
    return d12 <= d23;
}

int calc(const pii &seg, int x){
    return x * seg.first + seg.second;
}

int32_t main(){

    fastIO;

#ifdef LOCAL
    freopenI;
    freopenO;
#endif

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

    int tt = 1;
    int n, c, d;
    while(cin >> n >> c >> d, n){

        // d p r g

        vi D(n), P(n), R(n), G(n);
        for(int i = 0; i < n; i++){
            cin >> D[i] >> P[i] >> R[i] >> G[i];
        }

        n++;
        D.push_back(d+1);
        P.push_back(0);
        R.push_back(c);
        G.push_back(0);

        vi pos(n);
        iota(pos.begin(), pos.end(), 0);
        sort(pos.begin(), pos.end(), [&](int i, int j){
            return D[i] < D[j];
        });

        map<int, int> mp;
        mp[0] = c;
        int cans;

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

            int p = pos[i];
            int x = D[p];

            while(mp.size() > 1){

                auto f = mp.begin();
                auto s = mp.begin(); s++;

                if(calc(*f, x) <= calc(*s, x)){
                    mp.erase(f);
                }
                else{
```

```
                    break;
                }
            }

            cans = calc(*mp.begin(), x);
            if(cans < P[p]) continue;

            // dp[i] = dp[j] + arr[j].R - arr[j].P + arr[j].G * (arr[i].D - arr[j].D
            -1);
            // dp[i] = arr[i].D * arr[j].G + (dp[j] + arr[j].R - arr[j].P - arr[j].G
            * (arr[j].D+1))
            // x = arr[i].D
            // m = arr[j].G
            // c = (dp[j] + arr[j].R - arr[j].P - arr[j].G * (arr[j].D+1)

            int m = G[p];
            int c = cans + R[p] - P[p] - G[p] * (D[p]+1);

            auto l2 = mp.emplace(m, c).first;
            l2->second = max(l2->second, c);

            if(l2 != mp.begin() && l2 != --mp.end()){

                auto l1 = l2; l1--;
                auto l3 = l2; l3++;

                if(!usefull(*l1, *l2, *l3)){
                    mp.erase(l2);
                    continue;
                }
            }

            while(mp.size() > 1 && l2 != mp.begin() && l2 != ++mp.begin()){

                auto l1 = l2; l1--;
                auto l0 = l1; l0--;

                if(!usefull(*l0, *l1, *l2)){
                    mp.erase(l1);
                }
                else{
                    break;
                }
            }
        }

        cout << "Case " << tt++ << ": " << cans << endl;
    }
}

```