E. Two Platforms

time limit per test

2 seconds

memory limit per test

256 megabytes

input

standard input

output

standard output

There are nn points on a plane. The ii-th point has coordinates (xi,yi)(xi,yi). You have two horizontal platforms, both of length kk. Each platform can be placed anywhere on a plane but it should be placed **horizontally** (on the same yy-coordinate) and have **integer borders**. If the left border of the platform is (x,y)(x,y) then the right border is (x+k,y)(x+k,y) and all points between borders (including borders) belong to the platform.

Note that platforms can share common points (overlap) and it is not necessary to place both platforms on the same yy-coordinate.

When you place both platforms on a plane, all points start falling down decreasing their yy-coordinate. If a point collides with some platform at some moment, the point stops and is **saved**. Points which never collide with any platform are lost.

Your task is to find the maximum number of points you can **save** if you place both platforms optimally.

You have to answer tt independent test cases.

For better understanding, please read the **Note** section below to see a picture for the first test case.

**Input**

The first line of the input contains one integer tt (1≤t≤2⋅1041≤t≤2⋅104) — the number of test cases. Then tt test cases follow.

The first line of the test case contains two integers nn and kk (1≤n≤2⋅1051≤n≤2⋅105; 1≤k≤1091≤k≤109) — the number of points and the length of each platform, respectively. The second line of the test case contains nn integers x1,x2,…,xnx1,x2,…,xn (1≤xi≤1091≤xi≤109), where xixi is xx-coordinate of the ii-th point. The third line of the input contains nn integers y1,y2,…,yny1,y2,…,yn (1≤yi≤1091≤yi≤109), where yiyi is yy-coordinate of the ii-th point. All points are distinct (there is no pair 1≤i<j≤n1≤i<j≤n such that xi=xjxi=xj and yi=yjyi=yj).

It is guaranteed that the sum of nn does not exceed 2⋅1052⋅105 (∑n≤2⋅105∑n≤2⋅105).

**Output**

For each test case, print the answer: the maximum number of points you can save if you place both platforms optimally.

**Example**

**input**

**Copy**

4

7 1

1 5 2 3 1 5 4

1 3 6 7 2 5 4

1 1

1000000000

1000000000

5 10

10 7 5 15 8

20 199 192 219 1904

10 10

15 19 8 17 20 10 9 2 10 19

12 13 6 17 1 14 7 9 19 3

**output**

**Copy**

6

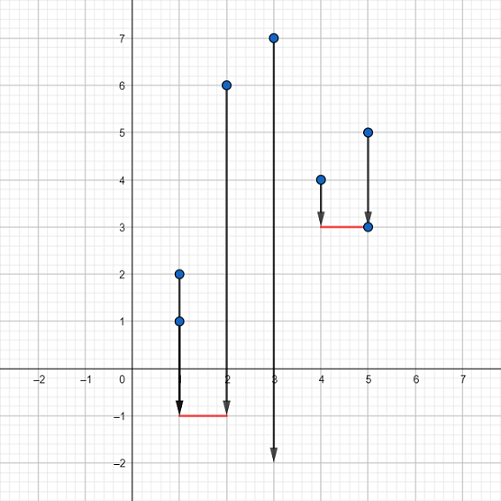
1

5

10

**Note**

The picture corresponding to the first test case of the example:



Blue dots represent the points, red segments represent the platforms. One of the possible ways is to place the first platform between points (1,−1)(1,−1) and (2,−1)(2,−1) and the second one between points (4,3)(4,3) and (5,3)(5,3). Vectors represent how the points will fall down. As you can see, the only point we can't save is the point (3,7)(3,7) so it falls down infinitely and will be lost. It can be proven that we can't achieve better answer here. Also note that the point (5,3)(5,3) doesn't fall at all because it is already on the platform.

#include<bits/stdc++.h>

#define pb push\_back

#define pii pair<int,int>

#define int long long int

#define vec vector<int>

#define mp map<int,int>

#define inf 1e18

using namespace std;

int32\_t main()

{

ios\_base::sync\_with\_stdio(false);

cin.tie(NULL);

cout.tie(NULL);

int tt=1;

cin>>tt;

while(tt--)

{

int n,k;

cin>>n>>k;

int a[n],i,j,b[n];

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

cin>>a[i];

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

cin>>b[i];

sort(a,a+n);

int l[n];

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

l[i]=1;

i=0,j=0;

while(j<n && i<n)

{

if(j+1<n && a[j+1]-a[i]<=k)

j++;

else

i++;

l[j]=max(j-i+1,l[j]);

if(j>0)

l[j]=max(l[j],l[j-1]);

}

int r[n];

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

r[i]=1;

i=n-1,j=n-1;

while(j>=0 && i>=0)

{

if(j-1>=0 && abs(a[j-1]-a[i])<=k)

j--;

else

i--;

r[j]=max(i-j+1,r[j]);

if(j<n-1)

r[j]=max(r[j],r[j+1]);

}

int ans=1;

for(i=0;i<n-1;i++)

{

ans=max(ans,l[i]+r[i+1]);

}

cout<<ans<<"\n";

}

}