# Hackerland Radio Transmitters

locked

by nabila_ahmed

## Editorial by nabila_ahmed

Sort all the $x[i]$s then avoid duplicate numbers. Duplicate numbers are avoided because they do not affect our answer.

Now, starting from left when we find an uncovered location ($x$), from $x$ try to give the longest jump which fits within $x+k$. The location that we get, let say $p$, on $p$ we place our radio transmitter. From $p$ again try to give the longest jump that fits within $p+k$. Therefore, by only two jumps we can cover the maximum possible area which is under the influence of one transmitter. Then we move to the next uncovered position which is just the next right position of the covered area.

But the question is how to find the longest jump?

There are two approaches:

1) Binary search (Complexity: *O(nlogn)*). Since the locations are sorted we can easily find the lower bound of our target positions. **See Setter's Code**

2) Two pointers (Complexity: *O(n)*). Just iterate through the $x[i]$ until the position is less than our target position. Since we are not testing any $x[i]$ more than once, we can solve this in *O(n)*. **See Tester's Code**

## Statistics

| | |
|---|---|
| Difficulty: | Medium |
| Time Complexity: | $O(n)$ Required |
| Knowledge: | Two pointer, Binary search |
| Publish Date: | Oct 21 2016 |

## Set by nabila_ahmed

Problem Setter's code :

```
#include<bits/stdc++.h>
using namespace std;

vector<int>v;
map<int, bool>mp;

int main() {
    v.clear();
    mp.clear();

    int n, k, x, low, center, cnt = 0, lft;

    cin>>n>>k;

    for(int i=0; i<n; i++)
    {
       cin>>x;
       if(mp[x] == 0)
         v.push_back(x);
       mp[x]= 1;
    }
    sort(v.begin(), v.end());

    n = v.size();
    lft = 0;
    while(lft < n)
    {
       x = v[lft]+k;
       low = lower_bound(v.begin(), v.end(), x) - v.begin();
       if(low >= n || v[low] > x)
        low--;
       center = v[low];

       x = center+k;
       low = lower_bound(v.begin(), v.end(), x) - v.begin();
       if(low >= n || v[low] > x)
        low--;

       lft = low+1;

       cnt++;
    }

    cout<<cnt<<endl;

    return 0;
}
```

**Tested by Shafaet**

Problem Tester's code :

```
#include <bits/stdc++.h>

using namespace std;

int a[100000+2];
int main(){

    int n, m;
    cin>>n>>m;
    for(int i=1;i<=n;i++)
    {
        cin>>a[i];
    }

    sort(a+1,a+n+1);
    int ans = 0, i=1;
    while(i<=n)
    {
        int maxijabe = a[i] + m;
        int j=i;
        ans = ans + 1;
        while(j<=n and a[j]<=maxijabe){
            i = j;
            j++;
        }

        maxijabe = a[i]+m;
        j = i;
        while(j<=n and a[j]<=maxijabe){
            i = j;
            j++;
        }
        i = i + 1;
    }
    cout<<ans<<endl;

    return 0;
}
```