Practice Compete Jobs Rank Leaderboard gadhiya

Dashboard > Algorithms > Dynamic Programming > Police Operation

Badge Progress ★★★★

Points: 4727.88 Rank: 491

Police Operation

by Bidhan

Problem Submissions Leaderboard Discussions Editorial

Roy is helping the police department of his city in crime fighting. Today, they informed him about a new planned operation.

Think of the city as a $2D$ plane. The road along the X-axis is very crime prone, because n criminals live there. No two criminals live at the same position.

To catch these criminals, the police department has to recruit some police officers and give each of them USD h as wages. A police officer can start his operation from any point a, drive his car to point b in a straight line, and catch all the criminals who live on this way. The cost of fuel used by the officer's car is equal to the square of the euclidean distance between points a and b (Euclidean distance between points (x_1, y_1) and (x_2, y_2) equals to $\sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$).

The police department asks Roy to plan this operation. So Roy has to tell them the number of officers to recruit and the routes these officers should take in order to catch all the criminals. Roy has to provide this information while minimizing the total expenses of this operation.

Find out the minimum amount of money required to complete the operation.

Input Format

The first line contains two integers n $(0 \leq n \leq 2 \times 10^6)$, number of criminals, and h $(0 \leq h \leq 10^9)$, the cost of recruiting a police officer. The next line contains n space separated integers. The i^{th} integer indicates the position of the i^{th} criminal on X-axis (in other words, if the i^{th} integer is x, then location of the i^{th} criminal is $(x, 0)$). The value of the positions are between 1 and 10^9 and are given in increasing order in the input.

Output Format

Print the minimum amount of money required to complete the operation.

Sample Input

```
5 10
1 4 5 6 9
```

Sample Output

```
34
```

Explanation

For the sample test case, police department recruits 3 officers who get paid $3 \times 10 = 30$. The first officer starts at point $(1, 0)$ and catches the criminal there. So he does not use any fuel. The second officer catches criminals at points $(4, 0)$, $(5, 0)$ and $(6, 0)$. He burns fuel worth USD 4. The third officer catches the criminal at point $(9, 0)$. He also does not burn any fuel. The total money spent by the department is, $30 + 4 = 34$.

Timelimits

Timelimits for this challenge are given here

Submissions: 301

Max Score: 100

Difficulty: Hard

Rate This Challenge:

☆☆☆☆☆

More

Current Buffer (saved locally, editable)

Java 8

```java
import java.io.*;
import java.util.*;

public class Solution {

    public static void main(String[] args) throws IOException{

        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        new Solution(br);
    }

    public  Solution(BufferedReader br) throws IOException{


        String[] num = br.readLine().split("\\s");

        int n = Integer.parseInt(num[0]);
        long h = Long.parseLong(num[1]);

        if(n == 0){
            System.out.println(0);
            return;
        }

        if(h == 1){
            System.out.println(n);
            return;
        }

        num = br.readLine().split("\\s");

        long bestPrev = 0;
        int fptr = 0;
        int bptr = 0;
        Sol[] dq = new Sol[n];

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

            int x = Integer.parseInt(num[i]);

            Sol part = new Sol(x,bestPrev+h);

            while(bptr - fptr >= 2 && dq[bptr - 2].to(dq[bptr - 1]) >= dq[bptr - 2].to(part))
                bptr--;

            dq[bptr++] = part;

            while(fptr+1 < bptr && dq[fptr].getCost(x) >= dq[fptr+1].getCost(x))
                fptr++;

            bestPrev = dq[fptr].getCost(x);

        }

        System.out.print(bestPrev);

    }

    class Sol{

        long x;
        long prevCost;

        Sol(long x,long prevCost){
            this.x = x;
            this.prevCost = prevCost;
        }

        long getCost(long y){
            long d = x - y;
            return d * d + prevCost;
        }

        long to(Sol rhs){

            long lo = rhs.x;
            long hi = rhs.x;

            while(getCost(hi) < rhs.getCost(hi)){
                lo = hi+1;
                hi *= 2;
            }

            while(lo < hi){
                long m = (hi + lo) / 2;

                if(getCost(m) >= rhs.getCost(m)){
                    hi = m;
                }
                else{
                    lo = m+1;
                }
            }

            return hi;

        }

    }

}
```

Line: 1 Col: 1

Upload Code as File □ Test against custom input Run Code Submit Code

Join us on IRC at #hackerrank on freenode for hugs or bugs.

Contest Calendar | Interview Prep | Blog | Scoring | Environment | FAQ | About Us | Support | Careers | Terms Of Service | Privacy Policy | Request a Feature