Cloudy Day

Problem | Submissions | Leaderboard | Discussions | Editorial

Quibdó in Colombia is one among the cities that receive maximum rainfall in the world.

All year round, the city is covered in clouds. The city has many towns, located on a one-dimensional line. The positions and populations of each town on the number line are known to you. Every cloud covers all towns located at a certain distance from it. A town is said to be in darkness if there exists at least one cloud such that the town is within the cloud's range. Otherwise, it is said to be sunny.

The city council has determined that they have enough money to remove exactly one cloud using their latest technology. Thus they want to remove the cloud such that the fewest number of people are left in darkness after the cloud is removed. What is the maximum number of people that will be in a sunny town after removing exactly one cloud?

Note: If a town is not covered by any clouds, then it is already considered to be sunny, and the population of this town must also be included in the final answer.

Complete the function `maximumPeople` which takes four arrays representing the populations of each town, locations of the towns, locations of the clouds, and the extents of coverage of the clouds respectively, and returns the maximum number of people that will be in a sunny town after removing exactly one cloud.

Input Format

The first line of input contains a single integer n, the number of towns.

The next line contains n space-separated integers p_i. The i^{th} integer in this line denotes the population of the i^{th} town.

The next line contains n space-separated integers x_i denoting the location of the i^{th} town on the one-dimensional line.

The next line consists of a single integer m denoting the number of clouds covering the city.

The next line contains m space-separated integers y_i the i^{th} of which denotes the location of the i^{th} cloud on the coordinate axis.

The next line consists of m space-separated integers r_i denoting the range of the i^{th} cloud.

Note: The range of each cloud is computed according to its location, i.e., the i^{th} cloud is located at position y_i and it covers every town within a distance of r_i from it. In other words, the i^{th} cloud covers every town with location in the range $[y_i - r_i, y_i + r_i]$.

Constraints

- $1 \leq n \leq 2 \times 10^5$
- $1 \leq m \leq 10^5$
- $1 \leq x_i, y_i, r_i, p_i, \leq 10^9$

Output Format

Print a single integer denoting the maximum number of people that will be in a sunny town by removing exactly one cloud.

Sample Input 0

```
2
10 100
5 100
1
4
1
```

Sample Output 0

Author	VastoLorde95
Difficulty	Medium
Max Score	45
Submitted By	1281

NEED HELP?

View discussions

View editorial

View top submissions

RATE THIS CHALLENGE

☆ ☆ ☆ ☆ ☆

MORE DETAILS

Download problem statement

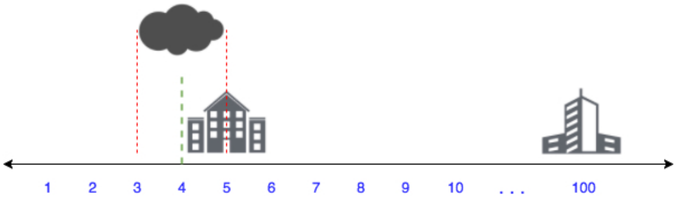
Download all test cases

Suggest Edits

```
110
```

Explanation 0

In the sample case, there is only one cloud which covers the first town. Our only choice is to remove this sole cloud which will make all towns sunny, and thus, all 110 people will live in a sunny town.



As you can see, the only cloud present, is at location 4 on the number line and has a range 1, so it covers towns located at 3, 4 and 5 on the number line. Hence, the first town is covered by this cloud and removing this cloud makes all towns sunny.

Change Theme   Java 7

```
import java.io.*;
import java.math.*;
import java.security.*;
import java.text.*;
import java.util.*;
import java.util.concurrent.*;
import java.util.regex.*;

public class Solution {

    // Complete the maximumPeople function below.
    static long maximumPeople(long[] p, long[] x, long[] y, long[] r) {
        // Return the maximum number of people that will be in a sunny town after removing exactly one cloud.

    }

    private static final Scanner scanner = new Scanner(System.in);

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

Line: 1 Col: 1

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