Lottery

JOI-kun is planning a lottery event. In this lottery event, an even number of bags will be used. Each bag initially contains some red balls and blue balls (possibly zero). Participants will keep coming to the lottery event until at least one of the bags becomes empty. Each participant draws one ball from each bag. If the total number of red and blue balls they have drawn ends up being equal, they receive one prize. Balls drawn are not returned to the bags.

As preparation, JOI-kun has prepared N bags, numbered from 0 to $N-1$. Bag i ($0 \leq i \leq N-1$) contains X_i red balls and Y_i blue balls.

In the lottery event, some of the N bags will be selected for use. There are Q plans for selecting bags. In the j-th plan ($1 \leq j \leq Q$), the bags $L_j, L_j + 1, \ldots, R_j$ are used. Here, $R_j - L_j + 1$ is even.

To prepare the prizes for the event, JOI-kun wants to know, for each plan, the maximum possible total number of prizes participants can obtain. Write a program that, given the bag contents and the plans, returns the maximum possible total number of prizes participants can obtain for each plan.

Implementation Details

You need to submit one file.

The file is `lottery.cpp`. The program should include `lottery.h` using the preprocessing directive `#include`.

In `lottery.cpp`, the following functions should be implemented.

- `void init(int N, int Q, std::vector<int> X, std::vector<int> Y)`
  - This function is called only once, at the beginning.
  - The parameter `N` is the number of bags N prepared by JOI-kun.
  - The parameter `Q` is the number of plans Q for selecting bags.
  - The parameter `X` is an array of length N. `X[i]` $(0 \leq i \leq N-1)$ is the number of red balls in bag i.
  - The parameter `Y` is an array of length N. `Y[i]` $(0 \leq i \leq N-1)$ is the number of blue balls in bag i.
- `int max_prize(int L, int R)`
  - This function is called Q times after the function `init` is called.
  - On the j-th call $(1 \leq j \leq Q)$ of this function:
    - The parameter `L` is the value L_j of the j-th plan.
    - The parameter `R` is the value R_j of the j-th plan.
    - This function must return the maximum possible total number of prizes participants can obtain in the j-th plan.

Important Notices

- Your program can implement other functions for internal use, or use global variables.
- Your program must not use the standard input and the standard output. Your program must not communicate with other files by any methods. However, your program may output debugging information to the standard error.

Compilation and Test Run

You can download an archive file from the contest webpage which contains the sample grader to test your program. The archive file also contains a sample source file of your program.

The sample grader is the file `grader.cpp`. To test your program, place the files `grader.cpp`, `lottery.cpp`, and `lottery.h` in the same directory, and compile them using the following command:

```
g++ -std=gnu++20 -O2 -o grader grader.cpp lottery.cpp
```

Alternatively, you can execute the `compile.sh` file included in the archive by running:

```
./compile.sh
```

If the compilation is successful, an executable file named `grader` will be generated.

Note that the actual grader is different from the sample grader. The sample grader will be executed as a single process, which will read input data from the standard input and write the results to the standard output.

Input for the Sample Grader

The sample grader reads input from standard input in the following format:

N Q
X_0 X_1 $\cdots$ X_{N-1}
Y_0 Y_1 $\cdots$ Y_{N-1}
L_1 R_1
L_2 R_2
$\vdots$
L_Q R_Q

Output of the Sample Grader

The sample grader outputs the return value of function `max_prize` in one line to the standard output, after every call of this function.

Constraints

- $2 \leq N \leq 200\,000$.
- $1 \leq Q \leq 500\,000$.
- $0 \leq X_i \leq 10^9$ $(0 \leq i \leq N-1)$.
- $0 \leq Y_i \leq 10^9$ $(0 \leq i \leq N-1)$.
- $0 \leq L_j < R_j \leq N-1$ $(1 \leq j \leq Q)$.
- $R_j - L_j + 1$ is even $(1 \leq j \leq Q)$.
- Given values are all integers.

Subtasks

1. (16 points) $Q \le 100$, $X_i \le 100$, $Y_i \le 100$ $(0 \le i \le N-1)$, $R_j - L_j + 1 \le 100$ $(1 \le j \le Q)$.
2. (16 points) $Q \le 100$, $R_j - L_j + 1 \le 100$ $(1 \le j \le Q)$.
3. (19 points) $Q \le 200\,000$, $L_j \le L_{j+1}$, $R_j \le R_{j+1}$ $(1 \le j \le Q-1)$.
4. (12 points) $N \le 20\,000$, $Q \le 50\,000$.
5. (14 points) $N \le 100\,000$, $Q \le 200\,000$.
6. (23 points) No additional constraints.

Sample Communication

Here is a sample input for the sample grader and corresponding function calls.

Sample Input 1	Sample Output 1

`5 3`
`2 1 3 1 0`
`1 1 0 2 0`
`0 3`
`1 4`
`2 3`
	`2`
	`0`
	`2`

Function Calls	Return Values
`init(5, 3, [2, 1, 3, 1, 0], [1, 1, 0, 2, 0])`	
`max_prize(0, 3)`	2
`max_prize(1, 4)`	0
`max_prize(2, 3)`	2

In the first call to `max_prize`, bags 0, 1, 2, 3 are used. If participants draw balls in the following way, the total number of prizes obtained will be 2:

- The first participant draws a red ball, a blue ball, a red ball, and a blue ball from bags 0, 1, 2, 3, respectively. Since the number of red and blue balls drawn is equal, they receive one prize.
- The second participant draws a blue ball, a red ball, a red ball, and a blue ball from bags 0, 1, 2, 3, respectively. Since the number of red and blue balls drawn is equal, they receive one prize.
- At this point, bag 1 becomes empty, and the lottery event ends.

It is not possible for participants to obtain more than 2 prizes. Therefore, the first call to `max_prize` must return 2.

In the second call to `max_prize`, bags $1, 2, 3, 4$ are used. Since bag 4 is empty from the beginning, the lottery event ends without any participant drawing a ball. Therefore, the second call to `max_prize` must return 0.

In the third call to `max_prize`, bags $2, 3$ are used. If participants draw balls in the following way, the total number of prizes obtained will be 2:

- The first participant draws a red ball and a red ball from bags 2 and 3, respectively. Since the number of red and blue balls drawn is not equal, they do not receive a prize.
- The second participant draws a red ball and a blue ball from bags 2 and 3, respectively. Since the number of red and blue balls drawn is equal, they receive one prize.
- The third participant draws a red ball and a blue ball from bags 2 and 3, respectively. Since the number of red and blue balls drawn is equal, they receive one prize.
- At this point, bags 2 and 3 become empty, and the lottery event ends.

It is not possible for participants to obtain more than 2 prizes. Therefore, the third call to `max_prize` must return 2.

This sample input satisfies the constraints of subtasks 1, 2, 4, 5, and 6.

Sample Input 2	Sample Output 2

`6 5`
`1 3 3 2 1 0`
`1 2 1 1 2 1`
`0 1`
`1 2`
`1 4`
`2 5`
`4 5`
	`2`
	`3`
	`3`
	`1`
	`1`

Function Calls	Return Values
`init(6, 5, [1, 3, 3, 2, 1, 0], [1, 2, 1, 1, 2, 1])`	
`max_prize(0, 1)`	2
`max_prize(1, 2)`	3
`max_prize(1, 4)`	3
`max_prize(2, 5)`	1
`max_prize(4, 5)`	1

This sample input satisfies the constraints of all subtasks.