# Spring 2022 mBIT Advanced Division

May 22, 2022


This contest is themed around our imaginary amusement park, mBIT Land, and you are the main character! A key feature of mBIT Land is it's main street - Blair Boulevard. Good luck and have fun at the park!

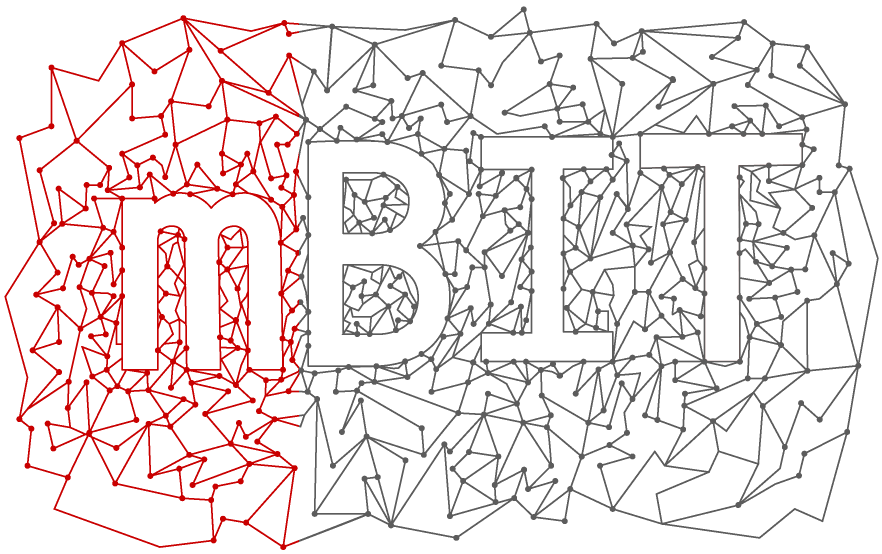
- The mBIT Team

P.S. Our art team retired :(

The problems **are not ordered by difficulty**. To gauge difficulty, check out the live scoreboard.

## Contents





# Program Specifications

Every problem has a memory limit of 256 MB and time limit of 2s, unless otherwise stated. The time and memory limits are constant across different languages.

# Advice

**Understand the pretests.** When you submit code during the contest, it will be run on 10 pretests and your verdict for each pretest will be displayed. The first pretest will always be the sample but you will not be able to access the data for any other pretest. The pretests have varying sizes but are together designed to cover the full range of input.

**Understand the system tests.** If your code passes all 10 pretests, shortly after the round ends your submission will be judged on 40 system tests. You will be considered to have solved the problem if and only if your code passes all 40 system tests.

**Understand the scoring system.** All problems are worth the same amount. Your team's score is the number of problems you fully solve. Your team's rank is determined firstly by your score and secondly by the time of your last correct submission. There is no penalty for wrong submissions.

**Use fast I/O.** For problems with large input sizes, you may want to use fast I/O methods to prevent a time limit error. Here is how to use fast I/O in each language:

- In Python, write `from sys import stdin, stdout` at the top of your program. When reading input, use `stdin.readline()`. To write output, use `stdout.write()`.
- In Java, use a custom Scanner class as shown here.
- In C++, write `ios_base::sync_with_stdio(false); cin.tie(NULL);` at the top of your `main` method. Then you can use `cin` and `cout` as usual. Printing a single newline character (`\n`) is faster than `endl`.

**Special considerations for Python.** Make sure you're using fast I/O methods as described above. You can increase the recursion limit if your functions use repeated recursion. Additionally, we strongly recommend submitting your solutions in **PyPy**, which is typically faster than Python. Make sure you strip your input of trailing white space (this is especially important for our grader). Finally, `exit()` and `quit()` do not work with our grader.

**Language versions.** Our grader uses C++ 17 (compiled with gcc 10 using `-Ofast` and `-march=native`), Java 17, and a PyPy version that implements Python 3.7.

**Ask for clarifications!** If you are confused about a problem statement, do not hesitate to message us. Message us in our discord here.

# §1 Boardwalk

You're creating a 2D boardwalk of length $N$ meters for mBIT Land. The $i$th section currently is of height $H_i$ meters. You don't want to be liable for visitors' injury, so you want to ensure a level boardwalk.

Luckily, there are $K$ construction workers you can hire to help you. The $i$th ($1 \le i \le K$) worker can increase the heights of sections $L_i, \ldots, R_i$ by one meter each, and charges $C_i$ in doing so. You may hire each worker an unlimited number of times, paying their requested charge each time.

Are you able to make $H_1 = \ldots = H_N$? If so, find the minimum cost.

**Constraints:**

- $1 \le N \le 200$
- $1 \le H_i \le 10^5$
- $1 \le K \le \frac{N(N+1)}{2}$
- $1 \le L_i \le R_i \le N$ $(1 \le i \le K)$
- $(L_i, R_i) \ne (L_j, R_j)$ $(1 \le i < j \le K)$
- $1 \le C_i \le 10^9$ $(1 \le i \le K)$

**Input:**

$N$ $K$
$H_1$ $\ldots$ $H_N$
$L_1$ $R_1$ $C_1$
$\vdots$
$L_K$ $R_K$ $C_K$

**Output:**

Output one line with the minimum cost or `-1` if it's impossible.

**Sample Input:**

```
4 4
4 1 1 2
2 3 2
2 4 3
3 4 2
2 2 2
```

**Sample Output:**

```
8
```

You could hire the first worker once and the second worker twice for a total cost of $1 \cdot 2 + 2 \cdot 3$. It can be shown that 8 is the minimum possible cost.

## §2 Double Deletion

From the corner of your eye you spot a directed graph with $N$ nodes and $M$ directed edges. The last thing you want to think about during your stay at mBIT Land is a graph, so you use your superpower *double deletion* to destroy its edges.

Double deletion allows you to simultaneously delete two edges that either both go in or out of a node. However, you can only use it for certain nodes and directions. In particular, you're given $K$ pairs of the form `(X,in)` or `(X,out)`, meaning you can perform a double deletion that involves deleting 2 edges going in/out of node $X$.

What's the maximum number of double deletions you can perform? Also, provide a maximal sequence of double deletions.

**Constraints:**

- $1 \le N \le 2 \cdot 10^5$
- $1 \le M \le \min(2 \cdot 10^5, N(N-1))$
- $0 \le K \le 2N$
- $1 \le X_i \le N$ $(1 \le i \le K)$
- $dir_i$ is either `in` or `out` $(1 \le i \le K)$
- $1 \le U_i, V_i \le N$ $(1 \le i \le M)$
- $U_i \ne V_i$ $(1 \le i \le M)$
- There are no multiedges or repeated restrictions

**Input:**

$N$ $M$ $K$
$X_1$ $dir_1$
$\vdots$
$X_K$ $dir_K$
$U_1$ $V_1$
$\vdots$
$U_M$ $V_M$

**Output:**

Print $D$, the maximum number of deletions. Then print $D$ pairs of distinct edge indices. You can print the pairs and the edges in each pair in any order.

$D$
$A_1$ $B_1$
$\vdots$
$A_D$ $B_D$

**Sample Input:**

```
5 6 5
1 in
1 out
4 in
4 out
3 out
2 1
4 1
1 3
2 4
4 3
5 4
```

**Sample Output:**

```
2
2 5
4 6
```

We can perform double deletion on pairs of edges (2, 5) and (4, 6). Alternatively, we can perform double deletion on pairs of edges (1, 2) and (4, 6). We cannot perform more than 2 double deletions.

# §3 Mascot Parade

The mascots of mBIT Land are having a parade! They are all in vehicles and are traveling down Blair Boulevard, waving and blowing kisses to happy children.

There $N$ mascots are numbered 1 through $N$ in increasing order of position. The $i$th $(1 \leq i \leq N)$ mascot has a speed $S_i$ and an initial position $P_i$. This means that every second they have the potential to advance $S_i$ meters, and at second 0 they are at position $P_i$ meters down the road.

The road is very narrow, which means the mascots can't pass each other. Instead, if $i$ is travelling faster than $i+1$, then on the second in which mascot $i$'s position would be greater than or equal to mascot $i+1$'s position, mascot $i$ instead gets held back and will lag exactly one meter behind mascot $i+1$ and travel at the same speed of mascot $i+1$ for the rest of the parade.

The children can only come to the parade at certain times and only want to see a certain mascot. There are $Q$ children and child $i$ wants to know how far down the road mascot $X_i$ will be at time $Y_i$. You have to answer the children's queries in order or else they will get upset (see Input for details).

Help the parade run smoothly by telling the children where their favorite mascots will be!

**Constraints:**

- $1 \leq N, Q \leq 2 \cdot 10^5$
- $1 \leq P_i, S_i \leq 10^9$ $(1 \leq i \leq N)$
- $P_i < P_{i+1}$ $(1 \leq i \leq N-1)$
- $0 \leq A_i \leq N-1$ $(1 \leq i \leq Q)$
- $0 \leq B_i \leq 10^9 - 1$ $(1 \leq i \leq Q)$

**Input:**

$N$ $Q$
$P_1 \ldots P_N$
$S_1 \ldots S_N$
$A_1$ $B_1$
$\vdots$
$A_Q$ $B_Q$

Let $ans$ be the answer to the previous query or 0 if there are no previous queries. The $i$th query $(1 \leq i \leq N)$ is decoded as follows:

$$X_i = (ans + A_i)\%N + 1$$
$$Y_i = (ans + B_i)\%10^9 + 1$$

**Output:**

Output $Q$ lines with the answers to the queries.

**Sample Input:**

```
7 4
3 4 7 9 11 12 20
2 5 1 8 7 10 4
0 3
1 999999992
2 999999976
2 999999981
```

**Sample Output:**

```
9
24
22
33
```

The input gets decoded to the following queries:

```
1 4
4 2
6 1
4 4
```

The following table shows the initial locations and speeds of the mascots.

| | $i = 1$ | $i = 2$ | $i = 3$ | $i = 4$ | $i = 5$ | $i = 6$ | $i = 7$ |
|---|---|---|---|---|---|---|---|
| $S_i$ | 2 | 5 | 1 | 8 | 7 | 10 | 4 |
| $P_i$ | 3 | 4 | 7 | 9 | 11 | 12 | 20 |

Below are their positions in the first 4 seconds:

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| $t = 1$ | 5 | 7 | 8 | 17 | 18 | 22 | 24 |
| $t = 2$ | 7 | 8 | 9 | 24 | 25 | 27 | 28 |
| $t = 3$ | 8 | 9 | 10 | 29 | 30 | 31 | 32 |
| $t = 4$ | 9 | 10 | 11 | 33 | 34 | 35 | 36 |

# §4 Memorable Attractions

You love visiting the shops on Blair Boulevard. There is a shop at $x = 0, \ldots, 10^{18}$ meters down the street. At day 0, you are at $x = 0$.

You started creating an itinerary, which can be represented as $P = [P_0 = 0, P_1, \ldots, P_N]$, but it's incomplete. If $P_i \neq -1$, you plan to visit the shop at $x = P_i$ on day $i$; otherwise you must decide where to go on day $i$. In other words, you must replace every $-1$ in $P$ with an integer between 0 and $10^{18}$.

You get tired easily so you can only walk up to $K$ meters between days (though your location must be between 0 and $10^{18}$ at all times). Also, at the end of some days you really miss a shop from a previous day and you'll only be cheered up if you're not that far from that spot on the road. In particular, there's $M$ moments of the form $A_i$ $B_i$ $D_i$: you must ensure $|P_{A_i} - P_{B_i}| \leq D_i$.

Can you complete your itinerary under these consideration?

**Constraints:**

- $1 \leq N \leq 2 \cdot 10^5$
- $0 \leq M \leq 2 \cdot 10^5$
- $1 \leq K \leq 10^9$
- $-1 \leq P_i \leq 10^{18}$ $(1 \leq i \leq N)$
- $0 \leq A_i, B_i \leq N$ $(1 \leq i \leq M)$
- $0 \leq D_i \leq 10^{18}$ $(1 \leq i \leq M)$

**Input:**

$N$ $K$ $M$
$P_1$ $\ldots$ $P_N$
$A_1$ $B_1$ $D_1$
$\vdots$
$A_M$ $B_M$ $D_M$

**Output:**

If it is impossible, print

`NO`

Otherwise, print your completed itinerary.

`YES`
$P_1, \ldots, P_N$

**Sample Input:**

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

**Sample Output:**

```
YES
3 2 5
```

# §5 Monkey on a Ladder

**The time limit for this problem is 3s.**

Tonight, there is a special animal show! In one of the acts, a monkey is jumping across ladders. There's $N$ ladders in a line, the $i$th having $H_i$ rungs labelled $1 \dots H_i$. If the monkey is at the rung $j$ of the $i$th ladder, its next jump must be to rung $k \leq j$ of the $(i+1)$th ladder (if it exists) *or* to the rung $j+1$ of the $i$th ladder (if it exists). The latter move takes $C_i$ seconds and the former is instantaneous.

You're wondering how quickly the monkey can finish its act. You know that the monkey is smart and always takes the fastest path, but don't know where the monkey plans to start and stop. For $Q$ queries, find the minimum time for the monkey to move from the $a_i$th rung of the $l_i$th ladder to the $b_i$th rung of the $r_i$th ladder $(1 \leq i \leq Q)$.

**Constraints:**

- $1 \leq N, Q \leq 5 \cdot 10^5$
- $1 \leq H_i, C_i \leq 10^9$ $(1 \leq i \leq N)$
- $1 \leq l_i < r_i \leq N$ $(1 \leq i \leq Q)$
- $1 \leq a_i \leq H_{l_i}$ $(1 \leq i \leq Q)$
- $1 \leq b_i \leq H_{r_i}$ $(1 \leq i \leq Q)$

**Input:**

$N$ $Q$
$H_1$ $C_1$
$\vdots$
$H_N$ $C_N$
$l_1$ $r_1$ $a_1$ $b_1$
$\vdots$
$l_Q$ $r_Q$ $a_Q$ $b_Q$

**Output:**

Output $Q$ lines.

**Sample Input:**

```
4 3
4 2
2 1
3 3
4 4
1 2 1 1
1 4 1 4
1 4 4 4
```

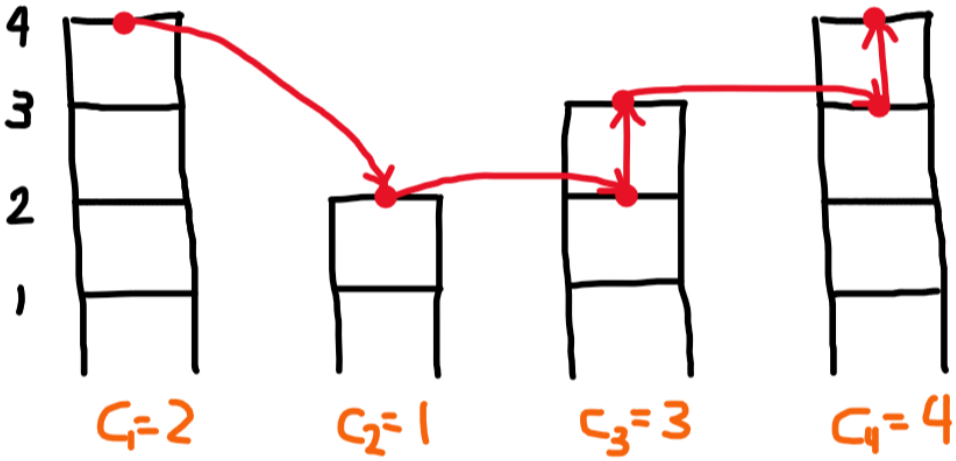
**Sample Output:**

```
0
8
7
```

For query 1, the monkey can simply move right.

For query 2, an optimal path is $(1,1) \to (2,1) \to (2,2) \to (3,2) \to (3,3) \to (4,3) \to (4,4)$. $(x,y)$ refers to rung $y$ of ladder $x$.

For query 3, an optimal path is $(1,4) \to (2,2) \to (3,2) \to (3,3) \to (4,3) \to (4,4)$. Below is an illustration of this path:



# §6 Not Nim

Your nerdy competitive programming friend is tagging along throughout your trip in mBIT Land. Your friend doesn't enjoy anything at the park and instead wants to play a game with you.

> There's $N$ piles of stones, the $i$th having $P_i$ stones. Two players alternate taking stones -

You cannot let your friend try to entice you with another combinatorial game. Your friend senses your irritation and assures you this is *Not Nim*.

> In *Not Nim*, two players alternate taking a *fixed* number of stones from any pile. Let the two players be Alice and Bob, with Alice going first. On Alice's turn, she takes exactly $A$ stones from a pile of her choice, or loses if there is no pile with at least $A$ stones. On Bob's turn, he takes exactly $B$ stones from a pile of his choice, or loses if there is no pile with at least $B$ stones.

Seeing your sustained apathy, your friend makes a compromise: if you can predict the winner for $T$ games, you don't have to play.

So, who wins each game, assuming Alice and Bob always play optimally?

**Constraints:**

- $1 \leq T \leq 2 \cdot 10^5$
- $1 \leq \sum N \leq 2 \cdot 10^5$
- $1 \leq A, B \leq 10^9$
- $1 \leq P_i \leq 10^9$ $(1 \leq i \leq N)$

**Input:**

The first line is $T$, the number of games. Each game consists of the following input:

$N$
$A$ $B$
$P_1$ ... $P_N$

**Output:**

Output $T$ lines. On the $i$th print the winner of the $i$th game, either `Alice` or `Bob`.

**Sample Input:**

```
2
3
3 2
2 4 4
4
4 4
7 2 9 12
```

**Sample Output:**

```
Bob
Bob
```

In the first game, Alice must take from a pile of 4 on her first turn. Then, Bob can take 2 from the other pile of 4 and Alice will be unable to move.

# §7 Park Passes

You are obsessed with mBIT Land and want to go on rides every day!

There are $N$ ticket packages you can buy. Package $i$ consists of $T_i$ tickets, expires after day $E_i$, and costs $C_i$ dollars. When a ticket package expires, every ticket that came with the package can no longer be used. You want to have an unexpired ticket to use for every day from day 1 to day $M$.

What is the minimum amount you need to spend to achieve your goal? If there is no way to buy packages such that you can go every day, report it.

**Constraints:**

- $1 \leq N \leq 1000$
- $1 \leq M \leq 1000$
- $1 \leq T_i \leq E_i \leq M$
- $1 \leq C_i \leq 10^9$

**Input:**

$N$ $M$
$T_1$ $E_1$ $C_1$
$\vdots$
$T_N$ $E_N$ $C_N$

**Output:**

If it is impossible to go every day, print `-1`. Otherwise, print the answer.

**Sample Input:**

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

**Sample Output:**

```
9
```

It is optimal to buy packages 1, 3, and 4.

# §8 Rating Report

The National Amusement Park Society (NAPS) is tasked with submitting mBIT Land's annual rating report. The committee consists of $N$ amusement-park-reviewers who will each submit their rating of mBIT Land.

The reviewers work in a hierarchy in which each reviewer, except reviewer 1 has a supervisor to whom they report to. Reviewer $i$'s supervisor is $P_i$ ($2 \le i \le N$). Out of the $N$ reviewers, $K$ of them have no subordinates (no one whom they are the supervisor to). Each of these $K$ reviewers will generate a first-hand rating of mBIT Land by visiting the park. Some of them have already done this, but some of them have yet to formulate their opinion, meaning you can still influence their rating. You can influence the undecided first-hand reviewers' ratings to anything so long as all $K$ first-hand ratings are in the range $[1, K]$ and pairwise distinct.

The other $N - K$ reviewers are lazy and use their subordinate's ratings to generate their own ratings. Each of these $N - K$ reviewers is either an optimist, in which case their rating is equal to the best rating out their subordinates, or a pessimist, in which case their rating is equal to the worst rating out of their subordinates. After generating their rating, each reviewer reports their rating to their supervisor. No reviewer generates their rating until each of their subordinates has generated their rating.

As input, you're given an array $A$:

- $A_i = 0$ means that $i$ is a first-hand reviewer who hasn't submitted a rating yet
- $A_i > 0$ means $i$ is a first-hand reviewer who has already submitted a rating $R_i = A_i$
- $A_i =$ `max` means that $i$ is not a first-hand reviewer, and is an optimist who submits a rating $R_i \coloneqq \max_{j \mid P_j = i} R_j$ ($j$ is a subordinate of $i$)
- $A_i =$ `min` means that $i$ is not a first-hand reviewer, and is a pessimist who submits a rating $R_i \coloneqq \min_{j \mid P_j = i} R_j$ ($j$ is a subordinate of $i$)

Ultimately, the manager (reviewer 1) submits $R_1$ as the park's final rating. As an avid supporter of mBIT Land, find the maximum possible final rating you can achieve by influencing the first-hand reviewers optimally.

**Constraints:**

- $2 \le N \le 2 \cdot 10^5$
- $1 \le K \le N$
- $1 \le P_i \le i - 1$ ($2 \le i \le N$)
- The tree formed has $K$ leaves (first-hand reviewers)
- $A_i$ is an integer between 0 and $K$ if $i$ is a leaf
- $A_i \ne A_j$ if $i$ is a leaf, $j$ is a leaf, $A_i > 0$, and $A_j > 0$
- $A_i$ is one of {`max`, `min`} if $i$ is not a leaf

**Input:**

You are given $N$, $K$, the array $P$, and the array $A$.

$N\ K$
$P_2\ \ldots\ P_N$
$A_1$
$\vdots$
$A_N$

**Output:**

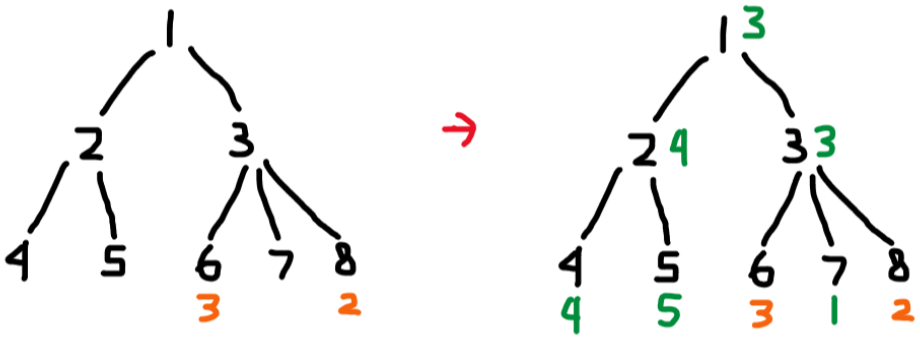
Output one line with maximum possible $R_1$.

**Sample Input:**

```
8 5
1 1 2 2 3 3 3
min
min
max
0
0
3
0
2
```

**Sample Output:**

```
3
```




One optimal assignment of first-hand ratings is:

$$R_4 = 4$$
$$R_5 = 5$$
$$R_7 = 1$$

# §9 Snapshot

You are operating the security cameras at mBIT Land and have been alerted of possible tampering with the security photos!

There were $N$ visitors walking around mBIT Land today. Each visitor was either walking right or left at a constant velocity of either 1 or $-1$, but you don't know which. You do know where each visitors started the day, represented by an increasing list $X$ of length $N$ where $X_i$ is the initial location of visitor $i$ (at time $t = 0$).

The only other information you have is $K$ time-stamped photos. The $i$th was taken at time $T_i$ and is described by $L_i$, a non-decreasing list of length $N$ where $L_{i,j}$ is the location of a visitor in photo $i$. In other words, each picture consists of the $N$ visitors' collective locations at a known time.

Unfortunately, your cameras are blurry and you can't decipher who is who across the photos. Can you deduce whether the photos have been tampered with? If the photos are consistent find the initial velocity of each visitor. If one of them has been tampered with, report it.

**Constraints:**

- $1 \leq N \cdot K \leq 2 \cdot 10^5$
- $1 \leq T_i \leq 10^9$
- $T_i < T_{i+1}$ $(1 \leq i \leq K-1)$
- $-10^9 \leq X_i \leq 10^9$ $(1 \leq i \leq N)$
- $X_i < X_{i+1}$ $(1 \leq i \leq N-1)$
- $-2 \cdot 10^9 \leq L_{i,j} \leq 2 \cdot 10^9$ $(1 \leq i \leq K,\ 1 \leq j \leq N)$
- $L_{i,j} \leq L_{i,j+1}$ $(1 \leq i \leq K,\ 1 \leq j \leq N-1)$

**Input:**

$N\ K$
$X_1\ \ldots\ X_N$
$T_1$
$L_{1,1}\ \ldots L_{1,N}$
$\vdots$
$T_K$
$L_{K,1}\ \ldots L_{K,N}$

**Output:**

If the photos have been tampered with, print

0

Otherwise, print the velocities of the visitors

$V_1\ \ldots\ V_N$

**Sample Input:**

```
5 3
1 3 6 7 11
2
-1 4 5 5 13
5
-4 1 2 8 16
7
-6 -1 0 10 18
```

**Sample Output:**

```
-1 1 -1 -1 1
```

# §10 Sugar Rush

You are addicted to cotton candy!

There are $N$ cotton candy stands along Blair Boulevard. The $i$th stand is located at $X_i$ meters along the park. No two stands are at the same location.

You start by visiting one of the cotton candy stands. After the visiting a stand, you recalculate which unvisited stand is closest to your current position and rush to that stand. If there are multiple closest stands, you go to the one with the highest coordinate.

Being indecisive, you don't know which stand you should visit first. For each $i = 1, \ldots, N$, find the total distance you will travel to visit every cotton candy stand if you started at stand $i$.

**Constraints:**

- $2 \leq N \leq 2 \cdot 10^5$
- $1 \leq X_i \leq 10^9$ $(1 \leq i \leq N)$
- $X_i < X_{i+1}$ $(1 \leq i \leq N-1)$

**Input:**

$N$
$X_1 \ \ldots \ X_N$

**Output:**

Output $N$ lines with the respective answers.

**Sample Input:**

```
7
2 5 6 10 12 20 22
```

**Sample Output:**

```
20
37
24
32
30
22
20
```