Delivery Detour

Mutanguha runs a delivery-robot service on a city grid. A robot starts at $(0, 0)$ and is already programmed with a list of N instructions $S[0], S[1], \ldots, S[N-1]$. Each instruction moves the robot one unit: from (x, y), the instruction `L` moves it to $(x-1, y)$, `R` to $(x+1, y)$, `D` to $(x, y-1)$, and `U` to $(x, y+1)$.

The robot's firmware allows only one kind of change: cancelling **one contiguous block** of instructions, possibly empty. In other words, pick two integers l and r such that $0 \le l \le r \le N$ and replace S with $S[0], S[1], \ldots, S[l-1], S[r], S[r+1], \ldots, S[N-1]$. The robot then executes the remaining instructions in their original order. The manufacturer charges a cancellation fee for every cancelled instruction, depending on its direction: C_L for `L`, C_R for `R`, C_D for `D`, and C_U for `U`.

Just before departure, Mutanguha learns that the customer is actually waiting at (X, Y).

Help Mutanguha find the minimum total fee so that the robot ends at (X, Y), or determine that no block of cancellations achieves this.

Implementation Details

You should implement the following procedure:

```
int64 minimum_fee(string S, int32 X, int32 Y, int32 CL, int32 CR,
                  int32 CD, int32 CU)
```

- S: a string of length N over the characters `LRDU`, the programmed instructions.
- X, Y: the customer's location.
- CL, CR, CD, CU: the cancellation fee of a single `L`, `R`, `D`, `U` instruction, respectively.
- This procedure is called exactly once.
- This procedure should return the minimum total fee so that the robot ends at (X, Y), or -1 if this is impossible.

Constraints

- $1 \le N \le 200\,000$
- $-N \le X, Y \le N$
- $0 \le C_L, C_R, C_D, C_U \le 10^9$

Subtasks

Subtask	Score	Additional Constraints
1	15	$N \leq 2\,000$
2	10	$C_L = C_R = C_D = C_U = 0$
3	20	S contains only `R` and `U`
4	20	S contains only `L` and `R`, and $Y = 0$
5	15	$C_L = C_R = C_D = C_U = 1$
6	20	No additional constraints.

Examples

Example 1

Consider the following call:

```
minimum_fee("RUURLLL", -1, 0, 2, 3, 3, 4)
```

Executing all seven instructions moves the robot to $(-1, 2)$, which misses the customer located at $(-1, 0)$.

If Mutanguha cancels the block `UU`, the robot executes `R`, `R`, `L`, `L`, `L` and reaches the target destination at $(-1, 0)$. The fee for this cancellation is $C_U + C_U = 8$.

While cancelling the larger block `UURL` also places the robot at $(-1, 0)$, its total fee is higher: $4 + 4 + 3 + 2 = 13$. Because no cancellation sequence yields the correct destination at a cost below 8, `minimum_fee` should return 8.

Example 2

Consider the following call:

```
minimum_fee("RR", 0, 1, 1, 1, 1, 1)
```

No block of cancellations can make the robot end at $(0, 1)$, as the robot never moves vertically. Therefore `minimum_fee` should return -1.

Example 3

Consider the following call:

```
minimum_fee("LU", -1, 1, 7, 7, 7, 7)
```

The robot already ends at $(-1, 1)$, so Mutanguha should cancel the empty block and pay nothing. Hence `minimum_fee` should return 0.

Sample Grader

Input format:

```
N X Y
CL CR CD CU
S
```

Output format:

```
F
```

Here F denotes the return value of `minimum_fee`.