Let's define make x operations in position y to add x to A_y, substract x to A_{y+1} and add x to A_{y+2}.

Note that the following greedy algorithm will always produce the correct answer:

Start from $i = 1$, and apply $B_i - A_i$ operations on position i, update next two positions according to that.

Let us denote the number of operations that are made to the position x in the previous algorithm, as $f(x)$, if $f(x) > 0$ addition operations are made, otherwise it is subtraction.

Let's analyze how $f(x)$ behaves during the algorithm, we can say that:

$f(x) = B_x - A_x + f(x-1) - f(x-2)$

And from A, B can be reached if and only if $f(n-1)$ and $f(n)$ are equal to 0.

Now let's analyze what happens to f when we add 1 to A_x, assuming that before we had the $f(i)$ as follows:

```
0 0 0 0 0 0 0 0 0 0 0 0
```

Now we would have the following:

```
-1 -1 0 1 1 0 -1 -1 0 1 1 0
```

It can be noted that $f(y)$ will increase or decrease the same as $f(x)$ if $y - x = 0 \mod 6$ or $y - x = 1 \mod 6$, will increase or it will decrease in the opposite of $f(x)$ if $y - x = 3 \mod 6$ or $y - x = 4 \mod 6$ and will not change if $y - x = 2 \mod 6$ or $y - x = 5 \mod 6$.

With the following we can keep $f(n-1)$ and $f(n)$ efficiently making updates of adding or subtracting x to a position.

Now to range updates:

We can observe that if x is added to a range of size 6, only the f of them 6 change, those on the right remain the same, as shown:

```
-1 -1  0  1  1  0 -1 -1  0  1  1  0
   -1 -1  0  1  1  0 -1 -1  0  1  1
      -1 -1  0  1  1  0 -1 -1  0  1
         -1 -1  0  1  1  0 -1 -1  0
            -1 -1  0  1  1  0 -1 -1
               -1 -1  0  1  1  0 -1
                  -1 -1  0  1  1  0
```

So any size change 6, which does not contain $n-1$ or n is negligible, and we can reduce an update from (l, r, x) to $(l+6, r, x)$ while possible, and when it is an update of size less than or equal to 6, we solve it as point updates.