Subtarea 1:

Analicemos el siguiente algoritmo, mientras A no sea igual a B, buscamos el primer índice en que difieren, llamémoslo x, si $x \geq n-1$, entonces no hay solución, de lo contrario, si $A_i < B_i$, aplicamos una operación para sumarle 1 a A_i, restarle 1 a A_{i+1} y sumarle 1 a A_{i+2}, y si $A_i > B_i$, aplicamos una operación para restarle 1 a A_i, sumarle 1 a A_{i+1} y restarle 1 a A_{i+2}.

La implementación directa de lo anteriormente explicado es suficiente para pasar la primera subtarea.

Subtarea 2:

El algoritmo de la subtarea anterior puede ser optimizado de la siguiente forma, en orden ascendente, para cada i de 1 a $n-2$ si $A_i < B_i$ hacemos $B_i - A_i$ operaciones de sumarle 1 a A_i, restarle 1 a A_{i+1} y sumarle 1 a A_{i+2}, si $A_i > B_i$ aplicamos $A_i - B_i$ operaciones de restarle 1 a A_i, sumarle 1 a A_{i+1} y restarle 1 a A_{i+2}. Si al terminar este algoritmo, ambos arreglos son iguales la respuesta es YES, de lo contrario es imposible.

Complejidad $O(n)$

Subtarea 3:

Denotemos la cantidad de operaciones que se le hacen a la posición x en el algoritmo anterior, como $f(x)$, si $f(x) > 0$ es que se le hacen operaciones de suma, de lo contrario de resta.

Analicemos como se comporta $f(x)$ durante el algoritmo, podemos decir que:

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

Y del arreglo A se puede llegar al arreglo B si y solo si $f(n-1)$ y $f(n)$ son iguales a 0
.

Ahora analicemos que ocurre con f cuando sumamos 1 a A_x, suponiendo que antes tuviéramos los $f(i)$ de la siguiente manera:

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

Ahora tendríamos lo siguiente:

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

Se puede notar que $f(y)$ aumentará o disminuirá lo mismo que $f(x)$ si $y - x = 0 \mod 6$ o $y - x = 1 \mod 6$, aumentará o disminuirá en el opuesto que $f(x)$ si $y - x = 3 \mod 6$ o $y - x = 4 \mod 6$ y no cambiará si $y - x = 2 \mod 6$ o $y - x = 5 \mod 6$
.

Con lo siguiente podemos mantener $f(n-1)$ y $f(n)$ eficientemente haciendo updates de sumar o restar x a una posición.

Complejidad $O(n+q)$

Subtarea 4

Podemos obseervar que si se le suma x a un rango de tamaño 6, solo cambian las f de ellos 6, las de la derecha permanecen igual, como se muestra:

```
-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
```

Entonces cualquier cambio de tamaño 6, que no contenga a $n-1$ o a n es insignificante, y podemos reducir un update de (l, r, x) a $(l+6, r, x)$ mientras se pueda, y cuando sea un update de tamaño menor o igual a 6, lo resolvemos como en la subtarea anterior.

Complejidad $O(n+q)$