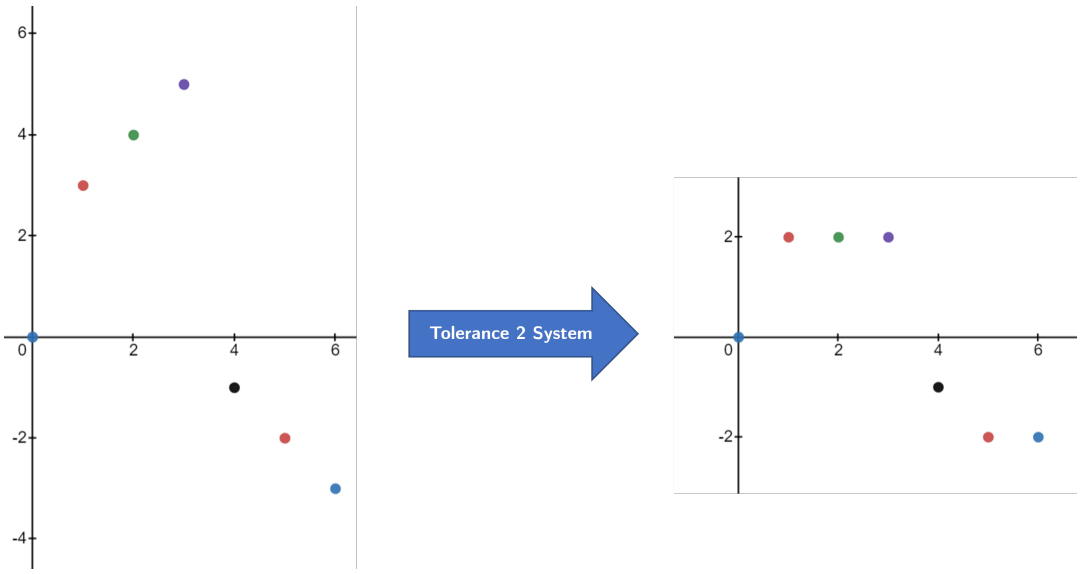
Plane Turbulences (`turbulences`)

Brianair, una aerolinea con mucha reputación, está realizando un estudio sobre turbulencias. Están trabajando para buscar un sistema óptimo de estabilización de su avión. Este sistema tendrá que estar activo durante todo el vuelo excepto en el despegue y el aterrizaje, o sea activado en la parte que el avión tendría que volar *en línea recta*.

Un sistema de estabilización de *tolerancia* x asegurará que el avión no se desvíe de la altura deseada (la que tendría si volara en línea recta a altura constante) con una diferencia absoluta superior a x. Es posible saber de antemano la altura de cada minuto del viaje si no equipamos el sistema de estabilización. Se te dará todas estas predicciones de las desviaciones de las alturas $A_0, \ldots, A_{N-1}$ para la duración del viaje N, en orden cronológico.

El siguiente ejemplo muestra como el sistema de estabilización de tolerancia 2 trabaja con las predicciones de las desviaciones $A_0 = 0, A_1 = 3, A_2 = 4, A_3 = 5, A_4 = -1, A_5 = -2, A_6 = -3$ para volar con las siguiente desviaciones $B_0 = 0, B_1 = 2, B_2 = 2, B_3 = 2, B_4 = -1, B_5 = -2, B_6 = -2$.



Alturas antes y después de aplicar el sistema de tolerancia 2.

Brianair sabe que sus usuarios les encanta volar alto, o sea que la satisfacción de un cliente después de un vuelo con sistema de estabilización x es igual a $\sum_{i=0}^{N-1} B_i$, donde B_i es la altura estabilizada en el tiempo i. Eso es, $B_i = \text{sign}(A_i) \cdot \text{mín}(|A_i|, x)$.

Pero el coste de sobornar a los reguladores para permitir un sistema de tolerancia x es igual a Kx, donde K es una constante no negativa. La aerolínea por lo tanto quiere maximizar su beneficio para el vuelo, $\left(\sum_{i=0}^{N-1} B_i\right) - Kx$.

Dados K y $A_0, \ldots, A_{N-1}$, ¿serías capaz de encontrar el beneficio máximo que se puede obtener de poner el sistema de tolerancia óptimo $x \geq 0$?

Implementación

Tienes que mandar un solo archivo `.cpp`.

☞ Entre los adjuntos de esta tarea, tendras un archivo template `turbulences.cpp` con un ejemplo de implementación.

Tienes que implementar la siguiente función:

C++
```
long long revenue(int N, int K, vector<long long> A);
```

- Entero N representando la duración del vuelo.
- Entero K representando el coeficiente del coste.
- Un vector A, indexado de 0 a $N-1$, conteniendo los valores $A_0, A_1, \ldots, A_{N-1}$, donde A_i es la predicción de la altura en el momento i.
- La función debe devolver el máximo beneficio que se puedes sacar.

El grader llamará la función `revenue` e imprimirá el valor en el archivo de salida.

Sample Grader

En el directorio de la tarea hay una versión simplificada del grader, que puedes usar para ejecutar tu solución en local. El grader simplificado lee el input de `stdin`, llama las funciones que tienes que implementar, y finalmente escribe la salida en `stdout`.

La entrada está formada por $N+1$ líneas, que contienen:

- Línea 1: el entero N y el entero K.
- Línea $2+i$ $(0 \leq i < N)$: el entero A_i.

La salida está hecha de una sola línea, conteniendo el valor devuelto por la función `revenue`.

Restricciones

- $1 \leq N \leq 2 \times 10^5$.
- $0 \leq K \leq 2 \times 10^5$.
- $-10^{12} \leq A_i \leq 10^{12}$.

Puntuación

Tu programa será probado con casos de pruebas agrupados en subtareas. Para obtener la puntuación en cada subtarea, tienes que resolver correctamente todos los juegos de prueba que la forman.

- **Subtask 1** [**0 puntos**]: Sample test cases.
- **Subtask 2** [**15 puntos**]: $N = 1$.
- **Subtask 3** [**30 puntos**]: $N \leq 10^2$, $K \leq 10^2$, $-10^2 \leq A_i \leq 10^2$ para cada $i = 0, \ldots, N-1$.
- **Subtask 4** [**17 puntos**]: Todos los A_i son iguales.
- **Subtask 5** [**18 puntos**]: Todos los A_i son no negativos.
- **Subtask 6** [**20 puntos**]: Sin restricciones adicionales.

Ejemplos de entrada/salida

stdin	stdout
7 1 0 3 4 5 -1 -2 -3	1
5 1 7 8 -2 5 -10	3
5 0 1000000000000 1000000000000 1000000000000 1000000000000 1000000000000	5000000000000

Explicación

In the **first sample case**, the situation is as described in the picture above. The optimal revenue is obtained with $x = 5$. En el **primer caso de pruebas**, la situación se describe como en la imagen superior. El beneficio óptimo se obtiene cuando $x = 5$.

En el **segundo caso de pruebas**, el beneficio óptimo se obtiene configurando $x = 5$. Así el beneficio obtenido es de $(5 + 5 + -2 + 5 + -5) - 1 \cdot 5 = 3$.

En el **tercer caso de pruebas**, el beneficio óptimo se obtiene configurandolo con cualquier valor $x \geq 10^{12}$.