Explicación del Problema: Worms
Topo ha llegado su hora de comer, pero como es ciego, su amigo Marmota le propone un juego.
Marmota organiza N montones de gusanos, donde:

- El i-ésimo montón contiene a_i gusanos.
- Todos los gusanos son etiquetados de forma consecutiva, comenzando desde 1.

Es decir:

- El primer montón contiene los gusanos etiquetados desde 1 hasta a_1.
- El segundo montón contiene desde $a_1 + 1$ hasta $a_1 + a_2$.
- Y así sucesivamente para los N montones.

Marmota quiere probar a Topo: le dirá la etiqueta de ciertos "gusanos jugosos" y Topo debe decir a qué montón pertenece cada gusano. Marmota solo le entregará el gusano si responde correctamente.
Entradas del Problema

1. n → número de montones ($1 \le n \le 10^5$).
2. $a_1, a_2, \ldots, a_n$ → cantidad de gusanos en cada montón.
3. m → número de consultas o gusanos jugosos ($1 \le m \le 10^5$).
4. $q_1, q_2, \ldots, q_m$ → cada q representa una etiqueta de gusano que Topo debe localizar.

Salida del Problema
Para cada consulta q_i, se debe imprimir el número del montón al que pertenece el gusano con esa etiqueta.
Ejemplo Explicativo
Supongamos que existen 3 montones:

- Montón 1: 3 gusanos
- Montón 2: 2 gusanos
- Montón 3: 4 gusanos

Las etiquetas quedarían así:

- Montón 1 → 1 a 3
- Montón 2 → 4 a 5
- Montón 3 → 6 a 9

Y Marmota le dice a Topo que los gusanos jugosos son: 1, 4 y 9.
Para determinar en qué montón está cada etiqueta, podemos construir un arreglo de sumas parciales (prefix sum):

Montón	Cantidad	Suma Parcial
1	3	3
2	2	5
3	4	9

Cada valor de la suma parcial indica el último número de etiqueta que corresponde a ese montón.
Así:

- 1 → está en el primer montón
- 4 → está en el segundo montón
- 9 → está en el tercer montón

Solución 1: Búsqueda Lineal (método cándido)
La primera solución consiste en recorrer los montones desde el inicio hasta encontrar el primero cuya suma parcial sea mayor o igual que la etiqueta buscada.
Por ejemplo, para la etiqueta 4:

- ¿$4 \le 3$? → No
- ¿$4 \le 5$? → Sí → está en el montón 2

Este método es correcto, pero tiene un problema:
Si n y m son muy grandes (hasta 100 000), esta búsqueda lineal puede volverse lenta.
Complejidad:
$O(n \times m)$
Es decir, en el peor caso debemos revisar todos los montones por cada consulta.
Solución 2: Búsqueda Binaria (método óptimo)
La identidad clave aquí es que el arreglo de sumas parciales está ordenado crecientemente, lo que nos permite aplicar búsqueda binaria.
El proceso es:

1. Construimos el mismo arreglo de sumas parciales.
2. Para cada etiqueta q, buscamos la primera posición del prefix sum que sea $\ge q$.
3. Esa posición corresponde directamente al número del montón.

Ejemplo para q = 4:

- Valores: [3, 5, 9]
- Buscamos el primer elemento ≥ 4
- Se encuentra en la posición 2 → Montón 2

La búsqueda binaria reduce drásticamente el tiempo.

Complejidad:

O(mlogn)

Esto la convierte en la solución ideal para el límite de datos del problema.

Conclusión

Ambas soluciones son correctas, pero la primera solo es viable para casos pequeños.

La búsqueda binaria aprovecha el orden natural del arreglo de sumas parciales y permite resolver el problema de forma eficiente, incluso en las peores condiciones.