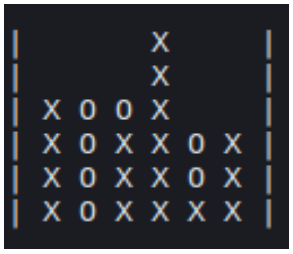
Dado un array de números enteiros maiores ou iguais que cero, onde cada un representa unidades de bloques amontoados, calcular cantas unidades de auga quedarán atrapadas entre eles.

- Exemplo: Dado o array *[4, 0, 3, 6, 1, 3]*.



Lenda
- *X: Bloque*
- *O: Auga*
- *| : Límite espazo debuxado. Sen incidencia.*

Representando bloque con X e auga con O, quedarán atrapadas 7 unidades de auga. Supoñemos que existe un chan impermeable na parte inferior que retén a auga. O carácter | representa os límites do espazo de debuxado pero non ten incidencia en ningún cálculo.

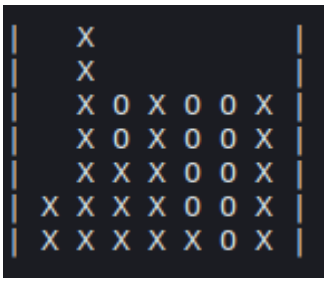
Características do array:
- Formado por números enteiros maiores ou iguais a cero.
- A lonxitude do array debe ser maior que 2.

Solicítase: Creación de método/función *calcularUdsAuga* que reciba un array de bloques válido e devolva un enteiro coas unidades de auga que quedarán atrapadas.

Cabeceiras das funcións:
- Java: *public static int calcularAuga(int[] bloques)*
- Python: *def calcularAuga(bloques)*

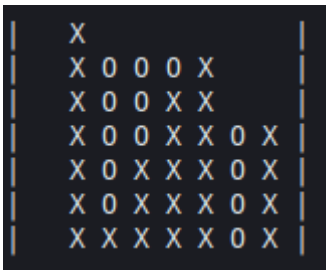
Casos de exemplo (X representa Bloques e O representa auga):

Caso a: *[2, 7, 3, 5, 1, 0, 5]*



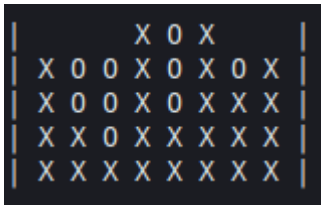
Total auga acumulada: 11

Caso b: *[0, 7, 1, 3, 5, 6, 0, 4]*



Total auga acumulada: 13

Caso c: *[4, 2, 1, 5, 2, 5, 3, 4]*



Total auga acumulada: 9