一、先想清楚：优惠券什么时候最有用？

每张优惠券在买每个商品时都能优惠 1 元，而且优惠券永久有效。

也就是说，假设现在手里有 x 张优惠券，那么买一个价格为 a 的商品时，需要支付：

$$\max(a-x,0)$$

买完这个商品后，还会再获得一张优惠券。

所以越晚买的商品，能用的优惠券越多。

因此，一个非常自然的策略是：

便宜的商品先买，贵的商品后买。

因为后面的商品能享受到更多优惠券，更应该把更贵的商品放到后面。

二、为什么一定可以按价格从小到大买？

考虑两个相邻购买的商品，价格分别是 x 和 y，且 $x > y$。

假设买第一个时有 c 张优惠券，那么买第二个时有 $c+1$ 张优惠券。

如果先买贵的 x，再买便宜的 y，花费是：

$$\max(x-c,0)+\max(y-(c+1),0)$$

如果交换顺序，先买便宜的 y，再买贵的 x，花费是：

$$\max(y-c,0)+\max(x-(c+1),0)$$

后者不会更大。

直观理解就是：

多出来的那一张优惠券应该优先给更贵的商品使用。

所以如果购买顺序中出现了“贵的在前、便宜的在后”，交换它们不会让答案变差。

不断交换后，就可以得到一个最优顺序：

所有商品按价格从小到大购买。

三、把题目化简

现在假设商品已经排序：

$$a_1 \le a_2 \le \cdots \le a_n$$

如果一开始不买优惠券，那么：

- 买第 1 个商品时，有 0 张额外获得的优惠券；
- 买第 2 个商品时，有 1 张额外获得的优惠券；
- 买第 3 个商品时，有 2 张额外获得的优惠券；
- $\cdots$
- 买第 i 个商品时，有 $i-1$ 张额外获得的优惠券。

所以第 i 个商品，先被“购买过程中获得的优惠券”抵消掉 $i-1$ 元。

它还剩下的有效价格是：

$$b_i = \max(a_i - (i-1), 0)$$

这一步非常关键。

原题就变成了：

现在有若干个有效价格 b_i。

一开始可以买 k 张优惠券，每张优惠券价格为 w。

买了 k 张优惠券之后，每个商品都能再被减少 k 元。

因此总花费是：

$$kw + \sum_{i=1}^{n} \max(b_i - k, 0)$$

问题变成：

选择一个非负整数 k，让总花费最小。

四、什么时候应该继续买优惠券？

现在只看 b_i 这个新问题。

假设已经决定买 k 张优惠券。

如果再多买一张优惠券，会多花 w 元。

但是它能让所有满足 $b_i > k$ 的商品各少付 1 元。

也就是说，如果当前还有 cnt 个商品没有被优惠到 0，那么多买一张优惠券可以省下 cnt 元。

所以：

- 如果 $cnt > w$，多买一张优惠券很划算；
- 如果 $cnt < w$，多买一张优惠券不划算；
- 如果 $cnt = w$，买不买都一样。

因此，我们应该一直买优惠券，直到：

还能继续产生有效优惠的商品数量不超过 w。

这就是本题最核心的贪心判断。

五、最优的初始优惠券数量是多少？

把所有正的 b_i 从大到小排序。

例如：

$$b_{p_1} \ge b_{p_2} \ge b_{p_3} \ge \cdots$$

当购买的初始优惠券数量 k 较小时，很多商品还没有被优惠到 0。

随着 k 增大，较小的 b_i 会先变成 0，仍然能继续产生优惠的商品数量逐渐减少。

根据上一节的判断：

我们需要找到一个位置，使得继续增加 k 时，能够受益的商品数量大约变成 w。

因此最优的 k 可以取：

所有正的 b_i 中第 w 大的值。

如果正数 b_i 的数量不足 w 个，那么一开始买优惠券不划算，取 $k = 0$ 即可。

六、完整算法

1. 将所有商品价格从小到大排序。
2. 对每个位置 i，计算 $b_i = \max(a_i - (i-1), 0)$。
3. 找出所有正的 b_i。
4. 如果正的 b_i 数量至少为 w，令 k 为第 w 大的 b_i；否则令 $k = 0$。
5. 答案为 $kw + \sum \max(b_i - k, 0)$。

七、样例 2 解释

原数据：

```
4 3
4 4 3 3
```

先按价格从小到大排序：

3 3 4 4

依次购买时，第 i 个商品已经天然拥有 $i-1$ 张优惠券，所以有效价格为：

商品位置	原价格	过程优惠	有效价格
1	3	0	3
2	3	1	2
3	4	2	2
4	4	3	1

所以得到：

$$b=[3,2,2,1]$$

每张优惠券价格 $w=3$。

正的 b_i 从大到小为：

$$3,2,2,1$$

第 3 大是 2，所以一开始买 2 张优惠券。

总花费：

$$2\times 3+\max(3-2,0)+\max(2-2,0)+\max(2-2,0)+\max(1-2,0)$$

也就是：

$$6+1=7$$

答案为 7。

八、复杂度

排序需要 $O(n\log n)$。

其余部分都是线性处理。

总时间复杂度为 $O(n\log n)$，空间复杂度为 $O(n)$。

九、核心总结

这道题的关键不是模拟优惠券，而是两次转化：

第一，把商品按价格从小到大买，让自动获得的优惠券尽量作用在更贵的商品上。

第二，把第 i 个商品变成有效价格 $b_i=\max(a_i-i+1,0)$，然后只需要考虑一开始买多少张优惠券。

最后利用“多买一张优惠券的成本是 w，收益是还没被优惠到 0 的商品数量”来确定最优购买数量。