排列鞋子 （Arranging Shoes）

Adnan擁有在Baku最大的鞋店。這間店剛收到一只包含n雙鞋子的箱子，每雙包含兩隻大小一樣的鞋子：一左腳鞋一右腳鞋。Adnan已將所有$2n$隻鞋子排成一列包含$2n$個**位置**，由左至右從0號編至$2n-1$號。

Adnan想要重新排列這些鞋子以形成一**合法排列** (valid arrangement)。一排列稱之為**合法** (valid)若且唯若對每一i $(0 \leq i \leq n-1)$滿足下列條件：

- 在第$2i$ 及 $2i+1$個位置上的鞋子大小一樣。
- 在第$2i$ 個位置的鞋子是左腳鞋。
- 在第$2i+1$ 個位置的鞋子是右腳鞋。

為達到這一目的，Adnan可以做一連串的交換動作。在每一交換動作，他挑選兩隻當下 **相鄰** (adjacent)的鞋子並將之互換位置 （也就是，挑出兩隻相鄰的鞋子，將其中每隻鞋子放到另一隻鞋子的位置）。如果兩隻鞋子的位置相差一，則稱他們相鄰。

請決定Adnan最少需要執行幾次交換動作才能獲得一合法排列。

實作細節

你應實作下列程序：

```
int64 count_swaps(int[] S)
```

- S: 包含$2n$個整數的陣列。對每一i $(0 \leq i \leq 2n-1)$，$|S[i]|$為一非零的值表示鞋子初始位置為i的鞋子大小。在此$|x|$表示x的絕對值，當$x > 0$時其值為x，當$x < 0$時其值為$-x$。如果$S[i] < 0$，則表示在位置i的鞋子是左腳鞋；否則為右腳鞋。
- 這程序應該回傳最少需要執行相鄰鞋子的交換次數，以獲得合法排列

Examples

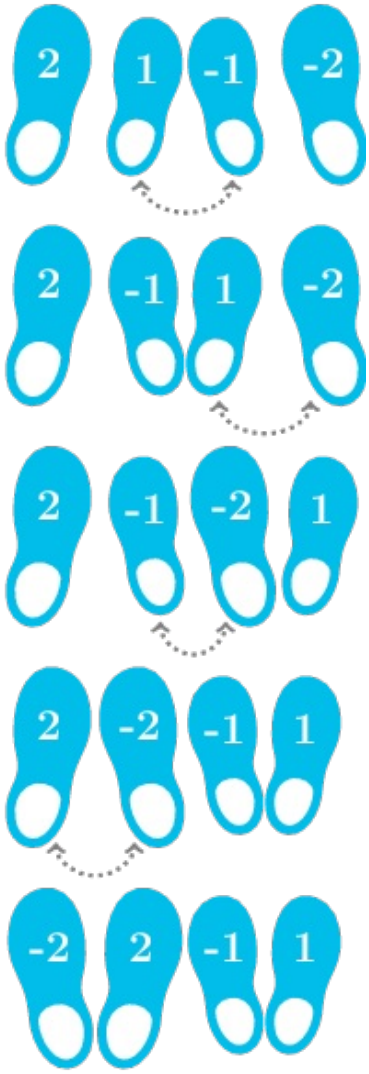
Example 1

考慮下列呼叫：

```
count_swaps([2, 1, -1, -2])
```

Adnan可以用4次交換以得到一合法排列。

例如他可以先交換鞋子1和鞋子-1，然後交換鞋子1和鞋子-2，然後交換鞋子-1和鞋子-2，最後交換鞋子2和鞋子-2。他最終將獲得合法排列：$[-2, 2, -1, 1]$。因不可能用少於4次交換獲得合法排列，故程序應回傳4。



Example 2

在下列例子，所有的鞋子大小都一樣。

```
count_swaps([-2, 2, 2, -2, -2, 2])
```

Adnan能交換在位置2 和位置3的鞋子來獲得合法排列$[-2, 2, -2, 2, -2, 2]$，故程序應該回傳1。

限制 (Constraints)

- $1 \leq n \leq 100\,000$
- 對每一i $(0 \leq i \leq 2n - 1)$, $1 \leq |S[i]| \leq n$。
- 一鞋子的合法排列可以經由執行某一連串交換動作獲得。

Subtasks

1. (10 points) $n = 1$
2. (20 points) $n \leq 8$
3. (20 points) 所有的鞋子大小都一樣。
4. (15 points) 所有在$0, \ldots, n-1$位置的鞋子都是左腳鞋，且所有在$n, \ldots, 2n-1$位置的鞋子都是右腳鞋。並且對每一i $(0 \leq i \leq n-1)$，在位置i 和 位置$i+n$的鞋子大小一樣。
5. (20 points) $n \leq 1000$
6. (15 points) 無其它限制。

Sample grader

此範例評分程式以下列格式讀取輸入：

- line 1: n
- line 2: $S[0]$ $S[1]$ $S[2]$ $\ldots$ $S[2n-1]$

此範例評分程式輸出一行，包含`count_swaps`的回傳值。