T1

当 $a = b$ 时，答案为 0。

当 $a < b$ 时，设 $d = b - a$。

- 若 d 为奇数，一次 $+d$ 即可。
- 若 d 为偶数，且 $d/2$ 为奇数，两次 $+d/2$ 即可。
- 若 d 为偶数，且 $d/2$ 为偶数，两次 $+(d+1)$，一次 $-(d+2)$ 即可。

当 $a > b$ 时，设 $d = a - b$。

- 若 d 为偶数，一次 $-d$ 即可。
- 若 d 为奇数，一次 $+1$，一次 $-(d+1)$ 即可。

T2

70 分:

直接模拟。

100 分:

在中位数处可以取到最优解。按坐标从小到大遍历，就可以找到中位数。之后直接计算即可。

T3

测试点 $1 \sim 2$：

枚举所有情况直接计算。

测试点 3：

输出 $\frac{n\times(n-1)}{2}$。

测试点 $4 \sim 10$：

为了简化问题，我们首先假设序列 A 中的元素是互不相同的。设 $maxA = M$。

我们要对所有 (i, j) 对进行求和，其中 $i < j$。由于加法是对称的，对于 i 和 j，我们可以重新排列序列 A 而不改变答案。现在我们假设 A 是按升序排列的。

当 A 按升序排列时，如果 $i < j$ 则 $A_i \le A_j$，因此

$$\left\lfloor \frac{\min(A_i, A_j)}{\max(A_i, A_j)} \right\rfloor = \left\lfloor \frac{A_i}{A_j} \right\rfloor$$

对于固定的 i，考虑每一个 $\left\lfloor \frac{A_i}{A_j} \right\rfloor$ 而不是 i。也就是说，将其变形为

$$\sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \left\lfloor \frac{A_i}{A_j} \right\rfloor = \sum_{i=1}^{N-1} \sum n \times f(A_i, n)$$

其中 $f(d, n)$ 是使得 $\left\lfloor \frac{d}{A_j} \right\rfloor = n$ 的 j 的数量。（这种替换相当于将例如“1+1+1+2+2+2+2+5+5”看作“1×3 + 2×4 + 3×0 + 4×0 + 5×2”。）

设 C_X 是 $A_i = X$ 的 i 的数量。通过预先计算 C 的累积和，可以在 $O(1)$ 时间内找到 $f(d, n)$。对于固定的 i，n 的范围是 $n \le \frac{A_i}{M}$，所以要求和的项数是

$$\sum_{i=1}^{N-1} \frac{A_i}{M} \le \sum_{d=1}^{N} \frac{d}{M} = O(M \log N) \text{（调和级数的和）}$$

因此，问题在总共 $O(M \log N)$ 时间内解决。

即使 A_i 有重复元素，也可以适当地一次性处理它们，以相同的复杂度找到答案。

T4

测试点 $1 \sim 4$:

形态为链。

枚举从根出发走到的位置，再在走过的路径上枚举起点和终点，并判断是否为合法的括号串。

时间复杂度 $O(n^4)$。

测试点 $5 \sim 7, 11 \sim 14$:

形态为链。

记 num_i 表示 s_i 中合法括号串数目。

可以用栈来模拟括号匹配情况。当走过的点 i 为左括号时，把 i 入栈；当走过的点 i 为右括号时，可以匹配到 stk_i。由于当前形态为链，不难发现 $num_i = num_{stk_{top}-1} + 1$。同时弹出栈顶元素。

这就得到了 $O(n)$ 的解决链形态的算法。

测试点 $8 \sim 10, 15 \sim 20$:

形态为树。

对链形态下的算法稍作调整即可。$num_i = num_{f_{stk_{top}}} + 1$。

同时，树形态下需要记录递归前的栈的状态，回溯时需要还原到该状态。