5.7 節末問題 5.7 的解答

問題 5.7.1

加法式中 2021 有 4 個， 1234 有 5 個。

因此，所求答案為 $2021 \times 4 + 1234 \times 5 = 8084 + 6170 = \mathbf{14254}$ 。

問題 5.7.2

考慮將優美程度的期望值分解為以下的 ${}_6C_2 = 15$ 個部分。

- 部分 1： 由第 1 個和第 2 個骰子的結果算出的優美程度
- 部分 2： 由第 1 個和第 3 個骰子的結果算出的優美程度
- 部分 3： 由第 1 個和第 4 個骰子的結果算出的優美程度
- 部分 4： 由第 1 個和第 5 個骰子的結果算出的優美程度
- ：
- 部分 15： 由第 5 個和第 6 個骰子的結果算出的優美程度

因此，由於以下原因，每個部分中「算出的優美程度」的期望值為 $1/6$ 。

骰子的點數有右圖中的 $6 \times 6 = 36$ 種等機率的情況。

另一方面，兩個骰子點數相同的情況有 6 種，所以其機率為 $6/36 = 1/6$ 。不瞭解的話請回到 3.4 節確認。

		骰子 2					
		1	2	3	4	5	6
骰子 1	1	○	×	×	×	×	×
	2	×	○	×	×	×	×
	3	×	×	○	×	×	×
	4	×	×	×	○	×	×
	5	×	×	×	×	○	×
	6	×	×	×	×	×	○

因此，所求答案為 $1/6 \times 15 = \mathbf{5/2}$ 。（根據期望值的線性性 [→ **3.4 節**]，整體優美的期望值為各部分期望值的和）

問題 5.7.3

首先，考慮 $A_1 \leq A_2 \leq A_3 \leq \cdots \leq A_N$ 的情況。這時，以下的式子成立，因此答案與例題 2（→ **5.7.3項**）相同。。

$$|A_j - A_i| = A_j - A_i\ (1 \leq i \leq j \leq N)$$

よって $$\sum_{i=1}^{N}\sum_{j=i+1}^{N}|A_j - A_i| = \sum_{i=1}^{N}\sum_{j=i+1}^{N} A_j - A_i$$

另一方面，所求答案是「將不同的兩個元素的差全部相加的值」，所以 $A_1, A_2, A_3, \ldots, A_N$ 的順序調換後答案不變。。

例如，當 $A = (1,4,2,3)$ 時的答案是 10，將其排序為 $A = (1,2,3,4)$ 後答案仍是 10。

因此，將數列 $A = (A_1, A_2, \ldots, A_N)$ 升冪排序（→**3.6節**）後，進行與例題 2 相同的處理方式，製作以下程式即可得到正解。

```
#include <iostream>
#include <algorithm>
using namespace std;

long long N, A[200009];
long long Answer = 0;

int main() {
    // 輸入
    cin >> N;
    for (int i = 1; i <= N; i++) cin >> A[i];

    // 排序（從程式碼 5.7.1 唯一增加的部分）
    sort(A + 1, A + N + 1);

    // 求出答案 → 輸出答案
    for (int i = 1; i <= N; i++) Answer += A[i] * (-N + 2LL * i - 1LL);
    cout << Answer << endl;
    return 0;
}
```

※ Python等原始碼請參閱 chap5-7.md。

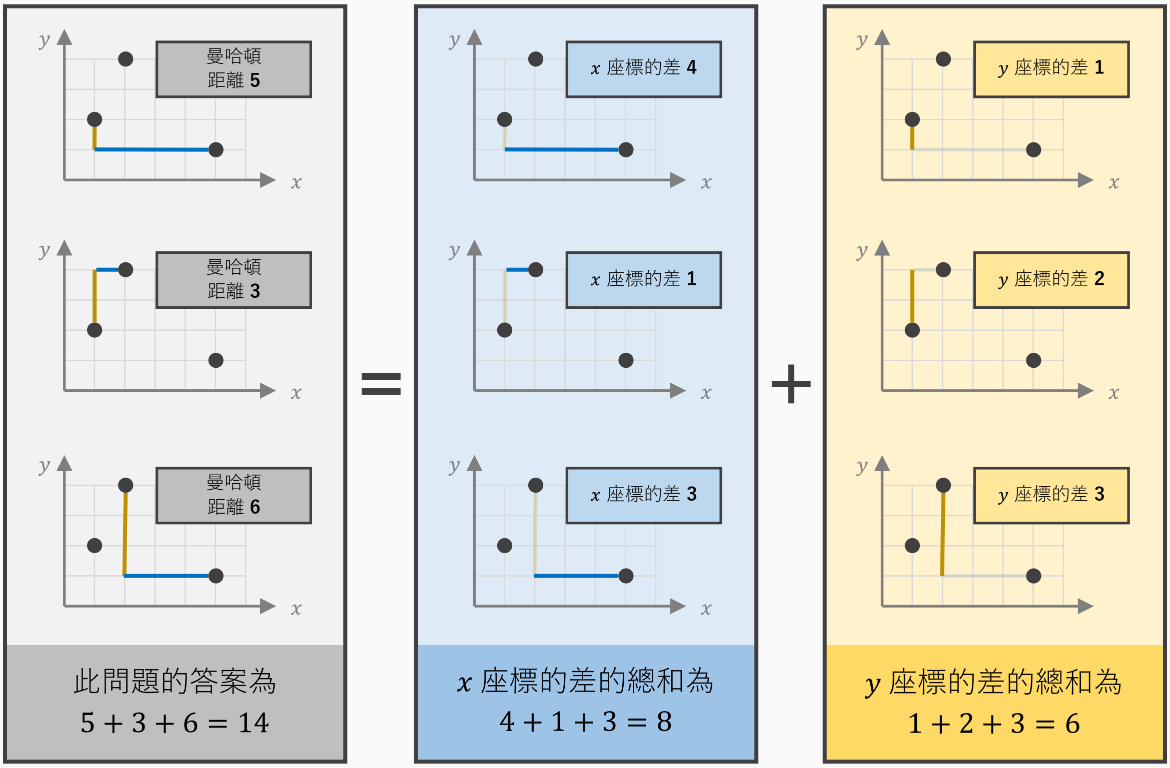
問題 5.7.4

首先，兩點間的曼哈頓距離是 x 座標的差的絕對值與 y 座標的差的絕對值相加的值。

因此，所求的「曼哈頓距離的總和」是以下兩部分的答案相加的值。

- 部分 1： x 座標的差的絕對值的總和
- 部分 2： y 座標的差的絕對值的總和

例如，考慮在座標 $(1,2),(5,1),(2,4)$ 的點。曼哈頓距離的總和為 $5+3+6=14$， x 座標的差的絕對值的總和為 8， y 座標的差的絕對值的總和為 6。



因此，部分 1 和部分 2 的答案可以用以下式子表示。這與節末問題 5.7.3 相同，若將 $(x_1,x_2,\dots,x_N)$ 和 $(y_1,y_2,\dots,y_N)$ 從小到大排序，之後可用 $O(N)$ 的計算複雜度內求出式子的值。

$$Part1=\sum_{i=1}^{N}\sum_{j=i+1}^{N}|x_i-x_j|$$

$$Part2=\sum_{i=1}^{N}\sum_{j=i+1}^{N}|y_i-y_j|$$

因此，撰寫以下程式即可得到正解。在 C++ 中，可以藉由使用標準庫 `std::sort` 來將陣列元素從小到大排序。（→ **3.6.1項**）

```
#include <iostream>
#include <algorithm>
using namespace std;

long long N;
long long X[200009], Y[200009];

int main() {
    // 輸入
    cin >> N;
    for (int i = 1; i <= N; i++) cin >> X[i] >> Y[i];

    // 將陣列排序
    sort(X + 1, X + N + 1);
    sort(Y + 1, Y + N + 1);

    // 部分 1 的答案（x 座標的差的絕對值總和）
    long long Part1 = 0;
    for (int i = 1; i <= N; i++) Part1 += X[i] * (-N + 2LL * i - 1LL);

    // 部分 2 的答案（y 座標的差的絕對值總和）
    long long Part2 = 0;
    for (int i = 1; i <= N; i++) Part2 += Y[i] * (-N + 2LL * i - 1LL);

    // 輸出
    cout << Part1 + Part2 << endl;
    return 0;
}
```

※ Python等原始碼請參閱 chap5-7.md。