5.7 節末問題 5.7 の解答

問題 5.7.1

足し算の式の中に 2021 は 4 個、1234 は 5 個あります。

したがって、求める答えは $2021 \times 4 + 1234 \times 5 = 8084 + 6170 =$ **14254** です。

問題 5.7.2

美しさの期待値を以下の ${}_6C_2 = 15$ 個のパーツに分解することを考えます。

- パーツ 1： 1 番目と 2 番目のサイコロの出方により加算される美しさ
- パーツ 2： 1 番目と 3 番目のサイコロの出方により加算される美しさ
- パーツ 3： 1 番目と 4 番目のサイコロの出方により加算される美しさ
- パーツ 4： 1 番目と 5 番目のサイコロの出方により加算される美しさ
- ：
- パーツ 15： 5 番目と 6 番目のサイコロの出方により加算される美しさ

そこで、以下の理由により、各パーツにおける「加算される美しさ」の期待値は 1/6 となります。

サイコロの出目は右図の $6 \times 6 = 36$ 通りが等確率で起こり得ます。

一方、そのうち 2 つの出目が同じとなるものは 6 通りであるため、その確率は $6/36 = 1/6$ です。分からない人は 3.4 節に戻って確認しましょう。

		サイコロ 2					
		1	2	3	4	5	6
サイコロ 1	1	○	×	×	×	×	×
	2	×	○	×	×	×	×
	3	×	×	○	×	×	×
	4	×	×	×	○	×	×
	5	×	×	×	×	○	×
	6	×	×	×	×	×	○

したがって、求める答えは $1/6 \times 15 =$ **5/2** となります。（期待値の線形性 [→**3.4 節**] より、全体の美しさの期待値は、各パーツの期待値の和となります）

問題 5.7.3

まず、$A_1 \leq A_2 \leq A_3 \leq \cdots \leq A_N$ のケースを考えましょう。このとき、以下の式が成り立つため、答えは例題 2（→**5.7.3項**）と同一となります。

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

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

一方、求めるべき答えは「異なる 2 つの要素の差を全部足した値」であるため、$A_1, A_2, A_3, \ldots, A_N$ の順序を入れ替えても答えは変わりません。

たとえば、$A = (1,4,2,3)$ の場合の答えは 10 であり、それを並べ替えた $A = (1,2,3,4)$ の場合の答えも 10 です。

したがって、数列 $A = (A_1, A_2, \ldots, A_N)$ を昇順にソート（→**3.6節**）した後、例題 2 と同じ処理を行う以下のようなプログラムを作成すると、正解が得られます。

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

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

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

        // ソート（コード 5.7.1 から追加した唯一の部分）
        sort(A + 1, A + N + 1);

        // 答えを求める → 答えの出力
        for (int i = 1; i <= N; i++) Answer += A[i] * (-N + 2LL * i - 1LL);
        cout << Answer << endl;
        return 0;
}
```

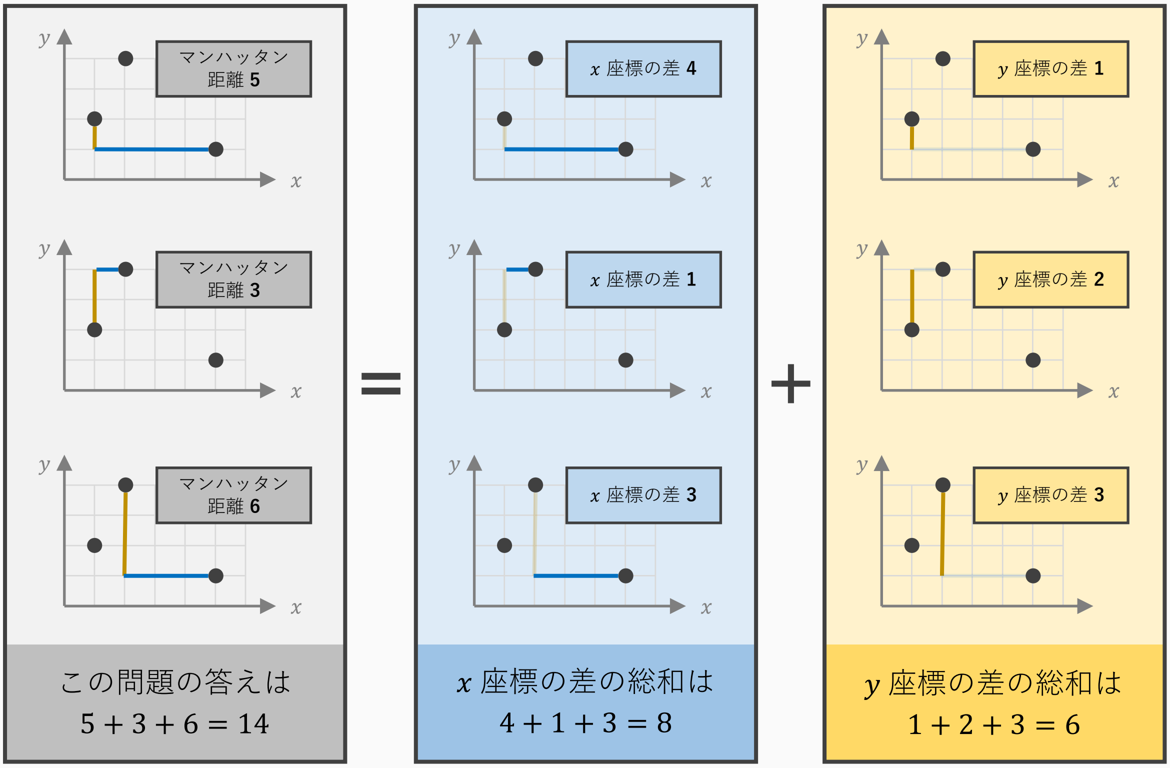
※ Python などのソースコードは chap5-6.md をご覧ください。

問題 5.7.4

まず、2 点間のマンハッタン距離は、x 座標の差の絶対値と y 座標の差の絶対値を足した値です。したがって、求めるべき「マンハッタン距離の総和」は、以下の 2 つのパーツの答えを足した値となります。

- パーツ 1： x 座標の差の絶対値の総和
- パーツ 2： y 座標の差の絶対値の総和

たとえば、座標 $(1,2),(5,1),(2,4)$ に点がある場合を考えましょう。マンハッタン距離の総和は $5+3+6=14$ である一方、x 座標の差の絶対値の総和は 8、y 座標の差の絶対値の総和は 6 です。



そこで、パーツ 1・パーツ 2 の答えは、以下のような式で表されます。これは節末問題 5.7.3 と同じ形であり、$(x_1, x_2, \dots, x_N)$ と $(y_1, y_2, \dots, y_N)$ を小さい順にソートすると、あとは計算量 $O(N)$ で式の値が求められます。

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

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

よって、以下のようなプログラムを書くと、正解が得られます。なお、C++ では標準ライブラリ `std::sort` を使うことで、配列の要素を小さい順にソートすることができます。（→**3.6.1項**）

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

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

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

	// 配列をソートする
	sort(X + 1, X + N + 1);
	sort(Y + 1, Y + N + 1);

	// パーツ 1 の答え（x 座標の差の絶対値の総和）
	long long Part1 = 0;
	for (int i = 1; i <= N; i++) Part1 += X[i] * (-N + 2LL * i - 1LL);

	// パーツ 2 の答え（y 座標の差の絶対値の総和）
	long long Part2 = 0;
	for (int i = 1; i <= N; i++) Part2 += Y[i] * (-N + 2LL * i - 1LL);

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