4.6 節末問題 4.6 の解答

問題 4.6.1 (1)

掛け算の式はどのようなタイミングで余りをとっても答えが変わらないため、

- $21 \times 41 \times 61 \times 81 \times 101 \times 121$ を 20 で割った余り
- $1 \times 1 \times 1 \times 1 \times 1 \times 1$ を 20 で割った余り

は等しいです。後者は明らかに 1 なので、この問題の答えは 1 です。

問題 4.6.1 (2)

計算前にすべて 100 で割った余りをとっても答えは変わらないため、

- 202112^5 を 100 で割った余り
- 12^5 を 100 で割った余り

は一致します。$12^5 = 248832$ なので答えは 32 だと分かりますが、手計算では少々面倒です。そこで、以下のように計算途中でも余りをとりましょう。

- $12 \times 12 = 144 \equiv 44 \pmod{100}$
- $44 \times 12 = 528 \equiv 28 \pmod{100}$
- $28 \times 12 = 336 \equiv 36 \pmod{100}$
- $36 \times 12 = 432 \equiv 32 \pmod{100}$

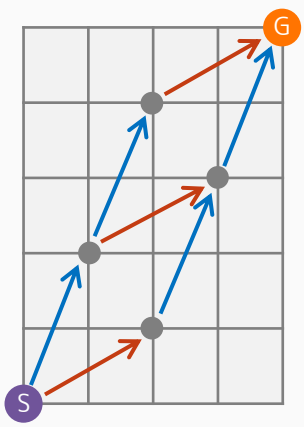
そうすると、高々 3 桁の数の計算で答えを求めることができます。

問題 4.6.2

まずは具体例から考えましょう。コマをマス $(4, 5)$ まで移動させるには、

- $(i, j) \to (i+1, j+2)$ の移動を 2 回
- $(i, j) \to (i+2, j+1)$ の移動を 1 回

行う必要があります。逆にその条件さえ満たせば必ず目的の場所にたどり着きます。3 回中 2 回が $(i+1, j+2)$ への移動なので、移動方法の数は ${}_3\mathrm{C}_2 = 3$ 通りです。



次に、一般化したケースを考えましょう。

- $(i,j) \to (i+1,j+2)$ の移動を a 回（移動 A とする）
- $(i,j) \to (i+2,j+1)$ の移動を b 回（移動 B とする）

行うとき、マス (X,Y) に移動させるには以下の 3 条件を満たす必要があります。

- a,b は非負整数である。
- x 座標の制約： $a+2b=X$
- y 座標の制約： $2a+b=Y$

ここで、2 つ目・3 つ目を同時に満たす組は、$(a,b)=\left(\frac{2Y-X}{3},\frac{2X-Y}{3}\right)$ のみです。

しかし、1 つ目の条件より、答えが 0 通りになる場合があります。

> - a,b は整数なので、$2Y-X, 2X-Y$ が両方 3 の倍数でなければ 0 通り
> - a,b は 0 以上なので、$2Y-X<0$ または $2X-Y<0$ であれば 0 通り

そうでない場合は、全体の移動回数 $(X+Y)/3$ 回のうち $(2Y-X)/3$ 回を移動 A にすれば必ず目的のマスにたどり着けるため、求める答えは ${}_{(X+Y)/3}\mathrm{C}_{(2Y-X)/3}$ 通りとなります。コード 4.6.5 を少し変えた以下のような実装をすると、正解が得られます。

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

const long long mod = 1000000007;
int X, Y;

long long modpow(long long a, long long b, long long m) {
    // 繰り返し二乗法（p は a^1, a^2, a^4, a^8, ... といった値をとる）
    long long p = a, Answer = 1;
    for (int i = 0; i < 30; i++) {
        if ((b & (1 << i)) != 0) { Answer *= p; Answer %= m; }
        p *= p; p %= m;
    }
    return Answer;
}

// Division(a, b, m) は a÷b mod m を返す関数
long long Division(long long a, long long b, long long m) {
    return (a * modpow(b, m - 2, m)) % m;
}
```

```cpp
int main() {
    // 入力
    cin >> X >> Y;

    // 場合分け（a, b が負になってしまう場合）
    if (2 * Y - X < 0 || 2 * X - Y < 0) {
        cout << "0" << endl;
        return 0;
    }

    // 場合分け（a, b が整数にならない場合）
    if ((2 * Y - X) % 3 != 0 || (2 * X - Y) % 3 != 0) {
        cout << "0" << endl;
        return 0;
    }

    // 二項係数の分子と分母を求める（手順 1.／手順 2.）
    long long bunshi = 1, bunbo = 1;
    long long a = (2 * Y - X) / 3, b = (2 * X - Y) / 3;
    for (int i = 1; i <= a + b; i++) { bunshi *= i; bunshi %= mod; }
    for (int i = 1; i <= a; i++) { bunbo *= i; bunbo %= mod; }
    for (int i = 1; i <= b; i++) { bunbo *= i; bunbo %= mod; }

    // 答えを求める（手順 3.）
    cout << Division(bunshi, bunbo, mod) << endl;
    return 0;
}
```

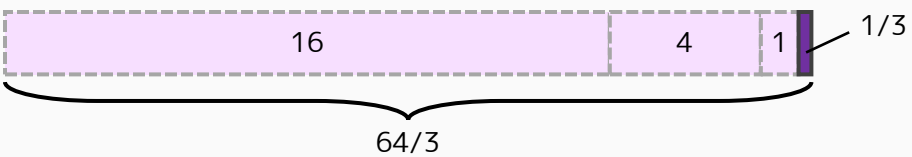
※ Python などのソースコードは chap4-6.md をご覧ください。

問題 4.6.3

まず、以下の式が成り立ちます。たとえば $N=2$ のとき $4^0+4^1+4^2=1+4+16=21$ である一方、$(4^{N+1}-1)/3=63\div 3=21$ であり、2 つの値が一致します。

$$4^0+4^1+4^2+\cdots+4^N=\frac{4^{N+1}-1}{3}$$

証明はやや難しいですが、長さ $4^{N+1}/3$ の棒を $3/4$ だけ切り取る操作を $N+1$ 回繰り返すと、1 回目から順に長さ $4^N, 4^{N-1}, \ldots, 4^0$ のものが取り出され、最終的に長さ $1/3$ だけ残ることを考えると良いです。（参考：和の公式 →**2.5.10項**）



したがって、以下のような方法で答えを $M = 1000000007$ で割った余りを求めることができます。

- $4^{N+1} - 1$ を M で割った余り V を求める（→**4.6.7項**）
- $V \div 3$ を M で割った余りを求める（→**4.6.8項**）

実装例は以下の通りです。なお、関数 `Division(a, b, m)` は、$a \div b$ を m で割った余りを返すものとなっています。また、制約が $N \leq 10^{18}$ であるため、`modpow` 関数のループ回数が $\log_2(10^{18}) \fallingdotseq 60$ 回程度必要であることに注意してください。

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

const long long mod = 1000000007;
long long N;

long long modpow(long long a, long long b, long long m) {
    // 繰り返し二乗法（p は a^1, a^2, a^4, a^8, ... といった値をとる）
    long long p = a, Answer = 1;
    for (int i = 0; i < 60; i++) {
        if ((b & (1LL << i)) != 0) { Answer *= p; Answer %= m; }
        p *= p; p %= m;
    }
    return Answer;
}

// Division(a, b, m) は a÷b mod m を返す関数
long long Division(long long a, long long b, long long m) {
    return (a * modpow(b, m - 2, m)) % m;
}

int main() {
    // 入力
    cin >> N;

    // 答えの計算
    long long V = modpow(4, N + 1, mod) - 1;
    long long Answer = Division(V, 3, mod);

    // 出力
    cout << Answer << endl;
    return 0;
}
```

※ Python などのソースコードは chap4-6.md をご覧ください。