問題

整数N, M, Pが与えられる．このとき，N^PをMで割った余りを求めよ．

制約

- $1 \leq N, M \leq 10^9$
- $1 \leq P \leq 10^{14}$

出典

AtCoder Typical Contest 002 B

コード解説

normal1.cpp

for文を用いてansにNをP回かける．計算量は$O(P)$．

```
#include<bits/stdc++.h>
using namespace std;

int main() {
    long long n, m, p;
    cin >> n >> m >> p;
    long long ans = 1;

    // 計算量はO(p)
    for (int _ = 0; _ < p; _++) {
        ans *= n;
        ans %= m;
    }

    cout << ans << endl;
    return 0;
}
```

normal2.cpp

再帰関数を用いた累乗のナイーブな実装.

$$N^X = N^{X-1} \cdot N$$

を用いて再帰的に計算する. 計算量は$O(P)$.

```
#include<bits/stdc++.h>
using namespace std;

// return n^p % m
// 計算量はO(p)
long long pow(long long n, long long p, long long m) {
    if (p == 0) return 1;
    return n * pow(n, p - 1, m) % m;
}

int main(){
    long long n, m, p;
    cout << pow(n, m, p) << endl;
    return 0;
}
```

kuhuu1.cpp

繰り返し2乗法の再帰帰数を用いた実装.

$$N^X = \begin{cases} N^{X/2} \cdot N^{X/2} & (X\text{が偶数}) \\ N^{(X-1)/2} \cdot N^{(X-1)/2} \cdot N & (X\text{が奇数}) \end{cases}$$

を用いて再帰的に計算する. 計算量は$O(\log P)$.

```
#include<bits/stdc++.h>
using namespace std;

// return n^p % m
// 計算量はO(log(p))
long long power(long long n, long long p, long long m) {
    n %= m;
    if (n < 0) n += m;
    if (p == 0) return 1;
    long long res = power(n, p / 2, m);
    if (p % 2 == 0) return (res * res) % m;
    else return (res * res) % m * n % m;
}

int main(){
    long long n, m, p;
    cin >> n >> m >> p;
```

```
    cout << power(n, p, m) << endl;
    return 0;
}
```

kuhuu2.cpp

繰り返し2乗法の非再帰的な実装．

$$X = \sum_{i=0}^{h} a_i \cdot 2^i,$$

のように書けたとき，

$$N^X = \left(N^{2^0}\right)^{a_0} \cdot \left(N^{2^1}\right)^{a_1} \cdot \cdots \cdot \left(N^{2^h}\right)^{a_h}$$

という性質 (分配法則です．) を使い，N^{2^i} を計算しながら，$a_i = 1$ の時に，ans に N^{2^i} をかける．

$$N^{2^{i+1}} = N^{2^i} \cdot N^{2^i}$$

であるから，N^{2^i} は簡単に計算できる．$h = \lceil \log_2(X) \rceil$ だから，計算量は $O(\log P)$．

```
#include<bits/stdc++.h>
using namespace std;

int main(){
    long long n, m, p;
    cin >> n >> m >> p;
```

```
    long long tmp = n % m, ans = 1, exp = p;
    // 再帰を使わない方法
    // 計算量はO(log(p))
    while (exp) {
        if (exp & 1) {
            ans *= tmp;
            ans %= m;
        }
        tmp *= tmp;
        tmp %= m;
        exp >>= 1;
    }
    cout << ans << endl;
    return 0;
}
```

まとめ

繰り返し2乗法の実装は，再帰関数を用いたものと，用いなかったもののどちらも，Github Copilotに大まかに実装してもらい，少し修正を加えている (余りを取るところ．) ．

参考記事

- 「998244353 で割ったあまり」の求め方を総特集！ 〜 逆元から離散対数まで 〜 4章 累乗