Contents

1 Math

1.1 快速冪

```
/*快速冪*/
ll mypow(ll x, ll y, ll p) {
    long long ans = 1;
    while (y) {
        if (y & 1)    ans = ans * x % p;    //prime
        x = x * x % p;    //每次把自己平方
        y >>= 1;    //每次右移一格
    }
    return ans;
}
```

1.2 快速乘

```
/*快速乘(a * b) mod m 大數乘法取餘數*/
ll mul(ll x, ll y, ll mod) {
    ll ret = x * y - (ll)((long double)x / mod * y) *
        mod;
    // LL ret=x*y-(LL)((long double)x*y/mod+0.5)*mod;
    return ret < 0 ? ret + mod : ret;
}
```

1.3 GCD

```
/*GCD*/
ll gcd(ll a, ll b){
    return b == 0 ? a : gcd(b, a % b);
}
```

1.4 ax+by=gcd(a,b)

```
/*ax+by=gcd(a,b)*/
ll a, b, x, y;
ll exgcd(ll a, ll b, ll& x, ll& y) {
    if (b) {
        ll d = exgcd(b, a % b, y, x);
        return y -= a / b * x, d;
    }
    return x = 1, y = 0, a;
}
```

1.5 Sieve Prime

```
/*Sieve_Prime*/
const int N = 20000000;//質數表大小
bool sieve[N];
vector<int> prime;
void linear_sieve(){
    for (int i = 2; i < N; i++)
    {
        if (!sieve[i]) prime.push_back(i);
        for (int p : prime)
        {
            if (i * p >= N) break;
            sieve[i * p] = true;
            if (i % p == 0) break;
        }
    }
}
```

1.6 Miller Rabin

```
/*Miller_Rabin 質數判定*/
// n < 4,759,123,141        3 :  2, 7, 61
// n < 1,122,004,669,633    4 :  2, 13, 23, 1662803
// n < 3,474,749,660,383          6 :  pirmes <= 13
// n < 2^64                       7 :
// 2, 325, 9375, 28178, 450775, 9780504, 1795265022
// Make sure testing integer is in range [2, n-2] if
// you want to use magic.
ll magic[N] = {};
bool witness(ll a, ll n, ll u, int t) {
    if (!a) return 0;
    ll x = mypow(a, u, n);  //快速冪
    for (int i = 0; i < t; i++) {
        ll nx = mul(x, x, n);   //快速乘
        if (nx == 1 && x != 1 && x != n - 1) return 1;
        x = nx;
    }
    return x != 1;
}
bool miller_rabin(ll n) {
    int s = (magic number size);
        // iterate s times of witness on n
        if (n < 2) return 0;
    if (!(n & 1)) return n == 2;
    ll u = n - 1; int t = 0;
    // n-1 = u*2^t
    while (!(u & 1)) u >>= 1, t++;
    while (s--) {
        ll a = magic[s] % n;
        if (witness(a, n, u, t)) return 0;
    }
    return 1;
}
```

1.7 Fibonacci

```
/*Fibonacci*/
int Fib[100005];
int F(int n) {
    Fib[0] = 0; Fib[1] = 1;

    for (int i = 2; i <= n; i++)
        Fib[i] = Fib[i - 1] + Fib[i - 2];

    return Fib[n];
}
```

1.8 josephus

```
/*約瑟夫問題：n個人圍成一桌，數到m的人出列*/
int josephus(int n, int m) {  //n人每m次
    int ans = 0;
    for (int i = 1; i <= n; ++i)
        ans = (ans + m) % i;
    return ans;
}
```

1.9 MOD

```
/*MOD*/
/// _fd(a,b)  floor(a/b).
/// _rd(a,m)  a-floor(a/m)*m.
/// _pv(a,m,r) largest x s.t x<=a && x%m == r.
/// _nx(a,m,r) smallest x s.t x>=a && x%m == r.
/// _ct(a,b,m,r) |A| , A = { x : a<=x<=b && x%m == r }.
int _fd(int a, int b) { return a < 0 ? (-~a / b - 1) :
    a / b; }
int _rd(int a, int m) { return a - _fd(a, m) * m; }
int _pv(int a, int m, int r) {
    r = (r % m + m) % m;
    return _fd(a - r, m) * m + r;
}
int _nt(int a, int m, int r) {
    m = abs(m);
    r = (r % m + m) % m;
    return _fd(a - r - 1, m) * m + r + m;
}
int _ct(int a, int b, int m, int r) {
    m = abs(m);
    a = _nt(a, m, r);
    b = _pv(b, m, r);
    return (a > b) ? 0 : ((b - a + m) / m);
}
```

1.10 Epsilon

```
/*精準度(Epsilon)*/
void Equal(float a, float b)    //判斷相等
{
    float eps = 1e-8;
    if ((fabs(a - b)) < eps)
        printf("Yes\n");
    else printf("No\n");
}
void NEqual(float a, float b)   //判斷不相等
{
    float eps = 1e-8;
    if ((fabs(a - b)) > eps)
        printf("Yes\n");
    else printf("No\n");
}
void Less(float a, float b) //判斷小於
{
    float eps = 1e-8;
    if ((a - b) < -eps)
        printf("Yes\n");
    else printf("No\n");
}
void Greater(float a, float b)  //判斷大於
{
    float eps = 1e-8;
    if ((a - b) > eps)
        printf("Yes\n");
    else printf("No\n");
}
```

1.11 Big number

```
/*大數(Big Number)*/
void add(int a[100], int b[100], int c[100])    //加法
{
    int i = 0, carry = 0;
    for (i = 0; i < 100; ++i) {
        c[i] = a[i] + b[i] + carry;
        carry = c[i] / 10;
        c[i] %= 10;
    }
}
void sub(int a[100], int b[100], int c[100])    //減法
{
    int i = 0, borrow = 0;
    for (i = 0; i < 100; ++i) {
        c[i] = a[i] - b[i] - borrow;
        if (c[i] < 0) {
            borrow = 1;
            c[i] += 10;
        }
        else
            borrow = 0;
    }
}
void mul(int a[100], int b[100], int c[100])    //乘法
{
    int i = 0, j = 0,carry = 0;
    for (i = 0; i < 100; ++i) {
        if (a[i] == 0) continue;
        for (j = 0; j < MAX; ++j)
            c[i + j] += a[i] * b[i];
    }

    for (i = 0; i < MAX; ++i) {
        carry = c[i] / 10;
        c[i] %= 10;
    }
}
void div(int a[100], int b[100], int c[100])    //除法
{
    int t[100];

    for (i = 100 - 1; i >= 0; i--) {
        for (int k = 9; k > 0; k--) // 嘗試商數
        {
            mul(b + i, k, t);
            if (largerthan(a + i, t))
            {
                sub(a + i, t, c + i);
                break;
            }
        }
    }
}
```