# 目录

# 基础

### 最大流最小割定理

```
struct Fraction
{
    int num, den;
    Fraction(int num = 0, int den = 1) {
        if (den < 0) {
            num = -num, den = -den;
        }
        assert(den != 0);
        int g = gcd(abs(num), den);
        num /= g, den /= g;
        this->num = num, this->den = den;
    }

    Fraction operator +(const Fraction &f) const {
        return Fraction(num*f.den+f.num*den,
den*f.den);
    }

    Fraction operator -(const Fraction &f) const {
        return Fraction(num*f.den-f.num*den,
den*f.den);
    }

    Fraction operator *(const Fraction &f) const {
        return Fraction(num*f.num, den*f.den);
    }

    Fraction operator /(const Fraction &f) const {
        return Fraction(num*f.den, den*f.num);
    }

    bool operator <(const Fraction &f) const {
        return (num*f.den < f.num*den);
    }

    bool operator ==(const Fraction &f) const {
        return (den == f.den && num == f.num);
    }

};
```

# 数据结构

### 左偏树

### 分数

```
struct Fraction
{
    int num, den;
    Fraction(int num = 0, int den = 1) {
        if (den < 0) {
            num = -num, den = -den;
        }
        assert(den != 0);
        int g = gcd(abs(num), den);
        num /= g, den /= g;
        this->num = num, this->den = den;
    }
};
```

# 图论

# 黑科技

### IO 优化

```
template<typename T = int>
inline T read() {
    T val = 0, sign = 1; char ch;
    for (ch = getchar(); ch < '0' || ch > '9'; ch =
getchar())
        if (ch == '-') sign = -1;
    for (; ch >= '0' && ch <= '9'; ch = getchar())
        val = val * 10 + ch - '0';
    return sign * val;
}
```