算法笔记

La5te2

October 25, 2025

Contents

I 数论 **9**

1 数论基础 **9**
1.1 素性检测 . . . 9
1.2 质因数分解 . . . 9
1.3 数论函数 . . . 9
1.4 素数计数 . . . 9
1.5 升幂引理 . . . 9

2 欧拉函数 **10**
2.1 定义与性质 . . . 10
2.2 欧拉定理 . . . 10
2.3 扩展欧拉定理 . . . 10
2.4 欧拉反演 . . . 10
2.5 欧拉筛 . . . 10

3 阶与原根 **11**
3.1 基本定义 . . . 11
3.2 性质与定理 . . . 11
3.3 Carmichael 函数 . . . 14

4 欧几里得算法 **15**
4.1 扩展欧式算法 . . . 15
4.2 类欧式算法 . . . 15

5 同余方程 **16**
5.1 裴蜀定理 . . . 16
5.2 多元不定方程 . . . 16
5.3 线性同余方程 . . . 16
5.4 模逆元筛 . . . 16
5.5 CRT 定理 . . . 16
5.6 扩展 CRT 定理 . . . 16
5.7 阶乘取模 . . . 16
5.8 N 次同余方程 . . . 16

6 离散对数和剩余 **17**
6.1 BSGS 算法 17
6.2 扩展 BSGS 算法 17
6.3 离散对数筛 18
6.4 N 次剩余 18
6.5 二次剩余 18

7 卢卡斯定理 **19**

8 数论分块 **20**
8.1 引理与结论 20
8.2 N 维数论分块 20
8.3 向上取整 21
8.4 扩展数论分块 21

9 **Dirichlet** 卷积 **22**
9.1 Dirichlet 前缀和 22
9.2 Dirichlet 后缀和 23
9.3 DGF 与卷积 23

10 积性函数筛 **24**

11 莫比乌斯函数 **25**

II 图论 **26**

1 图论基础 **26**

2 最短路问题 **27**

3 生成树问题 **28**

4 连通性问题 **29**
4.1 基本定义 29
4.2 强连通分量 29
4.3 2-SAT 问题 29

5 环计数问题 **30**

6 欧拉图问题 **31**

7 竞赛图问题 **32**

8 平面图问题 **33**
8.1 基本定义 33
8.2 广义串并联图 33

9 弦图问题 **34**

10 网络流 **35**
10.1 基本定义 35
10.2 最大流 35
10.3 最小割树 38
10.4 有界流 38
10.5 费用流 38
10.6 常见模型 38

11 序理论 **39**
11.1 基本定义 39
11.2 拓扑排序 39
11.3 Dilworth 定理 39

12 图的着色 **40**

13 图的匹配 **41**
13.1 基本定义 41
13.2 二分图最大匹配 41
13.3 二分图最大权匹配 41
13.4 树上匹配问题 41
13.5 一般图最大匹配 41

14 树论基础 **42**
14.1 树的遍历 42
14.2 树的直径 42
14.3 树的中心 42
14.4 树的重心 42
14.5 最近公共祖先 42
14.6 树上差分 42
14.7 树的同构 42
14.8 树上分治 42

15 树链剖分 **43**
15.1 基本定义 43
15.2 重链剖分 43
15.3 长链剖分 44

16 树分解问题 **45**

17 基环树问题 **46**

18 虚树问题 **47**

19 支配树问题 **48**

20 Prüfer 序列 **49**

III 线性代数 50

1 线性基 **50**
1.1 定义与性质 50
1.2 异或线性基 51
1.3 实数线性基 52

2 特征根 **53**
2.1 基本定义 53
2.2 斐波那契数列 53
2.3 带常线性递推 53

IV 组合数学 **54**

1 基本原理 **54**
1.1 计数原理 54
1.2 排列组合 54
1.3 鸽巢原理 54
1.4 容斥原理 54

2 卡特兰数 **55**

3 斯特林数 **56**

4 伯努利数 **57**

5 分拆数 **58**

6 范德蒙德卷积 **59**

7 Pólya 计数 **60**

8 图论计数 **61**

9 生成函数 **62**

V 计算几何 **63**

1 几何基础 **63**

VI 概率论 **64**

1 期望问题 **64**
1.1 基本定义 64
1.2 条件期望 64
1.3 经典模型 64

VII 博弈论 **65**

1 公平组合博弈 **65**
1.1 基本定义 65
1.2 Nim 游戏 65
1.3 SG 理论 66
1.4 经典模型 66

VIII 字符串 67

1 前缀数组 **67**

2 **Z** 函数 **68**

3 字符串哈希 **69**

4 **BM** 匹配算法 **70**

5 字典树 **71**

6 **AC** 自动机 **72**

7 后缀数组 **73**

8 后缀自动机 **74**
8.1 广义 SAM 74

9 后缀字典树 **75**

10 后缀平衡树 **76**

11 **Manacher** 算法 **77**

12 回文自动机 **78**

13 子序列自动机 **79**

14 **Lyndon** 分解 **80**

15 **ML** 算法 **81**

IX 数据结构 82

1 并查集 **82**
1.1 基本定义 82
1.2 路径压缩 82
1.3 按秩合并 82
1.4 带权并查集 82
1.5 种类并查集 82
1.6 可撤销并查集 82

2 哈希表 **83**

3 栈与队列 **84**

4 链表 **85**

5 稀疏表 **86**

6 珂朵莉树 **87**

7 树状数组 **88**

8 可并堆 **89**

9 线段树入门 **90**

10 平衡树入门 **91**
10.1 二叉搜索树 91
10.2 Treap 树 91
10.3 Spaly 树 91
10.4 笛卡尔树 91
10.5 替罪羊树 91

11 **K-D** 树 **92**

12 析合树 **93**

13 分块入门 **94**
13.1 基本思想 94
13.2 块状数组 94
13.3 SQRT 树 94
13.4 Bitset 相关 94

14 莫队算法 **95**

15 整体二分 **96**

16 **CDQ** 分治 **97**

17 可持久化 **98**

18 可追溯化 **99**

19 树套树 **100**

20 动态树 **101**
20.1 Link-Cut Tree 101
20.2 欧拉回路树 101
20.3 全局平衡二叉树 101
20.4 自适应 Top-Tree 101

X 动态规划 **102**

1 背包 DP **102**
1.1 0-1 背包 . . . 102
1.2 完全背包 . . . 102
1.3 多重背包 . . . 102
1.4 树形背包 . . . 102

XI 其他专题 **103**

1 均摊分析 **103**

2 计算理论 **104**
2.1 自动机 . . . 104
2.2 形式语言 . . . 104
2.3 可计算性 . . . 104
2.4 复杂性理论 . . . 104

3 随机化算法 **105**
3.1 随机函数 . . . 105
3.2 模拟退火 . . . 105
3.3 Moser 算法 . . . 105
3.4 梅森旋转算法 . . . 105
3.5 蒙特卡洛方法 . . . 105

4 启发式算法 **106**
4.1 人类智慧 . . . 106
4.2 启发式搜索 . . . 106
4.3 启发式合并 . . . 106
4.4 启发式分裂 . . . 106
4.5 DSU-on-Tree . . . 106

5 根号分治 **107**

6 数值算法 **108**
6.1 区间逼近算法 . . . 108
6.2 自适应辛普森 . . . 108
6.3 多项式插值 . . . 108

7 连续段问题 **109**

8 编码问题 **110**

9 拟阵问题 **111**

10 凸函数问题 **112**

11 最优化问题 **113**
11.1 拉格朗日乘数法 113
11.2 拉格朗日对偶 113
11.3 线性规划 113

12 独立集问题 **114**

Part I
数论

1 数论基础

1.1 素性检测

1.2 质因数分解

1.3 数论函数

1.4 素数计数

1.5 升幂引理

2 欧拉函数

2.1 定义与性质

2.2 欧拉定理

> 给定 $a, b, m \in \mathbf{N}^*, \gcd(a, m) = 1$，求 $a^b \bmod m$.

当 $\gcd(a, m) = 1$ 时，若 x_i 通过 m 的简化系，则 ax_i 通过 m 的简化系，即

$$\prod_{i=1}^{\varphi(m)} ax_i \equiv \prod_{i=1}^{\varphi(m)} x_i \pmod{m}$$

因为 $\gcd(m, \prod_{i=1}^{\varphi(m)} x_i) = 1$，有欧拉定理 $a^{\varphi(m)} \equiv 1 \pmod{m}$，从而 $a^b \equiv a^{b \bmod \varphi(m)} \pmod{m}$

2.3 扩展欧拉定理

> 给定 $a, b, m \in \mathbf{N}^*$，求 $a^b \bmod m$.

对于本题，当 b 小于 $\varphi(m)$ 时，直接做快速幂，否则有

$$a^b \equiv a^{b \bmod \varphi(m) + \varphi(m)} \pmod{m}$$

证明：当 $\gcd(a, m) = 1$ 时，做法同 1.2。

当 $\gcd(a, m) \neq 1$ 时，由算术基本定理，$a = \prod_{i=1}^{k} p_i^{e_i}$，即证 $p_i^b \equiv p_i^{r+\varphi(m)} \pmod{m}$。

记 p_i 为 p，若 $\gcd(p, m) = 1$，证明同 1.2，否则有 $m \geq 2p$。

令 $m = s \times p^t, t = \lfloor \log_p m \rfloor$，则 $\gcd(s, p^t) = \gcd(s, p) = 1$，有

$$p^{\varphi(s)} \equiv 1 \pmod{s}, \varphi(m) = \varphi(s)\varphi(p^t) \Rightarrow p^{\varphi(m)} = (p^{\varphi(s)})^{\varphi(p^t)} \equiv 1 \pmod{s}$$

两边同时乘以 p^t，得到 $p^{t+\varphi(m)} \equiv p^t \pmod{s \times p^t}$，即 $p^{t+\varphi(m)} \equiv p^t \pmod{m}$，从而

$$p^b = p^{b-t} \times p^t \equiv p^{b-t+t+\varphi(m)} = p^{b+\varphi(m)} \pmod{m}$$

记 $f(x) = p^x \bmod m, x \geq \varphi(m)$，则

$$f(x + \varphi(m)) = f(x) = f(x - \varphi(m))$$

又由定义域知 $x - \varphi(m) \geq \varphi(m)$，得

$$f(x) = f(x \bmod m + \varphi(m))$$

即

$$p^b \equiv p^{b \bmod \varphi(m) + \varphi(m)} \pmod{m}$$

得证。

2.4 欧拉反演

2.5 欧拉筛

3 阶与原根

3.1 基本定义

由欧拉定理可知，满足 $a^n \equiv 1 \pmod{m}$ 的最小 $n \in \mathbf{N}^*$ 存在，称为 a 模 m 的阶，记作 $\delta_m(a)$。若 $\gcd(a,m)=1$ 且 $\delta_m(a)=\varphi(m)$，称 a 为模 m 的原根。

3.2 性质与定理

> 给定 $n \in \mathbf{N}^*$，求它在 $[1,n)$ 中所有原根.

阶的三条性质：

1. 若 $a^n \equiv 1 \pmod{m}$，则 $\delta_m(a) \mid n$
2. 设 $m \in \mathbf{N}^*, a, b \in \mathbf{Z}, \gcd(a,m)=\gcd(b,m)=1$，则

$$\delta_m(ab)=\delta_m(a)\delta_m(b) \iff \gcd(\delta_m(a),\delta_m(b)=1)$$

3. 设 $k \in \mathbf{N}, m \in \mathbf{N}^*, a \in \mathbf{Z}, \gcd(a,m)=1$，则 $\delta_m(a^k)=\dfrac{\delta_m(a)}{\gcd(\delta_m(a),k)}$

拉格朗日定理：设 p 为素数，对于模 p 意义下的整系数多项式

$$f(x)=a_nx^n+a_{n-1}x^{n-1}+\cdots+a_0 \ (p \nmid a_n)$$

同余方程 $f(x) \equiv 0 \pmod{p}$ 在模 p 意义下至多有 n 个不同解。当 $n=0$ 时显然成立。若命题对于 $\deg f < n$ 的 f 都成立，反设存在一个满足题目条件的 f 在模 p 意义下有着至少 $n+1$ 个不同的解 $x_0, x_1, \cdots, x_n$，可设

$$f(x)-f(x_0)=(x-x_0)g(x)$$

则 $g(x)$ 在模 p 意义下 $\deg g < n$，至多有 $n-1$ 个解，且对于 $i \in [1,n]$，有

$$(x_i-x_0)g(x_i) \equiv f(x_i)-f(x_0) \equiv 0 \pmod{p}$$

又 $x_i \not\equiv x_0 \pmod{p}$，故 $g(x_i) \equiv 0 \pmod{p}$，从而 $g(x) \equiv 0 \pmod{p}$ 有至少 n 个根，矛盾。所以，命题对 n 次多项式也成立，定理获证。

原根存在定理：$m=1,2,4,p,p^\alpha,2p^\alpha$，其中 p 为奇素数，$\alpha \in \mathbf{N}^*$。

命题 **1:** 奇素数 p 有原根.
引理：设 $a,b \in \mathbf{Z}, \gcd(a,p)=\gcd(b,p)=1$，则存在 $c \in \mathbf{Z}$ 使得

$$\delta_p(c)=lcm(\delta_p(a),\delta_p(b))$$

由算术基本定理 (非严格，$\max(\alpha_j,\beta_j)>0$)，有

$$\delta_p(a)=\prod_{j=1}^{s}p_j^{\alpha^j}, \delta_p(b)=\prod_{j=1}^{s}p_j^{\beta^j}$$

令 $l=\prod_{j=1}^{s}p_j^{\alpha^j}[\alpha_j\le\beta_j], n=\prod_{j=1}^{s}p_j^{\alpha^j}[\alpha_j>\beta_j]$，记 $r=lx, t=ny$，则

$$\gcd(x,y)=1, \mathrm{lcm}(r,t)=xy$$

由阶的性质 3，

$$\delta_p(a^l)=x, \delta_p(b^n)=y$$

再由阶的性质 2，

$$\delta_p(c)=\delta_p(a^lb^n)=xy=\mathrm{lcm}(\delta_p(a),\delta_p(b))$$

回到原命题，对 $1\sim(p-1)$ 依次两两使用上述引理，可知存在 $g\in\mathbf{Z}$ 使得

$$\delta_p(g)=\mathrm{lcm}\left(\delta_p(1),\delta_p(2),\cdots,\delta_p(p-1)\right)$$

这表明

$$\delta_p(j)|\delta_p(g), j=1,2,\cdots,p-1$$

所以 $j=1,2...,p-1$ 都是方程 $x^{\delta_p(g)}\equiv 1 \pmod p$ 的根，由拉格朗日定理可知方程次数 $\delta_p(g)\ge p-1$，又由费马小定理可知 $\delta_p(g)\le p-1$，所以 $\delta_p(g)=p-1=\varphi(p)$。综上，$g$ 为模 p 的原根，奇素数都存在原根。

命题 **2:** p^α 有原根.
引理 1：存在模 p 的原根 g，使得 $g^{p-1}\not\equiv 1 \pmod{p^2}$。设 g 为模 p 的原根，易证 $g+kp$ 也是模 p 的原根。若 $g^{p-1}\equiv 1 \pmod{p^2}$，则

$$(g+p)^{p-1}\equiv C_{p-1}^0g^{p-1}+C_{p-1}^1pg^{p-1}\equiv 1+p(p-1)g^{p-2}\not\equiv 1 \pmod{p^2}$$

引理 2：若选取满足引理 1 的原根 g，对于 $\forall l\in\mathbf{N}^*$，均有

$$g^{\varphi(p^l)}=1+p^l\times k, p\nmid k$$

归纳法：$l=1$ 时由 g 的选取结论显然成立，假设上式对 l 时成立，则

$$g^{\varphi(p^{l+1})}=(g^{\varphi(p^l)})^p=(1+p^l\times k)^p\equiv 1+p^{l+1}\times k \pmod{p^{l+2}}$$

结合 $p\nmid k$ 可知引理 2 对 $l+1$ 成立，得证。

回到原命题，满足引理 1 的原根 g 是模 p^α 的原根，证明如下：
记 $\delta=\delta_{p^\alpha}(g)$，由欧拉定理和阶的性质，可知

$$\delta\mid\varphi(p^\alpha)=p^{\alpha-1}(p-1)$$
$$g^\delta\equiv 1 \pmod{p^\alpha}$$

故 $\varphi(p)=p-1\mid\delta$，可设 $\delta=p^{\beta-1}(p-1), 1\le\beta\le\alpha$。由引理 2，有

$$g^{\varphi(p^\beta)}\not\equiv 1 \pmod{p^{\beta+1}}\Rightarrow g^\delta\not\equiv 1 \pmod{p^{\beta+1}}$$

结合 $g^\delta\equiv 1 \pmod{p^\alpha}$ 可知 $\beta\ge\alpha$。综上，$\beta=\alpha$，即

$$\delta=p^{\alpha-1}(p-1)=\varphi(p^\alpha)$$

从而 g 是模 p^α 的原根。

命题 **3:** $2p^\alpha$ 有原根.
记 G 是模 p^α 的奇数原根，$\delta = \delta_{2p^\alpha}(G)$，则 $\gcd(G, 2p^\alpha) = 1$。由欧拉定理，$\delta \mid \varphi(2p^\alpha)$。而 $G^\delta \equiv 1 \pmod{2p^\alpha}$，故 $G^\delta \equiv 1 \pmod{p^\alpha}$，结合 G 是模 p^α 的原根可知 $\varphi(p^\alpha) \mid \delta$，所以 $\delta = \varphi(p^\alpha) = \varphi(2p^\alpha)$，即 G 是模 $2p^\alpha$ 的原根。

命题 **4:** 若 $m \neq 1, 2, 4, p^\alpha, 2p^\alpha$，则对任意 $a \in \mathbf{Z}, \gcd(a, m) = 1$，都有 $\delta_m(a) < \varphi(m)$。

若 $m = 2^\alpha, \alpha \in \mathbf{N}^*, \alpha \geq 3$，对于任意偶数均有 $\gcd(2k, m) \neq 1$，对于任意奇数 $a = 2k+1$，公式

$$\begin{aligned} a^{2^{\alpha-2}} &= (2k+1)^{2^{\alpha-2}} \\ &\equiv 1 + C^1_{2^{\alpha-2}}(2k) + C^2_{2^{\alpha-2}}(2k)^2 + 0 \\ &\equiv 1 + 2^{\alpha-1}k + 2^{\alpha-1}(2^{\alpha-2}-1)k^2 \\ &\equiv 1 + 2^{\alpha-1}(k + (2^{\alpha-2}-1)k^2) \\ &\equiv 1 \pmod{2^\alpha} \end{aligned}$$

而 $2^{\alpha-2} < \varphi(m) = 2^{\alpha-1}$。

若 m 不是 2 的幂，可设 $m = rt, 2 < r < t, \gcd(r, t) = 1$，由欧拉定理可知

$$a^{\varphi(r)} \equiv 1 \pmod{r}, a^{\varphi(t)} \equiv 1 \pmod{t}$$

注意到 $n > 2$ 时 $2 \mid \varphi(n)$，所以

$$\begin{aligned} a^{\frac{1}{2}\varphi(r)\varphi(t)} &\equiv 1 \pmod{r} \\ a^{\frac{1}{2}\varphi(r)\varphi(t)} &\equiv 1 \pmod{t} \end{aligned}$$

用 **CRT** 求通解得到

$$a^{\frac{1}{2}\varphi(r)\varphi(t)} \equiv 1 \pmod{rt}$$

进而有如下不等式：$\delta_m(a) \leq \dfrac{1}{2}\varphi(r)\varphi(t) = \dfrac{1}{2}\varphi(rt) = \dfrac{1}{2}\varphi(m) < \varphi(m)$。

综上，若 $m \neq 1, 2, 4, p^\alpha, 2p^\alpha$，则对任意 $a \in \mathbf{Z}, \gcd(a, m) = 1$，都有 $\delta_m(a) < \varphi(m)$，即模 m 的原根不存在。
由原根的定义，若 g 为模 m 的原根，则对于 $\varphi(m)$ 的任意素因数 p，必有

$$g^{\varphi(m)/p} \not\equiv 1 \pmod{m}$$

且 $g^{\varphi(m)} \equiv 1 \pmod{m}$。反之，由 $\delta \mid \varphi$，满足上述条件的 g 必定是模 m 的原根。
假设正整数 n 的一个原根是 g，由阶的性质 3 可知，任意满足 $\gcd(x, \varphi(n)) = 1$ 的 x，g^x 都是模 n 的原根，从而模 n 意义下一共有 $\varphi(\varphi(n))$ 个原根。可以证明在 $[1, n)$ 中的最小原根 $g_n = O(n^{0.25+\epsilon})$。
综上，预处理出 $\varphi(n)$ 的所有素因数，从小到大枚举 g，快速幂来判断是否是原根，找到最小原根后再枚举所有满足条件的 g^x 即可。

3.3 Carmichael 函数

求使得同余关系 $a^n \equiv 1 \pmod{m}$ 对所有 $\gcd(a, m) = 1$ 都成立的最小正整数 n.

4 欧几里得算法

4.1 扩展欧式算法

4.2 类欧式算法

5 同余方程

5.1 裴蜀定理

5.2 多元不定方程

5.3 线性同余方程

5.4 模逆元筛

5.5 **CRT** 定理

5.6 扩展 **CRT** 定理

5.7 阶乘取模

5.8 **N** 次同余方程

6 离散对数和剩余

6.1 BSGS 算法

给定质数 p，$a, b \in \mathbf{Z}$，计算最小的 $l \in \mathbf{N}$，满足 $a^l \equiv b \pmod{p}$.

由 $\gcd(a, p) = 1$，在模 p 意义下可以直接执行关于 a 的乘除运算。
由扩展欧拉定理，a^i 在模 p 意义下会出现循环节 c 且 $c|\varphi(p)$。可知对于原方程

$$a^l \equiv b \pmod{p}$$

至多枚举 $\varphi(p)$ 个数就能知道方程的解或无解，从而 $l \leq p$。
令 $m = \lceil \sqrt{p} \rceil$，$l = i \times m - j$，$i \in [1, m], j \in [0, m)$，$A = a^m \bmod p$，原方程变形为

$$A^i \equiv ba^j \pmod{p}$$

枚举 ba^j 并将结果存入哈希表，再枚举 A^i，如果枚举到 $i = i_0 \in [1, m]$ 时有

$$A^{i_0} \in hash$$

说明存在 j 使得

$$hash[A^{i_0}] = hash[ba^j] = j$$

答案为 $l = i_0 \times m - j$。按照顺序枚举，可以保证 $hash[ba^j]$ 对应得 j 最大，i_0 最小，从而保证 l 是最小非负整数。

6.2 扩展 BSGS 算法

给定 $a, b, m \in \mathbf{Z}$，计算最小的 $X \in \mathbf{N}$，满足 $a^X \equiv b \pmod{m}$.

若 $\gcd(a, m) \neq 1$，不能直接使用 **BSGS**，考虑将原方程化为

$$(a')^Y \equiv b' \pmod{m'}, \gcd(a', m') = 1$$

原方程等价于

$$a \cdot a^{X-1} + my = b$$

令 $d_1 = \gcd(a, m)$，由裴蜀定理知，方程有解当且仅当 $d_1 \mid b$，因此方程两边同除以 d_1：

$$\frac{a}{d_1} \cdot a^{X-1} + \frac{m}{d_1} y = \frac{b}{d_1}$$

如果此时 $d_2 = \gcd(a, \frac{m}{d_1}) \neq 1, d_2 \mid \frac{b}{d_1}$，方程两边同除以 d_2，得到

$$\frac{a^2}{d_1 d_2} \cdot a^{X-2} + \frac{m}{d_1 d_2} y = \frac{b}{d_1 d_2}$$

重复此过程，直到 $\gcd(a, \frac{m}{d_1 d_2 ... d_k}) = 1$，令 $D = d_1 d_2 ... d_k$，原方程等价于

$$\frac{a^k}{D} \cdot a^{X-k} \equiv \frac{b}{D} \ (mod \ \frac{m}{D})$$

此时应用 **BSGS** 即可。

6.3 离散对数筛

> 给定质数 p 以及正整数 g，有 q 组询问，每组询问给出整数 y，找到最小的 $x \in \mathbf{N}^*$ 使得 $g^x \equiv y \pmod p$.

假设对每一组询问单独运行 **BSGS**，时间复杂度为 $O(q\sqrt{p})$，不能够通过本题。

不妨设 g 是模 p 的原根，则 $x = ind(y)$ 一定有解。
考虑欧拉筛，记 $\pi(n)$ 为 n 及以内素数的个数，利用离散对数与对数相似的性质：

$$ind(ab) \equiv ind(a) + ind(b) \pmod{\varphi(p)}$$

只需要求 $\pi(n)$ 个离散对数就可以通过欧拉筛线性求出 n 及以内剩下的数的离散对数。
令 $x = ind(y) = i \times B + j, x \in [0, p), 0 \le j < B, 0 \le i \le \lfloor (p-1)/B \rfloor$，有

$$g^j \equiv y \times g^{-B} \pmod p$$

枚举 $j \in [0, B)$，将 g^j 插入哈希表，接着枚举 $i \in [0, \dfrac{p}{B}]$，查询时间复杂度为 $O(\pi(n)p/B)$，则求离散对数总时间复杂度为 $O(B + \pi(n)p/B)$，令 $B = \sqrt{\pi(n)p}$ 最优。
同时为了使单次询问复杂度尽可能小，取 $n = \sqrt{p}+1$，从而可以 $O(1)$ 回答 $\sqrt{p}+1$ 以内的离散对数值，对于 $y \in (\sqrt{p}+1, p)$ 的询问，考虑迭代。设当前要回答 $ind(a)$，则可令 $p = ba + c, b \le \sqrt{p}, c = p \bmod a$，发现 $ind(b), ind(b+1)$ 可以直接得到，考虑转化式子：

$$\begin{cases} a = \dfrac{p-c}{b} & \Rightarrow \operatorname{ind} a \equiv \operatorname{ind}(-c) - \operatorname{ind} b \equiv \operatorname{ind}(-1) + \operatorname{ind} c - \operatorname{ind} b, \\ a = \dfrac{p+a-c}{b+1} & \Rightarrow \operatorname{ind} a \equiv \operatorname{ind}(a-c) - \operatorname{ind}(b+1). \end{cases}$$

因为有 $min(c, a-c) \le 2a$，每次递归都会折半，故单次查询复杂度为 $O(\log p)$，总时间复杂度约为 $O(\sqrt{\pi(\sqrt{p})p} + q \log p)$。

若 g 不是模 p 的原根，考虑换底公式。令 h 是模 p 的原根，对 h 运行离散对数筛。假设询问以 g 为底 y 的离散对数，令 $k = ind_h(g), t = ind_h(y)$。由定义知：

$$h^k \equiv g, h^t \equiv y, g^x \equiv y \Rightarrow h^{kx} \equiv h^t \pmod p$$

从而 $kx \equiv t \pmod{\varphi(p)}$。解同余方程：若 $d = \gcd(k, \varphi(p)) \nmid t$ 则无解，否则令 $k' = \dfrac{k}{d}, t' = \dfrac{t}{d}, M = \dfrac{\varphi(p)}{d}$，$ind_2(y) \equiv t' \cdot (k')^{-1} \pmod M$。

6.4 N 次剩余

6.5 二次剩余

7 卢卡斯定理

8 数论分块

8.1 引理与结论

8.2 N 维数论分块

> 求值：$(\sum_{i=1}^{n}\sum_{j=1}^{m}(n \bmod i)\times(m \bmod j)) \bmod 19940417, i\neq j, 1\leq n,m\leq 10^9$

不妨设 $n\leq m$，由于 $i\neq j$，原式等价于

$$(\sum_{i=1}^{n}\sum_{j=1}^{m}(n \bmod i)\times(m \bmod j)-\sum_{i=1}^{n}(n \bmod i)\times(m \bmod i)) \bmod 19940417$$

又 $a \bmod b = a-\lfloor\frac{a}{b}\rfloor\times b$，故原式可以展开为：

$$\begin{aligned}
Ans &= \sum_{i=1}^{n}\sum_{j=1}^{m}(n \bmod i)\times(m \bmod j)-\sum_{i=1}^{n}(n \bmod i)\times(m \bmod i)\\
&= \sum_{i=1}^{n}(n \bmod i)\times\sum_{j=1}^{m}(m \bmod j)-\sum_{i=1}^{n}(n \bmod i)\times(m \bmod i)\\
&= \sum_{i=1}^{n}\big(n-\lfloor\frac{n}{i}\rfloor\cdot i\big)\times\sum_{j=1}^{m}\big(m-\lfloor\frac{m}{j}\rfloor\cdot j\big)-\sum_{i=1}^{n}\big(n-\lfloor\frac{n}{i}\rfloor\cdot i\big)\times\big(m-\lfloor\frac{m}{i}\rfloor\cdot i\big)\\
&= \big[n^2-\sum_{i=1}^{n}\lfloor\frac{n}{i}\rfloor\cdot i\big]\big[m^2-\sum_{j=1}^{m}\lfloor\frac{m}{j}\rfloor\cdot j\big]-\sum_{i=1}^{n}\Big(nm-mi\cdot\lfloor\frac{n}{i}\rfloor-ni\cdot\lfloor\frac{m}{i}\rfloor+\lfloor\frac{n}{i}\rfloor\lfloor\frac{m}{i}\rfloor\cdot i^2\Big)
\end{aligned}$$

对于形如 $\sum_{i=1}^{n}(\lfloor\frac{n}{i}\rfloor\times f(i))$ 的求和式，使用整除分块。注意到 $\lfloor\frac{n}{i}\rfloor$ 为一个不下降子序列，且呈块状分布。

对于每个以 l 为起点的块，块中每一个元素的值都是 $t=\lfloor\frac{n}{l}\rfloor$，块的终点为 $r=\lfloor\frac{n}{t}\rfloor$，因此可以 $O(1)$ 计算块中元素的和，从而 $O(T\sqrt{n})$ 计算 $\sum_{i=1}^{n}(\lfloor\frac{n}{i}\rfloor\times f(i))$，其中 T 取决于计算 $\sum_{i=l}^{r}f(i)$ 的复杂度。

对于 $\sum_{i=1}^{N}(\lfloor\frac{n}{i}\rfloor\lfloor\frac{m}{i}\rfloor\times f(i)), N=\min(n,m)$，同样考虑整除分块，对于每个以 l 为起点的块，令 $t_1=\lfloor\frac{n}{l}\rfloor, t_2=\lfloor\frac{m}{l}\rfloor$，块的终点为 $r=\min(\lfloor\frac{n}{t_1}\rfloor,\lfloor\frac{n}{t_2}\rfloor)$，块中元素和为 $\prod_{i=1}^{k=2}t_i\times\sum_{i=l}^{r}f(i)$。

该方法可以推广至 k 维整除分块：$\sum_{i=1}^{N}(\prod_{j=1}^{k}\lfloor\frac{n_j}{i}\rfloor\times f(i)), N=\min\{n_j\}$。

注：$N=\min\{n_j\}$ 的原因在于当 $i>N$ 时至少有一个 $\lfloor\frac{n_j}{i}\rfloor$ 为 0，则和为 0，可以忽略；同时排除 $t_j=0$ 的情况，避免除以 0。

8.3 向上取整

8.4 扩展数论分块

9 Dirichlet 卷积

9.1 Dirichlet 前缀和

> 给定数列 $\{a_n\}$。求出数列 $\{b_n\}$，满足 $b_k = \sum_{i|k} a_i$。$1 \le n \le 2 \times 10^7$，$0 \le a_i < 2^{32}$

首先介绍求解 k 维前缀和的一般方法。给定 k 维数组 A，大小为 $N = \prod_{i=1}^{k} N_i$，求其前缀和 S。若采用容斥原理，时间复杂度为 $O(N \times 2^k)$，当 k 较大时不够优秀。
对于一般的情形，可以将 S 表示为：

$$S_{i_1 \cdots i_k} = \sum_{i'_1 \le i_1} \cdots \sum_{i'_k \le i_k} A_{i'_1 \cdots i'_k}$$

注意到 k 维前缀和等价于进行 k 次求和。因此每次只考虑一个维度，固定所有其它维度，然后对 k 个维度分别求一维前缀和，得到的就是 k 维前缀和。这种方法称为逐维前缀和，其时间复杂度为 $O(kN)$。
逐维前缀和可以解决一类问题：考虑大小为 n 的集合的全体子集上定义的函数 f，求出其子集和函数 g，满足

$$g(S) = \sum_{T \subseteq S} f(T)$$

即 $g(S)$ 等于其所有子集 $T \subseteq S$ 上的函数值 $f(T)$ 的和，称其为子集和问题。
如果用朴素的子集枚举求解，复杂度为 $O(\sum_{i=1}^{n} \mathbf{C}_n^i \cdot 2^n = 3^n)$，而应用逐维前缀和可以达到 $O(n \cdot 2^n)$。首先，将子集和问题表示为高维前缀和的形式：注意到 S 的子集可以通过状态压缩的思想表示为长度为 n 的二进制数 s，将其每一位都看作数组下标的一个维度，则 f 就是一个 n 维数组，且每个下标状态都是 $\{0, 1\}$，即 $N = 2^n$。同时，子集的包含关系等价于下标的大小关系，即 $T \subseteq S \iff \forall i(t_i \le s_i)$。因此子集和问题等价于 n 维前缀和。

```
for(int i = 0; i < n; i++)
    for(int j = 0; j < (1 << i); j++)
        if((j >> i) & 1) f[j] += f[j ^ (1 << i)]; // g = f
```

回到原题，考虑对 b_k 产生贡献的 a_i，则有 $i \mid k$，由算数基本定理，令

$$i = \prod_{p \in \mathcal{P}} p_j^{\alpha_j}, k = \prod_{p \in \mathcal{P}} p_j^{\beta_j}$$

则 $\forall j(\alpha_j \le \beta_j)$ 成立，因此原问题等价于 $b_k = \sum_{I \subseteq K} a_i$，其中 I 表示 i 分解出的质数可重集合，K 表示 k 分解出的质数可重集合。对于单个的 k 而言，子集枚举的维度数为 K 中元素的种类数，每个维度的状态数都是 K 中对应元素的数量，K 的子集（k 的约数集）可以通过筛法得到。考虑埃氏筛，对于 n 以内每一个质数 p，标记其所有合数 k 的同时将 k 的约数 j 对应的 a_j 累加进 a_k，总的维度数为 $\pi(n)$，从而 $b_k = a'_k = \sum_{I \subseteq K} a_i$，时间复杂度为 $O(n \log \log n)$。

```
for(int i = 2;i <= n; i++) if(!vis[i])/*pi(n)*/
    for(int k = i, j = 1; k <= n; k += i, j++)
        a[k] += a[j], vis[k] = 1; // b = a
```

更直观地，假设已经筛出了 $1 \sim n$ 的所有质数，记第 i 个质数为 p_i，共有 tot 个质数：

```
for(int i = 1; i <= tot; i++)
    for(int d = 1; p[i] * d <= n; d++)
        a[p[i] * d] += a[d]; // b=a
```

这个过程称为 **Dirichlet** 前缀和。

9.2 Dirichlet 后缀和

> 给定正整数数列 $\{a_n\}$，设它的一个排列 p 的权值为 $w = \sum_{i=1}^{n} \gcd(a_1, \cdots, a_i)$，求 w 的最大值。$1 \le n \le 10^5, 1 \le a_i \le 2 \times 10^7$

9.3 DGF 与卷积

10　积性函数筛

11 莫比乌斯函数

Part II
图论

1 图论基础

2 最短路问题

3 生成树问题

4 连通性问题

4.1 基本定义

4.2 强连通分量

给定一个 n 个点，m 条边的有向图，求出这个图点数大于 1 的强连通分量个数。

4.2.1 Kosaraju 算法

Kosaraju 算法使用两次 DFS 求出强连通分量。
第一次 DFS，选取任意顶点作为起点，后序遍历所有未访问过的顶点并标号。
第二次 DFS，对于反向后的图，以原图标号最大的顶点作为起点开始 DFS。
这样遍历到的顶点集合就是一个强连通分量。对于所有未访问过的结点，选取标号最大的，重复第二次 DFS 过程直到所有点均被访问过。

算法正确性证明如下：

1. 必要性：在第二次 DFS 中，记反图 DFS 树的起点为 v，结点标号记为 $t(v)$，假设存在路径 $v \rightsquigarrow u$，则原图中存在路径 $u \rightsquigarrow v$ 且 $t(v) > t(u)$。考虑原图的 DFS 树，若 v 和 u 在同一子树中，则由 $t(v) > t(u)$，v 是 u 的祖先，原图中 v 可达 u。
   若 v 和 u 不在同一子树中，假设先访问 v 所在子树，则 $t(v) < t(u)$，矛盾；假设先访问 u 所在子树，则由路径 $u \rightsquigarrow v$，u 是 v 的祖先，同样与 $t(v) > t(u)$ 矛盾。
   综上，以 v 的根的反图 DFS 树中任意结点均与 v 在原图中互相可达，由传递性可得这些点位于原图的同一强连通分量中。

2. 充分性：易知若原图上两点 u, v 位于同一强连通分量，则反图上也位于同一强连通分量。而根据 DFS 的性质，如果 v 与 u 强连通，那么由 v 开始的 DFS 必定能搜到 u，不存在 $u \in [v], u \notin T_v$ 的情况。故该算法能求出所有的极大强连通分量。

一种理解思路为：记原图缩点后形成的有向无环图 G 的拓扑序意义下的一对源汇点分量为 S, T，显然从 T 中任意一个点 DFS 都能且只能访问到 T 内部所有点，但是从原图并不能轻易地找到当前 T 中的点，只能通过后序列表找到 S 中的点。考虑反图缩点形成的有向无环图 G'，易知 G' 是 G 的逆，则 G' 的汇点就是 G 的源点。因此，第一次在原图上 DFS 并将其后序列表 L 求出，第二次在反图上 DFS，每次起点为 L 中最后一个未访问的结点，从而 DFS 不能到达其他任何连通分量而只能遍历当前分量中的所有结点。后序列表的最后一个点（标号最大）即 G 的源点分量中的一点，同时也是 G' 的汇点分量中的一点。

由于有向图的强连通分量不会因为有向图求逆而改变，Kosaraju 算法也可以第一次对反图操作，第二次对原图操作，正确性证明同理。

4.2.2 Tarjan 算法

4.3 2-SAT 问题

5 环计数问题

6 欧拉图问题

7 竞赛图问题

8 平面图问题

8.1 基本定义

8.2 广义串并联图

9 弦图问题

10 网络流

10.1 基本定义

给定有源汇网络 (G,s,t)，定义容量函数和流函数 $c,f:\mathbf{E}\to\mathbb{Z}|\mathbb{R}$，其满足如下条件：

- 容量限制：$f(u,v)\le c(u,v)$，等号成立时称为满流，$f(u,v)=0$ 时称空流。
- 斜对称：$f(u,v)=-f(v,u)$，(v,u) 为反向边，$(u,v)\in\mathbf{E}^+,(v,u)\in\mathbf{E}^-$。
- 流量守恒：$\forall u(u\in\mathbf{V},u\neq s,u\neq t)$，净流量为 0（边不存在时 $f=0$）：

$$f(u)=\sum_{v\in\mathbf{V}}f(u,v)-\sum_{v\in\mathbf{V}}f(v,u)=0$$

有源汇网络的相关定义如下：

- 根据斜对称和流量守恒，有 $|f|=f(s)=-f(t)$，称为当前流 f 的流量。最大流量为所有合法 f 中 $|f|$ 的最大值。
- 定义流 f 在网络 G 上的残量网络 $G_f=(\mathbf{V},\mathbf{E}_f)$ 为容量函数 $c_f=c-f$ 的网络。根据容量限制，有 $c_f(u,v)\ge 0$，等号成立时视 (u,v) 在 G_f 上不存在。
- 定义增广路为残量网络 $G_f=(\mathbf{V},\mathbf{E}_f)$ 上一条从源点 s 到汇点 t 的路径。
- 将 $\mathbf{V}$ 分成不相交的两个点集 S,T 且 $s\in S,t\in T$，这种划分方式称为 $s-t$ 割，定义其容量为 $c_c=\sum_{u\in S}\sum_{v\in T}c(u,v)$，流量为 $f_c=\sum_{u\in S}\sum_{v\in T}f(u,v)$。
- 当 u,v 属于不同点集时，称 (u,v) 为割边。当 c_c 最小时称最小割，记为 $\|S,T\|$。

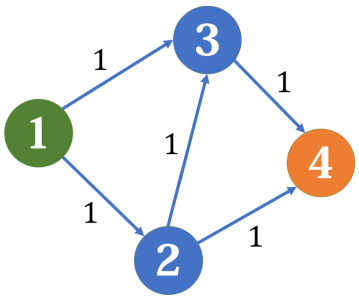
10.2 最大流

> 给定网络 $G=(\mathbf{V},\mathbf{E})$ 和源汇点，求最大流量 (Max Flow)。

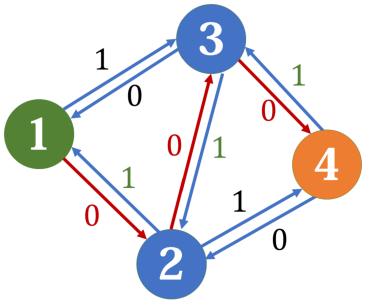
10.2.1 Ford-Fulkerson 增广

考虑贪心：找到残量网络 G_f 上的一条增广路 P 并为 P 上每一条边增加 $c_f(P)=\min\limits_{(u,v)\in P}c_f(u,v)$ 的流量，此过程称为增广，重复增广直到 G_f 上不存在增广路。

显然，如果增加的流量大于该值，一些边将不满足容量限制，而根据能流满就流满的思想，增加的流量也不应小于该值。这种贪心不一定是正确的，考虑下图：



令 $s=1,t=4$, 首先选择增广路 $(1,2,3,4)$ 会导致残量网络上没有其他的增广路，从而 $|f|=1$，但实际上的最大流为选择增广路 $(1,3,4)$ 和 $(1,2,4)$，此时 $|f|=2$。
为了保证正确性，对每条边建立满足斜对称且能快速访问的反向边，目的是支持反悔。具体地，在为当前边 $(u,v)\in P$ 增加流量 $c_f(P)$ 时，给其反边 (v,u) 的容量加上 $c_f(P)$。



此时选择增广路 $(1,2,3,4)$ 和 $(1,3,2,4)$，$|f|=2$，边 $(2,3)$ 与其反边相互抵消，等价于选择增广路 $(1,3,4)$ 和 $(1,2,4)$。因此，加入反悔意味着 G_f 中的每条可能的增广路都可以被访问，与访问顺序无关。对于反向边的存储，采用链式前向星和成对变换的技巧，即将每条边与它的反向边按编号连续存储，分别为 k 和 $k\oplus 1$，满足 $2\mid k$。

10.2.2 最大流最小割定理

> 证明：对于给定有源汇网络，其最大流等于最小割，即 $|f|=\|S,T\|$。

上述贪心的正确性证明和对最大流最小割定理的证明等价。
首先有引理：对于给定网络 G，任取一个流 f 和一个割 $\{S,T\}$，总是有 $|f|\le ||S,T||$，等号成立当且仅当所有 S 到 T 的割边为满流且所有 T 到 S 的割边为空流，证明如下：

$$
\begin{aligned}
|f| &= f(s)\\
&= \sum_{u\in S} f(u)\\
&= \sum_{u\in S}\left(\sum_{v\in V} f(u,v)-\sum_{v\in V} f(v,u)\right)\\
&= \sum_{u\in S}\left(\sum_{v\in T} f(u,v)+\sum_{v\in S} f(u,v)-\sum_{v\in T} f(v,u)-\sum_{v\in S} f(v,u)\right)\\
&= \sum_{u\in S}\left(\sum_{v\in T} f(u,v)-\sum_{v\in T} f(v,u)\right)+\sum_{u\in S}\sum_{v\in S} f(u,v)-\sum_{u\in S}\sum_{v\in S} f(v,u)\\
&= \sum_{u\in S}\left(\sum_{v\in T} f(u,v)-\sum_{v\in T} f(v,u)\right)\\
&\le \sum_{u\in S}\sum_{v\in T} f(u,v)(empty)\\
&\le \sum_{u\in S}\sum_{v\in T} c(u,v)(full)\\
&= \|S,T\|
\end{aligned}
$$

回到原定理，假设某一轮增广后得到流 f 使得 G_f 上不存在增广路，此时记 G_f 上从 s 出发可以到达的点集为 S，$T = \mathbf{V} - S$，显然 $\{S, T\}$ 为 G 的一个割（源汇点不连通）。在残量网络上讨论，此时不存在从 S 到 T 的割边，所以有：

$$\forall u \in S, \forall v \in T, c_f(u, v) = 0$$

将这些边分为存在于原图中的边和反向边两种情况讨论：

- $(u, v) \in \mathbf{E}$，此时 $c_f(u, v) = 0$，有 $c(u, v) = f(u, v)$，满流；
- $(v, u) \in \mathbf{E}$，此时 $c_f(u, v) = c(u, v) - f(u, v) = f(v, u) = 0$，空流。

从而流函数 f 满足引理的取等条件，为最大流，贪心正确，又任意流小于任意割，反证得此时 $\{S, T\}$ 为最小割，最大流最小割定理成立。

10.2.3 Edmonds-Karp 算法

EK 算法使用 BFS 求 G_f 中的最短增广路并应用上述贪心，其时间复杂度为 $O(|\mathbf{V}||\mathbf{E}|^2)$。

10.2.4 Dinic/ISAP 算法

Dinic 算法的核心思想是分层图以及相邻层之间增广，从而批量处理增广路。考虑增广前对 G_f 做 BFS 分层，记结点 u 的层数为 $\text{lev}(u)$，对于 $\forall u \in G_f$，暂时删除残量网络中所有满足 $\text{lev}(u) \geq \text{lev}(v)$ 的边 (u, v)，形成 DAG。此时分层图称为层次图，记为：

$$G_L = (\mathbf{V}, \mathbf{E}_L), \mathbf{E}_L = \{(u, v) | (u, v) \in \mathbf{E}_f, \text{lev}(u) + 1 = \text{lev}(v)\}$$

此时层次图中所有的增广路均为最短增广路，使用 DFS 遍历所有的增广路，满足批量求增广路的需求。记增广流为对若干条增广路增广后得到的流，如果在 G_L 上找到一个极大的增广流 f_b 使得 G_L 上不存在 $|f'| \geq |f_b|$，称 f_b 为 G_L 的阻塞流，将其累加进 f。将以上过程称为一次分层，重复此过程直到新的 G_L 中不存在增广路，即源汇点不连通，此时 f 为最大流。注意，在一次分层中删除的边集 $\mathbf{E}' = \mathbf{E}_f - \mathbf{E}_L$ 在之后的分层中可能还会用到，这种删除等价于在 DFS 过程中忽略边集 $\mathbf{E}'$。

注意到在 G_L 上 DFS 时，如果结点 u 同时具有大量入边和出边，并且 u 每次接受来自入边的流量时都遍历出边表来决定将流量传递给哪条出边，则局部时间复杂度最坏可达 $O(|\mathbf{E}|^2)$。为避免这一缺陷，如果某一时刻已知边 (u, v) 已经增广到极限，即边 (u, v) 已无剩余容量或 v 的后侧已增广至阻塞，则 u 的流量没有必要再尝试流向出边 (u, v)。据此，对于每个结点 u，维护 u 的出边表中第一条还有必要尝试的出边/指针，称其为当前弧，下一次 DFS 从当前弧开始遍历所有出边，跳过所有不可行边，称这个做法为当前弧优化，是算法不可或缺的一部分。Dinic 算法总的时间复杂度为 $O(|\mathbf{V}|^2|\mathbf{E}|)$。

10.2.5 预流推进算法

10.3 最小割树

将 $\mathbf{V}$ 分成互不相交的两个点集 S,T，当 $u \in S, v \in T$ 时将该割称为关于 u,v 的割，记为 $\{u,v\}$。称 $\{u,v\}$ 的最小割为所有关于 u,v 的割中容量最小的割，记为 $\|u,v\|$。给定一张带非负权的 n 阶无向图，边数为 m，有 q 个询问，每个询问给定一个非负整数 x，求图中有多少个无序点对的最小割的容量不超过 x。

首先新建一张无边 n 阶无向图 G'，然后在原图 G 中任意选两个点 s,t，求出其最小割 $c = \|s,t\|$，并在 G' 中在 s,t 之间连一条边权为 c 的无向边。递归地在 s 所属的点集和 t 所属的点集中重复上述过程，直到当前集合中只剩一个点。这个过程所构建的新图 G' 为一棵树，称为最小割树，树上任意两个点在原图中的对应点之间的最小割值等于这两个点的树上路径中边权的最小值，下面证明该结论：

回到原题，预处理出最小割树后，问题转化为：给定一棵 n 阶非负权无根树，定义两点之间的距离为两点之间路径中的最小边权，每个询问给定非负整数 x，求距离小于等于 x 的无序点对数量。考虑 Kruskal 重构树：将边权后放入大根堆，然后依次取出，在并查集上合并每条边两边端点所在的集合，并维护集合大小。设合并的两点 $\{x,y\}$ 所在的集合大小为 s_x, s_y，则有 $s_x \times s_y$ 个无序点对的距离为该边边权。设距离为 w 的无序点对数量总共有 p_w 个，则对于每次询问，通过在 p 对应的前缀和数组上二分得到答案。总的时间复杂度为 $O(n^3m + n\log n + q\log n)$。

10.4 有界流

10.5 费用流

给定网络 $G = (\mathbf{V}, \mathbf{E}, c, w)$，若单位流量费用 $w(u,v)$ 也满足斜对称性，则将网络中总花费 $f \times w$ 最小的最大流称为最小费用最大流，即最大化 $|f|$ 的前提下最小化

$$\sum_{(u,v)\in\mathbf{E}} f(u,v) \times w(u,v)$$

10.5.1 SSP 算法

10.5.2 原始对偶算法

10.5.3 ZKW 费用流

10.5.4 带负圈费用流

10.6 常见模型

10.6.1 点边转化

10.6.2 分数规划

10.6.3 模拟费用流

详见 IOI-2022.pdf，P203

10.6.4 最大权闭合图

11 序理论

11.1 基本定义

11.2 拓扑排序

11.3 Dilworth 定理

12 图的着色

13 图的匹配

13.1 基本定义

13.2 二分图最大匹配

13.3 二分图最大权匹配

13.4 树上匹配问题

13.5 一般图最大匹配

14 树论基础

14.1 树的遍历

14.2 树的直径

14.3 树的中心

14.4 树的重心

14.5 最近公共祖先

14.5.1 倍增法

14.5.2 欧拉序

14.5.3 DFS 序

14.6 树上差分

14.7 树的同构

14.8 树上分治

15 树链剖分

15.1 基本定义

当没有特别指出时，树链剖分等价于重链剖分。下面给出一些描述性定义：

- 重结点：子树结点数目最多的结点，记为 $\mathrm{son}(u)$；
- 轻结点：父亲节点中除了重结点以外的结点；
- 重边：父亲结点和重结点连成的边；
- 轻边：父亲节点和轻节点连成的边；
- 重链：由多条重边连接而成的路径，记为 C_i；
- 轻链：由多条轻边连接而成的路径；
- 端点：一条重链的顶端结点/起点，记为 $\mathrm{top}(u)$。

如果有多个候选重结点，任选一个。叶子结点没有轻/重结点，但可以作为重链的起点。对于长链剖分，其重结点表示其子结点中子树深度最大的子结点，其余定义描述不变。

15.2 重链剖分

考虑使用两次 DFS。第一次 DFS 记录每个结点的父结点 $\mathrm{fa}(u)$，深度 $\mathrm{dep}(u)$，子树大小 $\mathrm{siz}(u)$ 以及重结点 $\mathrm{son}(u)$。第二次 DFS 记录每个结点所在重链的端点 $\mathrm{top}(u)$，重边优先遍历时的 DFS 序 $\mathrm{dfn}(u)$ 以及 DFS 序对应的结点编号 $\mathrm{rnk}(x)$，即 $\mathrm{rnk}(\mathrm{dfn}(u)) \leftarrow u$。

观察到树上每个结点都属于且仅属于一条重链，即所有的重链将整棵树完全剖分，证明如下：

1. 由重链的构造方法，每个结点至多有一个重结点，所以从任意一个结点出发沿着重边向下，其路径是唯一的。
2. 对于 $\forall u \in \mathbf{T}$，如果不存在 i 使得 $u, \mathrm{fa}(u) \in C_i$，则 $u = \mathrm{top}(u)$，又 $\mathrm{root}(\mathbf{T})$ 必然是端点，从而任意结点都至少属于一条重链（覆盖性）。
3. 一个结点至多是一个结点的重结点，所以任意一个结点至多位于一条重链上，即 $C_i \bigcap C_j = \varnothing, i \neq j$（互斥性），从而每个结点都属于且仅属于一条重链。

在剖分时重边优先遍历，则重链内的 DFS 序是连续的，即按 DFN 排序后的序列即为剖分后的链。可以发现，当向下经过一条轻边时，所在子树的大小至少会除以二，证明如下：

记 $\mathrm{fa}(v) = p$ 且 (p, v) 为轻边，假设 $\mathrm{siz}(v) > \dfrac{\mathrm{siz}(p)}{2}$，则 $2\,\mathrm{siz}(v) > \mathrm{siz}(p) = 1 + \sum\limits_{(p,u)\in\mathbf{T}} \mathrm{siz}(u)$

即

$$\mathrm{siz}(v) > 1 + \sum_{u \neq v} \mathrm{siz}(u)$$

上式说明 v 的子树大小大于所有其它子结点子树之和，因而必然严格大于每一个其它子结点的子树大小，与 (p, v) 为轻边的假设矛盾。
因此，对于树上的任意一条路径，把它拆分成从 LCA 分别向两边往下走，分别最多走 $O(\log n)$ 次，从而树上的每条路径都可以被拆分成不超过 $O(\log n)$ 条重链。

15.2.1 路径上维护

给定树 **T**，多次询问树上两点间路径权值和。

15.3 长链剖分

16 树分解问题

17 基环树问题

18 虚树问题

19 支配树问题

20 Prüfer 序列

Part III
线性代数

1 线性基

1.1 定义与性质

称代数系统 $< V, +, \cdot, \mathbf{P} >$ 是 V 关于 $+, \cdot$ 构成 $\mathbf{P}$ 上的线性空间，$\mathbf{P}$ 为线性空间的基域，V 中元素称为向量，$\mathbf{P}$ 中的元素称为标量。V 的一个极大线性无关组为一组线性基，简称基。记 θ 为加法群的零元，规定线性空间 $\{\theta\}$ 的基为空集。可以证明任意线性空间均存在线性基，定义线性空间 V 的维数为线性基的元素个数，记作 $\dim V$。

线性基具有如下性质：

1. 对于有限维线性空间 V （可以推广至无限线性空间），设 $n = \dim V$，则：
   - V 中的任意 $n+1$ 个向量线性相关；
   - V 中任意 n 个线性无关的向量均为 V 的基；
   - 若 V 中任意向量均可被向量组 $(a_1, a_2, ..., a_n)$ 线性表示，则其是 V 的一个基；
   - V 中任意线性无关向量组均可通过插入若干向量使得其成为 V 的一个基。
2. 令 V_1, V_2 是关于 $\mathbf{P}$ 的有限维线性空间，且 $V_1 + V_2, V_1 \bigcap V_2$ 也是有限维的，则：
   - $\dim V_1 + \dim V_2 = \dim(V_1 + V_2) + \dim(V_1 \bigcap V_2)$；
   - 若 $(a_1, ...a_n)$ 是 V_1 的一组基，$(b_1, ...b_m)$ 是 V_2 的一组基，则 $(a_1, ...a_n, b_1, ...b_m)$ 是 $V_1 + V_2$ 的一组基。

从而通过线性基可以实现如下功能：

1. 求给定向量组的秩；
2. 对给定的向量组，找到一组极大线性无关组（或其张成的线性空间的一组基）；
3. 向给定的向量组插入某些向量，在插入操作后的向量组中找到一组其张成的线性空间的一组基；
4. 对找到的一组基，判断某向量能否被其线性表出；
5. 对找到的一组基，求其张成的线性空间中的特殊元素（如最大元、第 k 小元等）。

1.2 异或线性基

一般将 n 维布尔域线性空间下 $\mathbf{Z}_2^n$ 的线性基称为异或线性基，可以证明代数系统 $<\mathbf{Z}_2^n, \oplus, \&, \mathbf{Z}_2>$ 是线性空间。
回到原题，可以根据给定的 m 个数所转化的布尔序列 $X = \{x_1, ...x_m\}$ 构建一组异或线性基 $B = \{b_1, ...b_n\}$，满足如下性质：

1. B 中任意非空子集异或和不为 0；
2. 对 X 中任意元素都可以表示为 B 中若干元素的异或和；（数乘为与且域 $\mathbf{Z}_2$ 中元素为 $\{0, 1\}$）
3. 对任意满足以上两条的集合 B'，其元素个数不少于 B 的元素个数。

异或线性基可以实现如下功能：

1. 判断一个数能否表示成某数集子集的异或和；
2. 求一个数表示成某数集子集异或和的方案数；
3. 求某数集子集的最大/最小/第 k 大/第小 k 异或和；
4. 求一个数在某数集子集异或和中的排名。

1.2.1 最大异或和

> 给定 m 个整数（数字可能重复），在这些数中选取任意个使得他们的异或和最大，求该最大异或和的值。

有两种构造异或线性基的方法，消元法和贪心法。消元法即将 m 个布尔序列构成二进制矩阵通过高斯消元法消成线性基。其步骤如下：

1. 从高到低位枚举（等价于按列枚举）。
2. 确定主元（某一行对应的值）。假设枚举到了第 i 位，找到一个该位为 1 的数，将其置换到第 i 行。
3. 对其他的数：假如该数第 i 位为 1，那就让这个数异或主元，让这一位变成 0，否则不操作。

例如对于序列 $\{7, 5, 9, 2\}$，有如下行列变换：

$$\begin{bmatrix}0&1&1&1\\0&1&0&1\\1&0&0&1\\0&0&1&0\end{bmatrix} \xrightarrow{\text{swap}(R_1,R_3)} \begin{bmatrix}1&0&0&1\\0&1&0&1\\0&1&1&1\\0&0&1&0\end{bmatrix} \xrightarrow{R_3=R_3\oplus R_2} \begin{bmatrix}1&0&0&1\\0&1&0&1\\0&0&1&0\\0&0&1&0\end{bmatrix} \xrightarrow{R_4=R_4\oplus R_3} \begin{bmatrix}1&0&0&1\\0&1&0&1\\0&0&1&0\\0&0&0&0\end{bmatrix}$$

从而 $B = \{9, 5, 2\}$。易证 B 中元素严格递减，且二进制矩阵是一个行简化阶梯形矩阵。这意味着基中所有元素异或和即是最大异或和。
相较于消元法，贪心法可以做到在线插入元素并在线更新线性基。其过程如下：

1. 令 p 表示原序列的线性基数组，p_i 表示第 i 位的向量。初始时基为空集，$p_i = 0, i \in [0, Bits]$；

2. 当插入一个数 x 时，从高到低位枚举 $x_{(2)}$，如果 $x_{(2)}$ 的第 i 位是 1，判断 p_i 是否存在：
   - 若不存在则插入并退出，即 $p_i = x$；
   - 否则将 x 异或上 p_i 并枚举 $x'_{(2)}$ 的下一位。

从而 $B = \{p_i \mid p_i \neq 0\}$。例如对于序列 $\{7, 5, 9, 2\}$，有如下插入操作：

$$p_{3\to 0} = \begin{bmatrix}0\\0\\0\\0\end{bmatrix} \xrightarrow{\text{insert}(7), p_2=7} \begin{bmatrix}0\\7\\0\\0\end{bmatrix} \xrightarrow{\text{insert}(5), p_1=2} \begin{bmatrix}0\\7\\2\\0\end{bmatrix} \xrightarrow{\text{insert}(9), p_3=9} \begin{bmatrix}9\\7\\2\\0\end{bmatrix} \xrightarrow{\text{insert}(2), p_0=0} \begin{bmatrix}9\\7\\2\\0\end{bmatrix}$$

假设存在 i, j, k 使得 $p_i \oplus p_j \oplus p_k = 0$，则 p_k 不会被插入，从而 B 中任意非空子集异或和不为 0 且 B 中元素不能被其他元素表出。
由异或性质，如果 $x \neq 0$ 最终没有插入 B，则 $x \oplus \{p_i | x_i = 1\} = 0 \Rightarrow \{p_i | x_i = 1\} = x$，即基中若干元素可以异或出 x，从而原序列中的每一个数都可以通过线性基表出。
此外，$p_i \oplus p_j \oplus p_k = x \iff p_i \oplus p_j \oplus x = p_k$，即插入顺序会影响贪心法得出的基，又因为每个插入的数对应着一个二进制位，所以 x 与 p_k 相排斥只会影响一个元素，即线性基里的数可能不同，但是总数肯定是一定的。
贪心法求最大值时，需要对线性基求前缀最大值（从高到低位），即枚举到第 i 位时，需要用前缀的异或最大值去异或 p_i，更新最大值。

```
for(int i = Bits; i>= 0; i--) ans = max(ans, ans ^ p[i]);
```

1.2.2 求第 k 小异或和

> 给由 n 个数组成的一个可重集 S，每次给定一个数 k，求集合 $T \subseteq S$ 的异或和，其中集合 T 满足：在 S 的所有非空子集的不同的异或和中，其异或和 $T_1 \oplus T_2 \oplus \ldots \oplus T_{|T|}$ 是第 k 小的。$1 \leq n, m \leq 10^5, 0 \leq S_i, k \leq 2^{50}$

若用贪心法得到 S 的线性基 B，就再用消元法处理成 B'，使得基中任何一个二进制位至多有一个元素是 1，从而可以不重复使用任意一个元素就张成线性空间。
设 $k_{(10)} = (k_M \ldots k_0)_{(2)}, p_i \in B'$，则第 k 小异或和为 $\sum_{i=0}^{Bits} k_i \cdot p_i$。原序列中有 0 则 $k \leftarrow k-1$。

1.3 实数线性基

2 特征根

2.1 基本定义

2.2 斐波那契数列

2.3 带常线性递推

> 有一个 1 行，$n+1$ 列的棋盘，编号为 0 到 n。初始时刻，在位置 m 有一颗棋子。如果某一秒棋子不位于 n，那么它将有 p 的概率向左移动一格，$1-p$ 的概率向右移动一格；否则，棋子向左移动一格。求期望多少秒之后棋子能够到达 0。答案对于 10^9+7 取模。$1 \le m \le n \le 10^9$，保证 p 和答案是有理数。

设 E_i 表示当棋子位于位置 i 时期望走到 0 的时间，则有 $n+1$ 个方程：

$$E_i = \begin{cases} 0, & i = 0, \\ pE_{i-1} + (1-p)E_{i+1} + 1, & 0 < i < n, \\ E_{n-1} + 1, & i = n. \end{cases}$$

考虑第 $i+1$ 个方程，变形得

$$E_i - E_{i-1} = \frac{1-p}{p}(E_{i+1} - E_i) + \frac{1}{p}$$

令 $F_i = E_i - E_{i-1}$，则 $F_i = \frac{1-p}{p}F_{i+1} + \frac{1}{p}, E_x = \sum_{i=1}^{x} F_i$ ，根据特征根法得出通项公式为：

$$F_i = (\frac{1-p}{p})^{n-i}(F_n + \frac{1}{1-2p}) - \frac{1}{1-2p}, F_n = 1, p \neq \frac{1}{2}$$

当 $p \neq \frac{1}{2}$ 时，这是等比数列求和，否则 $F_i = F_{i+1} + 2$，为等差数列求和。

此外，还可以考虑构造伴随矩阵并利用矩阵快速幂求解 E_i。对于第 $i+1$ 个方程，变形得带常数的齐次线性递推式

$$E_{i+1} = \frac{1}{1-p}E_i - \frac{p}{1-p}E_{i-1} - \frac{1}{1-p}, p \neq 1$$

令 $v_{n+1} = Mv_n \Rightarrow v_n = M^{n-1}v_1$，有

$$v_i = \begin{bmatrix} E_i \\ E_{i-1} \\ 1 \end{bmatrix}, v_1 = \begin{bmatrix} E_1 \\ 0 \\ 1 \end{bmatrix}, M = \begin{bmatrix} \frac{1}{1-p} & -\frac{p}{1-p} & -\frac{1}{1-p} \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

通过 $E_n = E_{n-1} + 1$ 解出 E_1，继而求出 $E_m, 1 \le m \le n$。当 $p = 1$ 时 $E_i = E_{i-1} + 1$。

Part IV
组合数学

1 基本原理

1.1 计数原理

1.2 排列组合

1.3 鸽巢原理

1.4 容斥原理

2 卡特兰数

3 斯特林数

4 伯努利数

5　分拆数

6 范德蒙德卷积

7 Pólya 计数

8 图论计数

9 生成函数

Part V
计算几何

1 几何基础

Part VI
概率论

1 期望问题

1.1 基本定义

1.2 条件期望

1.3 经典模型

1.3.1 随机游走

详见 IOI-2019.pdf，P17

Part VII
博弈论

1 公平组合博弈

1.1 基本定义

公平组合博弈（Impartial Game）指的是满足如下条件的博弈：

- 两个玩家交替行动、信息完全、没有概率成分、局面有限且终局明确的博弈（组合博弈）；
- 在任意确定状态下，所有参与者可选择的行动完全相同，仅取决于当前状态，与身份无关（公平博弈）；
- 博弈中的同一个状态不可能多次抵达，博弈以参与者无法行动为结束，且博弈一定会在有限步后以非平局结束（非反常博弈）；
- 交换两个玩家的身份，收益函数不变（对称博弈）。

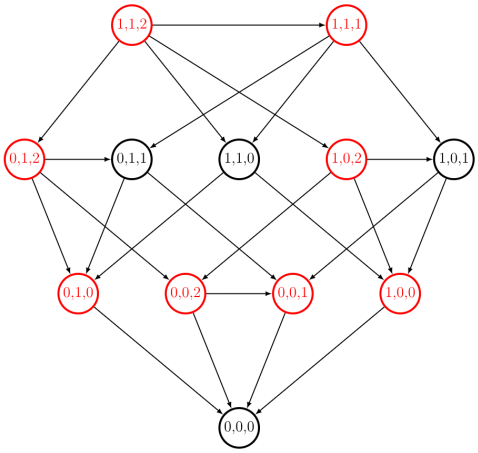
在一个公平组合游戏中，每个玩家是否有必胜策略，只取决当前游戏所处的状态而与玩家的身份无关。因此，所有状态可以分为先手必胜状态和先手必败状态，分别记为 $\mathcal{N}$ 态和 $\mathcal{P}$ 态。这个定义适用于所有公平组合游戏，从而可以归纳地将正常规则的公平组合游戏中所有状态标记为必胜状态和必败状态：

- 没有后继状态的状态是必败状态 $\mathcal{P}$；
- 一个状态是必胜状态 $\mathcal{N}$ 当且仅当存在至少一个它的后继状态为必败状态 $\mathcal{P}$；
- 一个状态是必败状态 $\mathcal{P}$ 当且仅当它的所有后继状态均为必胜状态 $\mathcal{N}$。

1.2 Nim 游戏

> 共有 n 堆石子，第 i 堆有 a_i 枚石子。两名玩家轮流取走任意一堆中的任意多枚石子，但不能不取。取走最后一枚石子的玩家获胜。判断先手是否必胜。

容易验证，Nim 游戏是正常规则的公平组合游戏，其局面/状态（双方玩家过往的行动、已经实现的随机变量值、双方已知信息的内容等）可能的变化可以用博弈图来描述：将每一个可能的状态都看作是图中的一个结点，并将状态向它的后继状态（即通过一次操作可以达到的状态）连边，得到一个有向无环图。博弈图是无环的，因为 Nim 游戏中，每次操作，石子的总数量都是严格减少的。例如，对于初始局面有 3 堆石子，且每堆石子的数量分别为 $1, 1, 2$ 的 Nim 游戏，可以绘制如下的博弈图：



其中红色结点表示必胜状态，黑色结点表示必败状态。
假设在某一时刻第 i 堆有 $A_i(0 \le A_i \le a_i)$ 枚石子，定义其 Nim 和为 $k = A_1 \oplus A_2 \cdots \oplus A_n$，其状态为必败状态 $\mathcal{P}$ 当且仅当 Nim 和为 0，证明如下：

1. 如果对于 $\forall i \in [1, n], A_i = 0$，则该状态没有后继状态，且 $k = 0$，命题成立。
2. 如果 $k \neq 0$，构造一个合法移动，使得后继状态为必败状态：由归纳假设，只需要证明存在后继状态满足 $k' = 0$ 即可。设 k 的二进制表示中，最高位的 1 是第 d 位，则一定存在某个 A_i，使得其二进制第 d 位是 1。对于相应的石子堆，一定有 $A_i > A_i \oplus k$。考虑将该石子堆减到 $A_i' = A_i \oplus k$，此时 $k' = A_1 \oplus \cdots A_i' \oplus \cdots \oplus A_n = k \oplus k = 0$。
3. 如果 $k = 0$，需要证明该状态是必败状态。由归纳假设可知，只要证明它的所有后继状态的 Nim 和都不是 0。这是必然的，任何合法移动将 A_i 变为 $A_i' \neq A_i$，都会使 $k' = k \oplus A_i \oplus A_i' \neq 0$。

1.3 SG 理论

Sprague-Grundy 理论指出，所有公平组合游戏都等价于单堆 Nim 游戏。当游戏由若干个相互独立的子游戏组成时，游戏的状态判定可以通过计算子游戏的 SG 函数值的 Nim 和来完成。

1.4 经典模型

Part VIII
字符串

1 前缀数组

2　Z 函数

3 字符串哈希

4　BM 匹配算法

5 字典树

6 AC 自动机

7 后缀数组

8 后缀自动机

8.1 广义 SAM

9 后缀字典树

10 后缀平衡树

11 Manacher 算法

12 回文自动机

13 子序列自动机

14 Lyndon 分解

15 ML 算法

Part IX
数据结构

1 并查集

1.1 基本定义

1.2 路径压缩

1.3 按秩合并

1.4 带权并查集

1.5 种类并查集

1.6 可撤销并查集

2 哈希表

3 栈与队列

4 链表

5　稀疏表

6 珂朵莉树

7 树状数组

8 可并堆

9 线段树入门

10 平衡树入门

10.1 二叉搜索树

10.2 Treap 树

10.3 Spaly 树

10.4 笛卡尔树

10.4.1 基本定义

10.4.2 广义笛卡尔树

10.5 替罪羊树

11 K-D 树

12 析合树

13 分块入门

13.1 基本思想

13.2 块状数组

13.3 SQRT 树

13.4 Bitset 相关

14 莫队算法

15　整体二分

16　CDQ 分治

17　可持久化

18 可追溯化

详见 Retroactive-DS.pdf

19 树套树

20 动态树

20.1 Link-Cut Tree

20.2 欧拉回路树

20.3 全局平衡二叉树

20.4 自适应 Top-Tree

Part X
动态规划

1 背包 DP

详见 pack.pdf

1.1 0-1 背包

> 有 N 件物品和一个容量为 V 的背包。放入第 i 件物品耗费的费用/容量是 C_i，得到的价值是 W_i。求解将哪些物品装入背包可使价值总和最大。

将此问题分解为子问题：将前 i 件物品放入容量为 v 的背包中得到的最大值，从而每次只用考虑第 i 件物品的策略（不放/放），得到转移方程：

$$f_{i,v} = \max\{f_{i-1,v}, f_{i-1,v-C_i} + W_i\}$$

注意到方程中第 i 项仅与第 $i-1$ 项有关，为了保证在第 i 次循环中推 f_v 时 f_v 和 f_{v-C_i} 分别保存其在 $i-1$ 项的值，需要以 $v \leftarrow V \cdots C_i$ 的递减顺序计算 f_v，从而有：

$$f_v = \max\{f_v, f_{v-C} + W\}, v \leftarrow (V \to C)$$

该过程简记为 $\mathrm{ZOP}(f, C, W)$，即原问题等价于 $\{\mathrm{ZOP}(f, C_i, W_i) | i \leftarrow (1 \to N)\}$。一个简单的常数优化是，在第 i 次 ZOP 中，v 的范围是 $V \to \max\{C_i, V - \sum_{j=i}^{N} W_j\}$。

考虑此时 f_v 的含义，其表示容量为 v 的背包的最大合法解。因此对于要求恰好装满背包的 0-1 背包问题，初始时应该 $f_0 \leftarrow 0, f_{1\cdots N} \leftarrow -\infty$，即初始时只有容量为 0 的背包有一个合法解 0。如果仅要求最后背包中物品的价值和最大，初始时 $f \leftarrow 0$ 即可。例题为 2022HangZhou-ICPC-C。

1.2 完全背包

> 有 N 种物品和一个容量为 V 的背包。放入第 i 种物品耗费的容量是 C_i，得到的价值是 W_i，且每种物品有无限个。求解将哪些物品装入背包可使价值总和最大。

显然，第 i 种物品放入背包的数量上限是 $\lfloor V/C_i \rfloor$，同时由于每种物品是无限的，如果存在 $i, j \in N$ 使得 $C_i \le C_j, W_i \ge W_j$，将第 j 种物品去掉不会影响答案。

1.3 多重背包

1.4 树形背包

1.4.1 回退背包

Part XI
其他专题

1 均摊分析

2 计算理论

2.1 自动机

2.2 形式语言

2.3 可计算性

2.4 复杂性理论

3 随机化算法

3.1 随机函数

3.2 模拟退火

3.3 Moser 算法

3.4 梅森旋转算法

3.5 蒙特卡洛方法

4 启发式算法

4.1 人类智慧

4.2 启发式搜索

4.3 启发式合并

4.4 启发式分裂

4.5 DSU-on-Tree

5　根号分治

6 数值算法

6.1 区间逼近算法

6.1.1 区间二分

6.1.2 区间三分

6.1.3 WQS 二分

6.1.4 牛顿迭代

6.2 自适应辛普森

6.3 多项式插值

7 连续段问题

详见 IOI-2025.pdf，P1

8 编码问题

9 拟阵问题

10 凸函数问题

详见 IOI-2023.pdf，P232

11 最优化问题

11.1 拉格朗日乘数法

详见 IOI-2024.pdf，P37

11.2 拉格朗日对偶

11.3 线性规划

12 独立集问题

详见 IOI-2017.pdf，P32