

# Das Konstruktionsproblemen Drei

SJTU\_Dreadnought  
TankEngineer

倪昊斌

# The Hardest Question

ftiasch:认真学习了您的《构造题选讲》，上面的  
题目我都会做啦！



TankEngineer:很好啊！




# The Hardest Question

[Some time later...]

ftiasch:萝莉控啊，这个M怎么捉啊..... 

TankEngineer:我感觉随便构造一下就可以了

啊..... 

# The Hardest Question

TankEngineer:@#)\$%^&\* (! (讲解中.....)

悲伤这么大....

ftiasch:为什么窝构造题还是不会做.....



TankEngineer:



# ~~Das Konstruktionsproblemen Drei~~ 构造题选讲改改改

SJTU\_Dreadnought  
TankEngineer  
倪昊斌



# How to AC it

SJTU\_Dreadnought  
TankEngineer

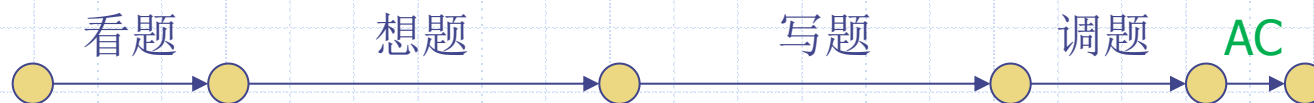
倪昊斌

# A Trivial Question

- 选取15个元素的 $\geq 200$ 个大小为7的子集，使得任意两个被选取的子集交 $\geq 4$ 。
- Solution:  $\binom{6}{5} \times \binom{9}{2} = 216 > 200$

# The Path to AC

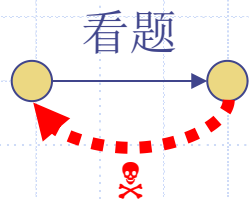
- 理想[大神]:





# The Path to AC

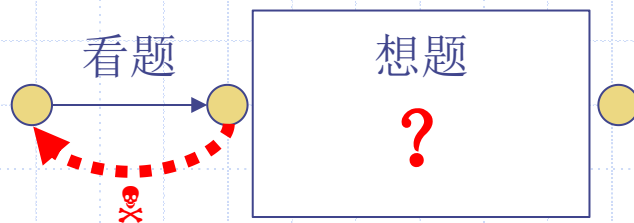
- 现实[菜鸡]:



这种**SB**题我会看错?

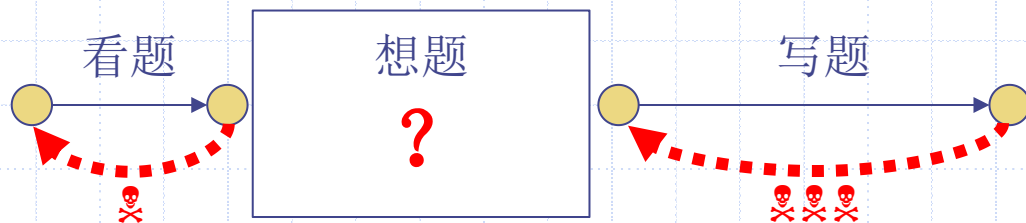
# The Path to AC

- 现实[菜鸡]:



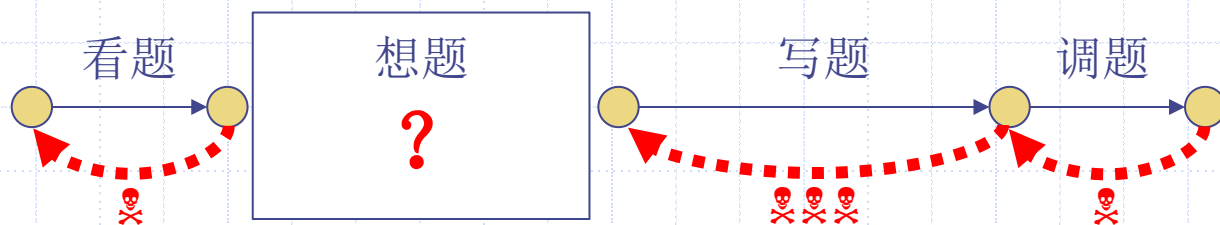
# The Path to AC

- 现实[菜鸡]:



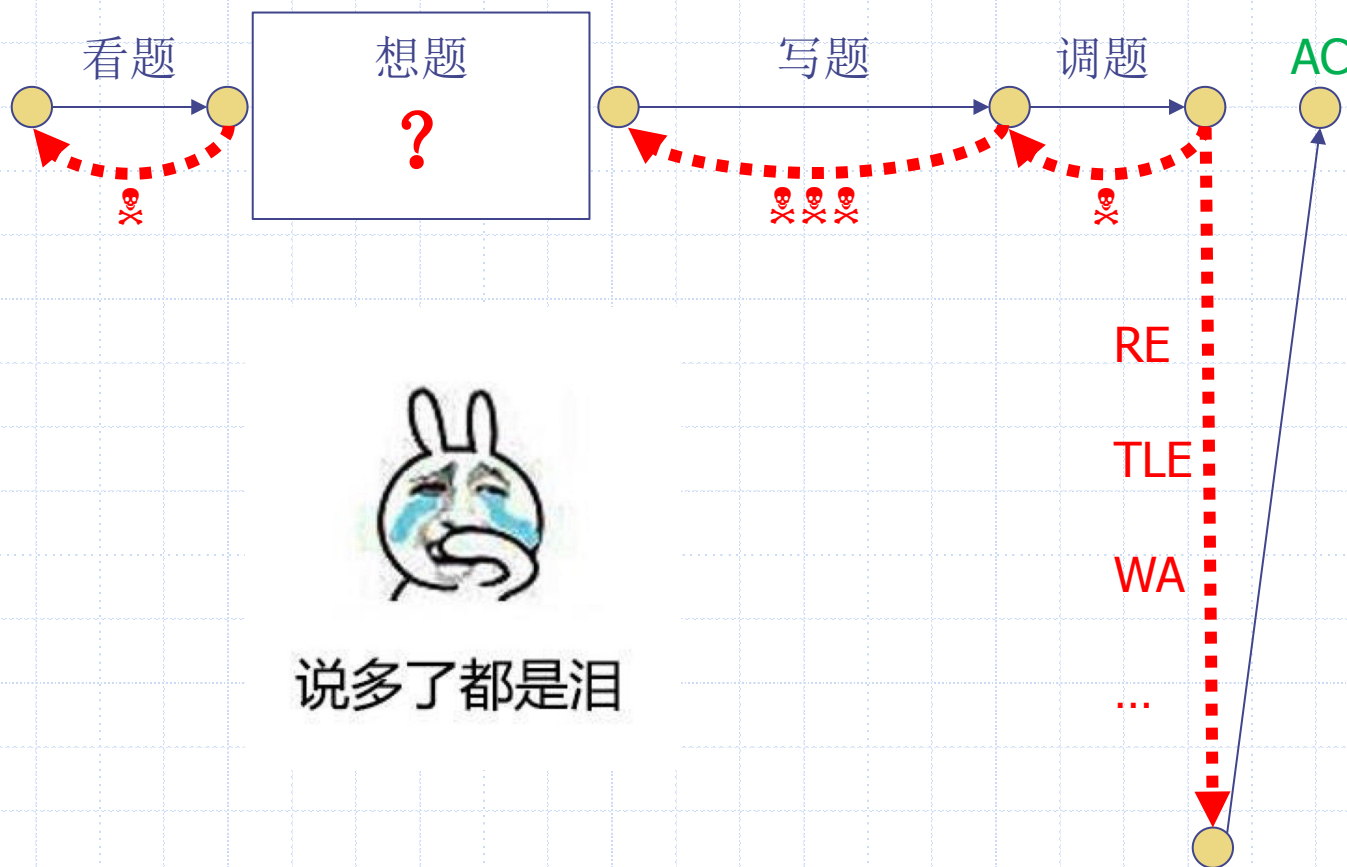
# The Path to AC

- 现实[菜鸡]:



# The Path to AC

- 现实[菜鸡]:



# The Path to AC

- 现实[菜鸡]:

想题

?

# The Path to AC

- 现实[菜鸡]:



# The Path to AC

- Random Guess VS Educated Guess
- 启发式方法(Heuristic Methods)
- 玄学? 干货?



# How to AC it - Overview

## 1. 我们的目标 & Basic Idea

## 2. 行动计划

1. 做题-计划与步骤
2. 积累-知识与方法

## 3. 生动而有趣的例子（讲题）

# The Goal - A Top-down View

- 竞赛水平 → 解题能力 → 想题能力
  - 狠 – 解出难题
  - 快 – 快速得解
  - 准 – 考虑全面
  - 稳 – 发挥稳定

# The Goal – A Bottom-up View

- 一套系统性的方法，能在程序设计竞赛中，以清晰的思路来进行想题的尝试

# The Basic Idea

两个方面的努力：

- 在做题中加入计划与步骤
- 有意识地积累和组织知识和方法

# The Plan - What

- 在开始（想/码）一个题前，制定一个行动计划，明确一下有哪些步骤
- 在上题之前的准备
  - 公式题-推好公式
  - 动规题-列好转移方程
  - 模拟题-写清楚需要实现的各项细节

# The Plan - What

- 在开始想题之前思考一下要采取怎样的步骤
  - 整体思考方向-几何/数据结构/分类讨论.....
  - 猜测结论/推导性质/枚举模型/辅助问题.....
- 确定性方法 VS 启发式方法

# The Plan - Why

1. 是将随机、混乱的思维过程变得清晰、可靠的基础

- The Hardest Question
- 防止思维定式，减少yy错误

1. 能够更加充分地发挥积累

- 题量与水平

# The Plan - Why

## 3. 更有利于自身水平的提高

- 克服偶发错误
- 便于知识的归纳收集



# The Plan - How

## 1. 在开始之前先定计划

- 养成习惯

## 1. 在执行之中灵活调整

## 2. 针对不同的情况进行变通

- 好记性和烂笔头

# The Plan - How

4. 面对困难和失败坚持不懈

5. 不需要砍掉重练，充分利用已有经验和习惯

# The Steps

- 步骤对应了一系列具体的操作流程，是构成计划的基础
- 步骤有不同的层次和不同的类型

# The Steps - In General

- 理解题意
- 想算法
- 得到算法/得不到算法gg
- 上机实现
- 调试
- 提交通过/提交未通过回到之前的步骤
- （偶尔）重新思考改进算法

# The Steps – A Closer Look

- 一些例子:
- 能否引入分块然后暴力?
- 能否将不同的条件/情况分开处理?
- 能否对于不同的对象进行计数再容斥?
- 能否转化为某一经典网络流模型?
- 能否使用Knuth优化?
- 题目的条件可以进行怎样的化简?

# Accumulation of Knowledge

Lev0: XXX是啥

Lev1: XXX大概听说过

Lev2: XXX有没有性质A? 有

Lev3: XXX有哪些性质? A1, A2, A3.....

Lev4: XXX的性质A1可以推出A2.....

Lev5: XXX的一般化YYY, 存在特例ZZZ

.....

# Accumulation of Knowledge

- 只有熟练掌握的知识，才能在竞赛的高强度环境下使用出来
- 而积累知识时就需要为临场时的使用作好准备
- 为此需要对知识进行组织、管理和维护，使其结构性、系统性更强

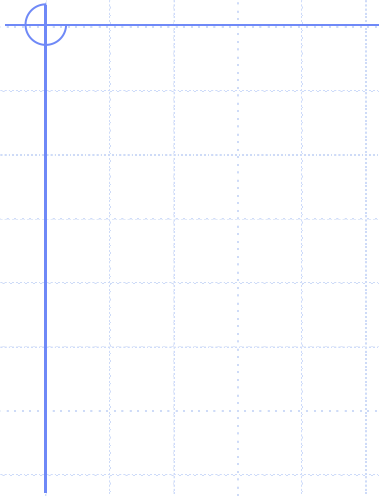
# Accumulation of Knowledge

- 组织的要点是抓住特征
  - 少量而具有明确区分度的信息
- 对熟练掌握、良好管理的知识就能形成解题中的具有可行性的“步骤”
  - DP
  - 网络流



# Heuristic Methods

- 解题策略（启发式方法）的积累
  - 构造
- 使用时机的积累
  - 经验-建立在清晰了解策略的基础上
- 直觉(or潜意识)方法



**EASY**

# [Makoto Soejima Contest 3]

## Tournament

- 求 $n \leq 2000$ 个点的所有竞赛图（点有标号）的强连通分量个数之和。

# Solution

## 1. 竞赛图的强连通分量

1. 竞赛图任意两个元素之间均存在有向边
  2. 缩完强连通分量，非同一连通分量的两个元素一定有祖孙关系
  3. 只能是一条链，转化为算链的长度和
- 如果熟悉这一性质就能立即反应出来

# Solution

## 2. 算链的长度和

1. 链的长度=边的条数+1=有多少种方法可以将图切开，中间的边都同向
  2. 转化为对所有竞赛图总共有多少种方法可以切开
  3. 可以通过枚举一边的大小简单计算
- 枚举一个集合和另一边割开是多见的处理方法

# Comments

- 体现了熟练掌握知识点的重要性
- 利用类似的已有问题和解法解决新问题

# [IX Open Cup GP of Azov Sea]

## HAL 9000

- 长宽 $\leq 200$ 的棋盘上两个人轮流移动两个棋子。移动方式同车，但不能跨越对手的攻击范围。将对手吃掉赢。赢家总想最快赢，输家则希望尽可能拖时间。求谁赢及所经过的步数。

# Solution

## 1. 分析必胜的条件

1. 先看一个**non-trivial**的具体例子

◆ 对角相邻，且一人在角上

1. 推广，倘若不在角上，则可一步步逼入角上

2. 再推广，倘若并非对角相邻但先手已知要输则可采用同样策略

3. 两人的相对位置没有发生变化



# Solution

## 1. 分析必胜的条件 cont'

1. 胜负仅与两人的相对位置有关
2. 想要dp，观察到两人一定越走越近
  1. 如果败者走远，胜者只需追上即可
  2. 如果胜者走远，则走到必败态，而必败态的所有后继状态都是必胜态，则存在走近的必胜方案
3. 得到判断胜负可以dp

# Solution

## 2. 分析时间

1. 注意到最后一定是两个人对角相邻，则贪心的拖时间策略是每次挪一步
2. 而在这之前，胜者一定会缩短相对距离，败者若向后方退却则不如同样缩短距离可以苟活更久
3. 于是时间也可以在dp的同时计算出来（再加上初始位置的影响）

# Comments

- 博弈题常见解法——打牌
- 博弈题分析策略——从简单的non-trivial情况入手，向一般情况扩展寻找规律
- 一般判定比计数要容易分析一些

# [XVI Open Cup GP of Japan]

## Laser Cutter

- 平面上有 $n \leq 300$ 条线段，保证这些线段上任意两点可达，和一个初始在线段上某点的切割机。必须顺着给定方向切割这 $n$ 条线段并回到初始点，求移动的最短距离。

# Solution

## 1. 简化问题

1. 因为形成回路，所以起点没有用
2. 因为整个区域连通，所以如果有多个部分的回路，一定可以形成一个完整的回路
  - ◆ 类似于构造欧拉回路
1. 考虑关键点
  - ◆ 倘若从线段中某个点跑出则一定要再回到这个点重新切，可将这两个回路分离

# Solution

## 1. 简化问题cont'

- 4. 只有线段端点是关键点
- 5. 要形成一些回路，只需考虑每个点的后继
  - ◆ Global问题转化为Local问题
  - ◆ 对于线段起点，后继一定是线段终点（长度固定）
  - ◆ 对于线段终点，则后继一定是某个线段起始点
- 4. 二分图→KM

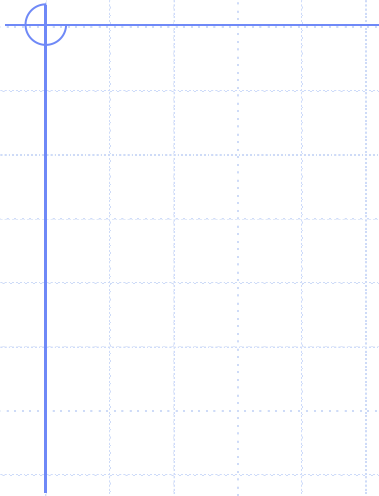
# Comments

- 当题目条件较多且关系（看起来）不紧密时，尝试简化问题
- 关键点、事件点——哪些因素会对答案造成影响
  - 几何/数据结构/动态规划

# Comments

- Global VS Local – 看问题的不同角度，通常Local的问题更容易些
- 对于二分图等模型要十分熟悉，能敏锐察觉





**MEDIUM**

# [XVI Open Cup GP of St. Pb]

## Gardening Lesson

- 给出一棵树(大小 $\leq 10^5$ )和其删掉一个叶子之后的同构，求被删除掉的叶子的编号，有多解输出字典序最小的。

# Solution

## 1. 寻找不变量

1. 树的重心在同构的意义下不会变
2. 添加一个叶子，重心几乎不变
  1. 单重心最多向叶子方向移动一步
  2. 双重心变为单重心
3. 枚举重心（常数），得到一对对应点

# Solution

## 2. 先不考虑字典序求解答案

1. 经典问题：判断两棵树是否自同构

- ◆ 树Hash

1. 对于仅有一个叶子不同则只有可能在儿子的Hash比对有小的不同

- ◆ 多了一个儿子（叶子）

- ◆ 两边各有一个儿子匹配不上

- ◆ 其他情况非法

1. 在树上递归下去找到那个多出的叶子

# Solution

## 3. 再考虑字典序

1. 在Hash比对过程中难以判断从哪个儿子递归下去字典序答案最小
2. 多个答案的情况说明树本身存在自同构，利用类似方法求出树本身的自同构找到字典序最小的解

# Comments

- 在发生复杂变化（如同构、Nim）时，寻找不变量
- 依次加强对答案的限制
- 对于经典做法的积累-快速搜寻相关问题

# [XVI Open Cup GP of Europe]

## Greenhouse Growth

- 有 $n \leq 10^5$ 棵树高度不一排成一排。有两种操作，操作L从左到右check每棵树，如果它是最左或者它比左边的矮就+1，操作R从右到左check如果它是最右或者它比右边的矮就+1。给出一串操作序列 $\leq 10^5$ 求最终每棵树的高度

# Solution

## 1. 寻找不变量

1. 有连续的一段高度相同，操作后一定保持高度相同
2. 基于“段”做处理

## 2. 考虑变化的关键点

1. L或R对一段施加的影响之和这一段及其左/右邻居的高度关系有关



# Solution

## 2. 考虑变化的关键点cont'

2. 当大小关系发生改变的时候，一定是从不等变为等于，发生合并
3. 需要重新计算和邻居的合并事件

# Solution

## 3. 需要维护的对象

1. 段
2. 每段的高度
3. 相邻段大小关系
4. 合并事件

## 4. 其关系为

1. 合并事件→更新哪些段的高度
2. 段的高度→发生合并，更新大小关系
3. 大小关系→产生新的合并事件

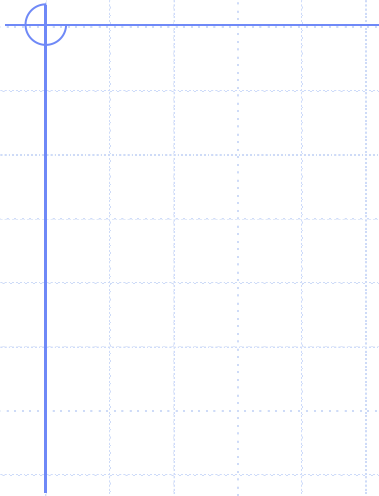
# Comments

- 写题是想题的延伸和具体化，只有想得清楚才能写得清楚
- 对于维护信息的题目，明确维护的对象以及对象之间的作用关系才能更好地写对

# [AIM Fund Cup 2015]

## Decomposition

- 定义两个图G和H，是将H的每个点变成一个G，并且H中有边的两个点变成的G的相应点也连边。给一个图G，点边 $\leq 2 \times 10^5$ ，将其分解为一个尽可能大的Hamming Cube和另一个图的乘积。



**HARD**

# [2015 NEERC Northern]

## Insider's Information

- 有 $n \leq 10^5$ 个元素和 $m \leq 10^5$ 条限制，每条限制是说某三个元素 $a, b, c$ 要么满足 $a < b < c$ 要么满足 $c < b < a$ 。求一个满足 $\geq m/2$ 条限制的排序，保证存在一个排序满足所有限制。

# Solution

## 1. 特殊要求: $\geq m/2$

1. 有些怎样的思路?

1. 常见思路一: 随机/近似算法

2. 常见思路二: 类似均摊分析 挂权重+贪心

## 2. 特殊条件:

1. 要求  $a < b < c$  或  $c < b < a$

◆ 说明  $a/c$  有且仅有一个在  $b$  前边

# Solution

## 2. 特殊条件cont'

2. 保证存在一个序列满足所有要求
  - ◆ 这有什么用？怎么和其他条件建立联系？

## 2. 弱化方法

1. 假设只需保证a和c中有人在b前面
  1. a、c到b连有向边——有环！
  2. 只算为1的度——因为保证有满足所有要求的序列，这样一定可以排出某个拓扑序



# Solution

## 4. 按照这一拓扑序贪心

1. 按逆拓扑序从中间向两边放，每次放在满足条件多的那边
2. 由于**a**和**c**当中一定有一个在**b**后面被贪到，所以一条限制至少给答案贡献了**0.5**，满足题目要求

# Comments

- 在计划中看好思路方向，防止思维定式
- 不常见的特殊条件和特殊要求通常很关键
- 采用弱化/强化条件的方法寻找联系

# [2015 JAG Spring]

## New Game AI

- 有 $n \leq 10^5$ 个人，每个人有防御力 $d_i$ 和血量 $h_i$ ，保证没有完全相同的两个人，要选一个人打。现依次考虑每一个人，如果当前人和已选人的血量差不超过 $c$ ，则留下防御力小的，否则留下血量小的，问有哪些个人最后可能留下。

# Solution

## 1. 先考虑所有 $d_i$ 和 $h_i$ 均不同

### 1. 先考虑特殊Case

- ◆  $h_i$ 最小的那一块中， $d_i$ 最小的人一定可以成为答案[简单构造☺]

1. 一般化，如果存在一条链能够直接/间接战胜他的人也可以赢，则也一定可以成为答案[简单构造☺]
2. 其他人遇到他一定会被T掉，无法成为答案

# Solution

## 2. 再引入有 $d_i/h_i$ 相同的情况

1. 刚才的结论仍然部分成立，但此时由于 $d_i$ 相同，存在谁先上谁就赢的case
  - ◆ 所有人 $h_i$ 仅有微小差距， $d_i$ 均相同
1. 需要打补丁

# Solution

## 3. 如何才能利用先上的优势取得最终的胜利

1. 设刚才求出的答案集合为 $X$ ，显然一旦 $X$ 的某人上位，答案就只能在 $X$ 中，所以 $X$ 中的每一个人都必须找到一个人利用先上优势挤掉
2. 贪心选择这个人，则是 $d_i$ 和他相同且 $h_i$ 最小（一定大于他）的这人。如果这还挤不掉他， $X$ 就是答案。

# Solution

## 3. 如何才能利用先上的优势取得最终的胜利cont'

3. 于是现在对于每个X的人都能想办法挤掉，此时，利用之前的思路，去掉X之后的人当中 $h_i$ 最小的一块中 $d_i$ 最小的人以及能打败他的人就能成为答案[构造证明☺]

# Solution

## 4. 这就完了吗？

1. 未必。在利用前面的思路，如果对于现在扩充的 $X$ 集合，仍然对于每个人都可以挤掉，还是可以进一步扩充。
2. 于是不断迭代直到不能再扩充 $X$ 为止就是答案。
3. 惊奇地发现这个过程可以用**BFS+线段树维护**



# Comments

- 解难题要用更加复杂的计划和更多的步骤。这对于随机猜测地想题是极为困难的。但如果能够将思路结构化，清晰地组织起来便也是可能在赛场上解出的。
- 这不是个构造吗？
- 这不是个构造吗？
- 这不是个构造吗？

# [Potyczki Algorytmiczne 2011]

## Plotter

- 定义串  $L_1=L$ ,  $L_i$  是在  $L_{i-1}$  中间隔插入  $L$  和  $R$ ,  $L_2=\underline{L}L\underline{R}$ ,  $L_3=\underline{L}L\underline{R}L\underline{L}R\underline{R}$ ..... $L_i$  对应了一条平面上的折线: 从  $(0,0)$  出发先走到  $(1,1)$ , 遇到  $L$  左转  $90^\circ$  走一步, 遇到  $R$  右转  $90^\circ$  走一步。问在  $L_n$  的什么时候到达了某个点  $(x_i, y_i)$
- $n \leq 2000$  询问个数  $\leq 2000$   
 $|x_i|, |y_i| \leq 10^9$

# Summary

## 1. 建立工作框架

- 有计划、有组织地想题
- 通过可操作的步骤运用积累

## 1. 进行充分积累

- 围绕知识点进行组织，形成知识体系
- 运用启发式的方法，用经验判断时机

# Acknowledgement

- 感谢ICPCCamp2016提供的平台
- 感谢ftiasch的指导和反馈
- 感谢诸位的听讲

# Further Readings

---

- G Polya. 1957. How to Solve It 2<sup>nd</sup>Ed
- G Polya. 1954. Mathematics and Plausible Reasoning
- Coursera: Learning How to Learn