校赛命题指南

刘珊

ACM 编程协会

2021 年 8 月 1 日

明确定位

- 如果你有一个优质的 idea，并且在网上搜索过后找不到相关解法，在确认做法可行难度阻间后，可以放心的将它放到你的题库中。

明确定位

- 如果你有一个优质的 idea，并且在网上搜索过后找不到相关解法，在确认做法可行难度阳间后，可以放心的将它放到你的题库中。

- 如果没有，在确定考察方向后，可以考虑利用现有的题库，来组一套校赛的试题。

明确定位

- 如果你有一个优质的 idea，并且在网上搜索过后找不到相关解法，在确认做法可行难度阳间后，可以放心的将它放到你的题库中。
- 如果没有，在确定考察方向后，可以考虑利用现有的题库，来组一套校赛的试题。
- 我们重点讲述第二种情况。

明确定位

- 首先，我们出的是弱校校赛题，而不是区域赛等大型比赛的题目。

明确定位

- 首先，我们出的是弱校校赛题，而不是区域赛等大型比赛的题目。
- 因此可以选择一些较简单的，或者网上已经出现过的题目来作为赛题。

明确定位

- 首先，我们出的是弱校校赛题，而不是区域赛等大型比赛的题目。
- 因此可以选择一些较简单的，或者网上已经出现过的题目来作为赛题。
- 但需要注意，这并不意味着我们可以使用经典题，模板题，或是区域赛/天梯赛/中出现的真题。

不应出现的题目

- [noip2005 普及组] 采药（01 背包模板题）

不应出现的题目

- [noip2005 普及组] 采药（01 背包模板题）
- 平面最近点对（分治算法经典题）

不应出现的题目

- [noip2005 普及组] 采药（01 背包模板题）
- 平面最近点对（分治算法经典题）
- [蓝桥杯 2021 国赛] 异或变换（大型比赛真题）

选择已有的题目

- 参考标准：选手不太容易写到原题；题目质量要有保障；最好能找到现成的题解和测试数据。

选择已有的题目

- 参考标准：选手不太容易写到原题；题目质量要有保障；最好能找到现成的题解和测试数据。
- CodeForces/AtCoder 这类国外热门的 OnlineJudge

选择已有的题目

- 参考标准：选手不太容易写到原题；题目质量要有保障；最好能找到现成的题解和测试数据。
- CodeForces/AtCoder 这类国外热门的 OnlineJudge
- Project Euler 一些利用计算机编程解决的数学问题合集

选择已有的题目

- 参考标准：选手不太容易写到原题；题目质量要有保障；最好能找到现成的题解和测试数据。
- CodeForces/AtCoder 这类国外热门的 OnlineJudge
- Project Euler 一些利用计算机编程解决的数学问题合集
- LOJ 等选题有保障，且能下载数据的国内 OnlineJudge

CodeForces

- CodeForces 能够提供一些小数据的下载。

CodeForces

- CodeForces 能够提供一些小数据的下载。
- 太大的数据会被省略而不显示。

CodeForces

- CodeForces 能够提供一些小数据的下载。
- 太大的数据会被省略而不显示。
- 可以下载构造的小数据，较大的数据用生成器造。

CodeForces

- CodeForces 能够提供一些小数据的下载。
- 太大的数据会被省略而不显示。
- 可以下载构造的小数据，较大的数据用生成器造。
- 顺便可以学习构造的数据与题目的关联，以后如果有自己的 idea 想要出强数据这也会是一份不错的经验。

CodeForces

- 提交后进入 submission，往下拉就能看到小数据。

Time: 15 ms, **memory:** 3796 KB
Verdict: OK
Input
```
1
4
4 1 2 3
```
Participant's output
```
3
```
Jury's answer
```
3
```
Checker comment
```
ok 1 number(s): "3"
```

9

Time: 30 ms, **memory:** 3800 KB
Verdict: OK
Input
```
1
3
3 1 2
```
Participant's output
```
1
```
Jury's answer
```
1
```
Checker comment
```
ok 1 number(s): "1"
```

10

Time: 0 ms, **memory:** 3812 KB
Verdict: OK
Input
```
1
6
3 3 3 1 3 3
```
Participant's output
```
1
```
Jury's answer
```
1
```

AtCoder

- AtCoder 有一个公开网盘用来存比赛的测试数据。

AtCoder

- AtCoder 有一个公开网盘用来存比赛的测试数据。
- 官方公告：https://atcoder.jp/posts/20

AtCoder

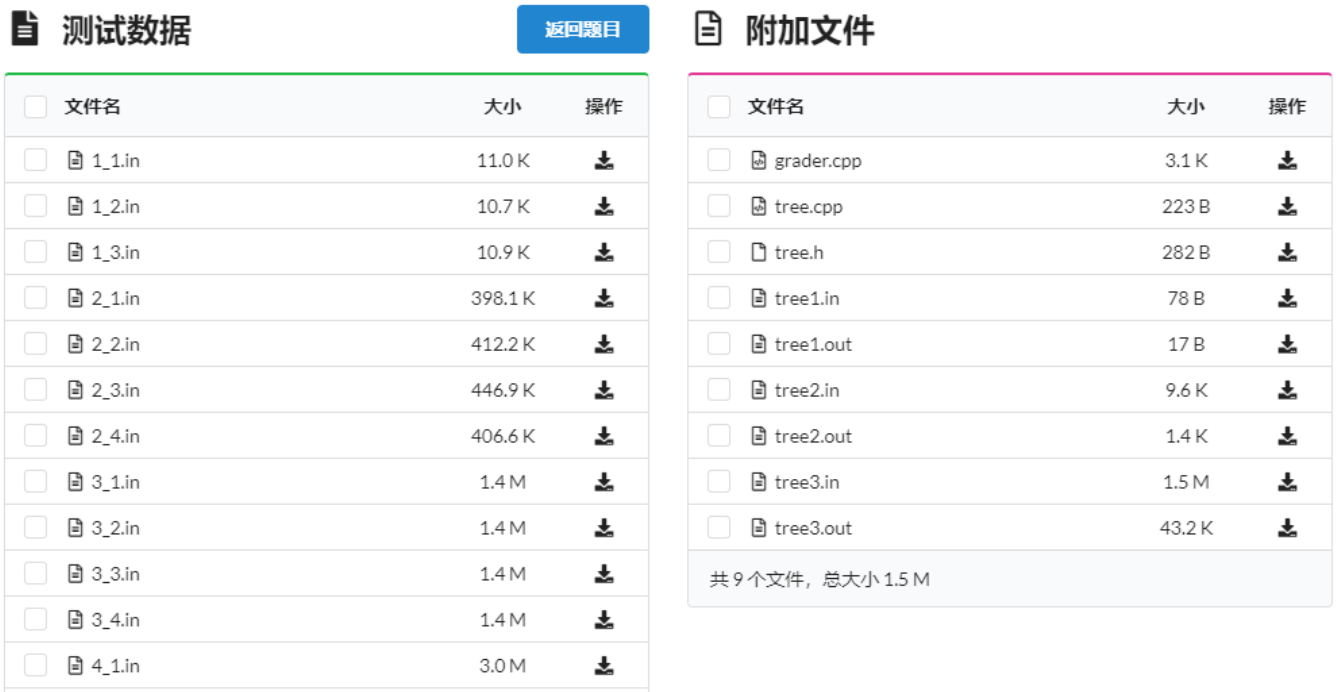
- AtCoder 有一个公开网盘用来存比赛的测试数据。
- 官方公告：https://atcoder.jp/posts/20
- 网盘地址：https://www.dropbox.com/sh/nx3tnilzqz7df8a/AAAYlTq2tiEHl5hsESw6-yfLa?dl=0

AtCoder

- AtCoder 有一个公开网盘用来存比赛的测试数据。
- 官方公告：https://atcoder.jp/posts/20
- 网盘地址：https://www.dropbox.com/sh/nx3tnilzqz7df8a/AAAYlTq2tiEHl5hsESw6-yfLa?dl=0
- 需要能够访问 Dropbox 才能下载。

LOJ

- LOJ 通常指的是 LibreOJ。
- 在题目界面，有个叫[文件]的入口，点进去就能下载测试数据和其他一些下发数据，SPJ 等。



测试数据 返回题目

文件名	大小	操作
1_1.in	11.0 K	
1_2.in	10.7 K	
1_3.in	10.9 K	
2_1.in	398.1 K	
2_2.in	412.2 K	
2_3.in	446.9 K	
2_4.in	406.6 K	
3_1.in	1.4 M	
3_2.in	1.4 M	
3_3.in	1.4 M	
3_4.in	1.4 M	
4_1.in	3.0 M	

附加文件

文件名	大小	操作
grader.cpp	3.1 K	
tree.cpp	223 B	
tree.h	282 B	
tree1.in	78 B	
tree1.out	17 B	
tree2.in	9.6 K	
tree2.out	1.4 K	
tree3.in	1.5 M	
tree3.out	43.2 K	

共 9 个文件，总大小 1.5 M

Cyaron

- 对于大部分校赛级别的题目，Luogu 的 Cyaron 是个不错的选择。

Cyaron

- 对于大部分校赛级别的题目，Luogu 的 Cyaron 是个不错的选择。
- 项目地址：https://github.com/luogu-dev/cyaron

Cyaron

- 对于大部分校赛级别的题目，Luogu 的 Cyaron 是个不错的选择。
- 项目地址：`https://github.com/luogu-dev/cyaron`
- 优势：实现了常见的数据生成器的需求（比如生成图生成树生成随机序列），并有较为详细的官方文档，在 Python 环境下不需要什么额外配置就能运行。

Cyaron

- 官方文档：https://github.com/luogu-dev/cyaron/wiki

图 Graph

Graph库可以用来生成各种各样的树、图、链等结构。

这个库中本身已经配备了各种常用的图的模板，不过我们先来看看手动建立一个图的方法。

```
graph = Graph(10) # 建立一个10个节点的无向图
graph = Graph(10, directed=True) # 建立一个10个节点的有向图
# 这两个图的节点编号范围都为1到10

graph.add_edge(1, 5) # 建立一条从1到5，权值为1的边，若是无向图，还会建立从5到1的边
graph.add_edge(1, 6, weight=3) # 建立一条从1到6，权值为3的边，若是无向图，还会建立从6到1的边

graph.edges # 一个邻接表数组，每一维度i保存的是i点出发的所有边，以Edge对象存储
for edge in graph.iterate_edges(): # 遍历所有边，其中edge内保存的也是Edge对象
    edge.start # 获取这条边的起点
    edge.end # 获取这条边的终点
    edge.weight # 获取这条边的边权
    io.input_writeln(edge) # 输出这条边，以u v w的形式

io.input_writeln(graph) # 输出这个图，以每条边u v w一行的格式
io.input_writeln(graph.to_str(shuffle=True)) # 打乱边的顺序并输出这个图
io.input_writeln(graph.to_str(output=my_func)) # 使用my_func函数替代默认的输出函数，请查看源代
io.input_writeln(graph.to_str(output=Edge.unweighted_edge)) # 输出无权图，以每条边u v一行的格
```

不过在大多数情况下您不需要手动建图，我们为您准备了大量模板，用法如下：

Pages 17

首页
Python 快速入门
文档
基本入门
输入输出 IO
图 Graph
多边形 Polygon
向量 Vector
字符串 String
序列 Sequence
对拍器 Compare
工具函数
常用常数
数学函数
高级使用

Cyaron

- 代码片段展示

```python
#!/usr/bin/env python

from cyaron import *

#1-3的数据通过CodeForces复制粘贴下来了，因此这里是range(4,11)，生成较大的数据。
for i in range(4, 11):
    test_data = IO(file_prefix="", data_id=i)
    #测试数据文件名格式，这里prefix为空，输出就为1.in/1.out这种样子

    if i <= 6:
        t = randint(1, 1000)
        n = randint(1, 300)
        test_data.input_writeln(t)
        for k in range(1, t + 1):
            n = randint(1, 300)
            tmp = []
            for j in range(1, n + 1):
                tmp.append(randint(1, 100000000))
            test_data.input_writeln(n)
            test_data.input_writeln(tmp)
    elif i <= 9:
        t = 15000
        n = 20
        test_data.input_writeln(t)
```

构造测试数据

- 测试数据的强度是每一个出题人都需要考虑的事情。

构造测试数据

- 测试数据的强度是每一个出题人都需要考虑的事情。
- 工具再智能，也不能自动构造数据。

构造测试数据

- 测试数据的强度是每一个出题人都需要考虑的事情。
- 工具再智能，也不能自动构造数据。
- 在 CodeForces 上，你可以在通过一道题的 pretest 后锁定自己的提交，然后去构造数据卡其他人的程序，这个行为称为 HACK。

构造测试数据

- 测试数据的强度是每一个出题人都需要考虑的事情。
- 工具再智能，也不能自动构造数据。
- 在 CodeForces 上，你可以在通过一道题的 pretest 后锁定自己的提交，然后去构造数据卡其他人的程序，这个行为称为 HACK。
- 如果你使用 CodeForces 的 Polygon 来出题，会提示你没有出边界数据的情况。但由于这个工具比较复杂，因此略过不表。

构造测试数据

- 实例：[NC9982-J] 牛牛想要成为 hacker

在算法竞赛中"hack"一般指用一组测试数据触发程序的缺陷，从而导致本来通过题目的AC代码无法通过该测试数据。
一般情况见得比较多的是用hack数据导致别人WA掉，当然也有一些会导致原本的AC代码TLE和MLE。

牛牛在一些简单的练习题时遇到了这样一个问题。
给定一个大小为n的数组a$(1 \leq a_i \leq 10^9)$，然后请你判断数组元素是否能够从中选出三个组成一个三角形。

牛牛发现AC通过的代码中有这样一种暴力逻辑，该逻辑的伪代码如下。

```
FOR i = 1 ... n
    FOR j = i + 1 ... n
        FOR k = j + 1 ... n
            IF isTriangle(a[i],a[j],a[k])
                print("yes")
                EXIT
            END IF
        END FOR
    END FOR
END FOR
print("no")
EXIT
```

其实就是三重循环枚举数组的三个元素，检查是否为三角形。这段代码很取巧的地方在于它存在一种"短路"逻辑，一旦发现存在三角形就立刻终止程序。
这样在随机数据下其实很容易发现三角形，所以如果数据纯随机，显然这就是一段AC代码。

牛牛当然知道这个代码很明显就存在缺陷，如果数据构造的好的话应该可以卡TLE，但是牛牛发现，他并不会构造出能够hack这个暴力算法的数据，所以他请你来帮他。

我们以这段程序调用isTriangle的次数作为时间复杂度的计算依据，请你构造数据hack这段暴力程序，使它TLE掉。

构造测试数据

- 核心代码，我们需要让 isTriangle 调用的次数足够多。

```
1  FOR i = 1 ... n
2      FOR j = i + 1 ... n
3          FOR k = j + 1 ... n
4              IF isTriangle(a[i],a[j],a[k])
5                  print("yes")
6                  EXIT
7              END IF
8          END FOR
9      END FOR
10 END FOR
11 print("no")
12 EXIT
```

构造测试数据

- 这里引用一下Kur1su 的究极小迷弟的题解:

Solution

part.1 fibnoacci数列

这里应该很多人都能想到是利用fibnoacci数列进行构造,但是不知道怎么处理.

我们可以联想到fibnoacci数的性质$fib_i = fib_{i-1} + fib_{i-2}$.

这样我们在构造fibnoacci数列的时候,使该数列形成一个非递增的数列,因为$i < j < k$,这样对于$a_i \neq 1$ 的情况,都可以保证得到$a_i \geq a_j + a_k$,这样就可以确保后面两个循环跑满.

而符合数据范围的fibnoacci数有42个,这样构造使得运行次数大约为$n \times n \times 42$,而$\log_2 1e5 \approx 17$,可知满足题意.

part.2 等比数列

以2为公比的等比数列也是满足题意的.

与fibnoacci数列类似,以2为公比的等比数列满足$a_i > a_{i+1} + a_{i+2}$

所以思路也就变得跟上面一样了,同样满足$i < j < k$ 并且$a_i! = 1$ 时,都能保证$a_i \geq a_j + a_k$.

符合数据范围的数有$\log_2 1e9 \approx 30$ 个,这种构造的运行次数大约为$n \times n \times 30$,同样满足$\log_2 1e5 \approx 17$ 的需求.

构造测试数据

- 还有许多关于数据强度的技巧。

构造测试数据

- 还有许多关于数据强度的技巧。
- 最常见的大概是考虑运算中是否会出现 long long。

构造测试数据

- 还有许多关于数据强度的技巧。
- 最常见的大概是考虑运算中是否会出现 long long。
- 以及如果需要构造一棵树，则大部分时候都会出现退化成链，菊花，等的测试数据。

构造测试数据

- 还有许多关于数据强度的技巧。
- 最常见的大概是考虑运算中是否会出现 long long。
- 以及如果需要构造一棵树，则大部分时候都会出现退化成链，菊花，等的测试数据。
- 还有一些比较经典的，比如，关于 SPFA，它死了，具体看：如何看待 SPFA 算法已死这种说法？的题解：

- 一般来说，最容易上手的是使用 markdown 进行排版，许多 OJ 提供的在线排版功能也都提供了 markdown 支持。

- 一般来说，最容易上手的是使用 markdown 进行排版，许多 OJ 提供的在线排版功能也都提供了 markdown 支持。

- 在一些大型比赛中，可能更多使用 LaTex 进行排版，

- 一般来说，最容易上手的是使用 markdown 进行排版，许多 OJ 提供的在线排版功能也都提供了 markdown 支持。
- 在一些大型比赛中，可能更多使用 LaTex 进行排版，
- 前人提供了一个开源项目tuack，可以帮助我们从 markdown 快速排版出效果不错的 LaTex 风格题面。

tuack

- tuack 需要本地有 Tex 环境，具体要求可以看文档。

tuack

- tuack 需要本地有 Tex 环境，具体要求可以看文档。
- https://git.thusaac.com/publish/tuack/wikis/home。

tuack

- tuack 需要本地有 Tex 环境，具体要求可以看文档。
- https://git.thusaac.com/publish/tuack/wikis/home。
- PS：文档目前好像挂了（20210714），还没修好，所以目前不能展开多少内容了。

tuack

- tuack 需要本地有 Tex 环境，具体要求可以看文档。
- https://git.thusaac.com/publish/tuack/wikis/home。
- PS：文档目前好像挂了（20210714），还没修好，所以目前不能展开多少内容了。
- PS2：使用此工具可以参考 THUPC 的结构，题目工程在 Github 上均已开源。

tuack

- 一个简单的示例

conf.yaml

```yaml
compile:
  c: -O2 -std=c11
  cpp: -O2 -std=c++11
  java: ''
  pas: ''
  py: ''
data:
- cases:
  - 1
  score: 100
folder: problem
memory limit: 512 MiB
name: candy
partial score: true
pre: []
samples: []
time limit: 1.0
title:
  zh-cn: 分糖果
type: program
users: {}
version: 2
# 单个题目的yaml文件
```

conf.yaml

```yaml
end time: 2021-03-01 17:00:00+0800
folder: day
name: day1
start time: 2021-03-01 13:00:00+0800
subdir:
- power
- matrix
- stairs
- number
- transform
- color
- string
- candy
- count
- dove
title:
  zh-cn: XX学校 校内模拟赛
version: 2

#比赛日的yaml文件
```

zh-cn.md

```
{{ self.title() }}

{{ s('description') }}

% 题目描述 %

{{ s('input format') }}

% 输入格式 %

{{ s('output format') }}

% 输出格式 %

{{ self.sample_text() }}
% 读取down文件夹内的数据 %

{{ s('hint') }}

% 别的 %
```

tuack

- 编译后效果（原题为小奇取石子）

H. 分糖果 / Candy

时间限制： 1.0 秒
空间限制： 512 MiB

【题目描述】

小 A 和小 E 是好朋友，今天是儿童节，她们今天要分糖果给小学的同学们吃。

但是小 A 不想那么容易就把糖果分出去，于是出了个问题考考其他的小朋友们：

现在有 n 堆糖果，每堆糖果有 a_i 颗。你们有最多 m 次的选择机会，每次可以选择一堆糖果拿走。在拿走的糖果总量不超过 k 的情况下，能够拿走的糖果最多是多少？

聪明的小朋友们很快利用手里的计算机编程解决了问题。

在拿到糖果后他们很兴奋，于是想拿这道题考考你，请你告诉小朋友们能拿走的糖果最多是多少？

【输入格式】

第一行三个正整数 n，m，k，意义见题目描述。

第二行一共 n 个正整数，代表每堆糖果的数目。

【输出格式】

输出包括一行一个正整数，代表小朋友们能拿走的糖果最大数量。

【样例输入】

- 出题不是出题人与做题人之间的较量。

- 出题不是出题人与做题人之间的较量。
- 因此，我个人认为，特别是在难度不高的校赛中，在题解中体现自己对于这道题的看法是不错的。

- 如欧拉计划 63，可以用于考察的 ___int128 和数学能力。
- 当时编写的题解：

本题更多是针对 C/C++ 选手。
第一题其实是前三题里面最难的，把这一题放在前面是为了提示，有的时候后面的题会更简单。
该题一是对标 2017A 第三题魔方。如果有群论基础，很容易可以完成 2017A 的第三题，否则要写较为复杂的代码判断“本质相同”的问题。在当下这一题，如果会数学解法，很少代码就可以完成，否则就可能需要编写复杂的高精度代码。

该题二是对标 2019*A* 第五题 *RSA* 解密，那题使用快速幂求解时会超过 *long long* 的表示范围，同样是需要编写高精度代码的。那么有没有不用写高精度的方法？可以尝试学习 ___*int*128 类型，在数字相对较小但又超过 *long long* 表示范围时非常有用。

Solution1

分析：设题目要求的幂的底为 n，指数为 k，则这个幂应为 k 位数，则有：

$10^{k-1} < n^k < 10^k \Rightarrow k-1 < k \cdot log_{10} n < k$

因为 $k \geq 1$，则对于不等式 $k \cdot log_{10} n < k$，有

$log_{10} n < 1 \Rightarrow n < 10$。对于另一边有：

$k-1 < k \cdot log_{10} n \Rightarrow k < \frac{1}{1-log_{10} n}$

则我们有 $1 \leq n < 10, 1 \leq k < \frac{1}{(1-log_{10} n)}$。

因此我们只需要遍历所有符合条件的 n，统计在特定的 n 时有多少个符合条件的 k 并加总，即为题目所求。

Solution2

很好证明，底数一定是 $1-9$。
我们假设底数 $w \geq 10$，则 f^n 可以变为 $w' * 10^{n+1}$，
$1 \leq w' < 10$，得到一个 $n+1$ 位数，与题意不符。
因此，底数一定是 $1-9$。
然后考虑猜测它的一个上界，我们上机编程时，是可以使用计算器的。考虑边界情况 $9^n = \lfloor log_{10} n \rfloor + 1$，用计算器可以测试出这个边界在 $n = 25$ 左右，因此可以使用 `__int128` 辅助我们进行暴力枚举。

- 当然，某些题目你还可以给出部分分的解法，指导选手在天梯赛等比赛中如何骗分。

- 当然，某些题目你还可以给出部分分的解法，指导选手在天梯赛等比赛中如何骗分。

- 如传球游戏，这一经典问题有爆搜，记忆化搜索，递推，矩阵快速幂等多种做法。

- 当然，某些题目你还可以给出部分分的解法，指导选手在天梯赛等比赛中如何骗分。
- 如传球游戏，这一经典问题有爆搜，记忆化搜索，递推，矩阵快速幂等多种做法。
- 你可以利用lemon-lime等工具对你的部分分代码进行评测，并以此为依据编写题解。

- Lemon 对部分分的评测结果导出：

排名表

排名	名称	strong	总分
1	100pts-矩阵	100	100
2	60pts-记搜	60	60
2	60pts-递推	60	60
4	10pts-爆搜	10	10

- 主要是介绍一些能生成更好插图的网站，帮助更好的完善题解 PPT 或者是题面的样例说明。

- 主要是介绍一些能生成更好插图的网站，帮助更好的完善题解 PPT 或者是题面的样例说明。

- 函数绘图：https://www.geogebra.org/

- 主要是介绍一些能生成更好插图的网站，帮助更好的完善题解 PPT 或者是题面的样例说明。

- 函数绘图：https://www.geogebra.org/

- 简单有向图无向图绘图：
  https://csacademy.com/app/graph_editor/

- 主要是介绍一些能生成更好插图的网站，帮助更好的完善题解 PPT 或者是题面的样例说明。
- 函数绘图：https://www.geogebra.org/
- 简单有向图无向图绘图：https://csacademy.com/app/graph_editor/
- 算法与数据结构可视化：https://visualgo.net/zh

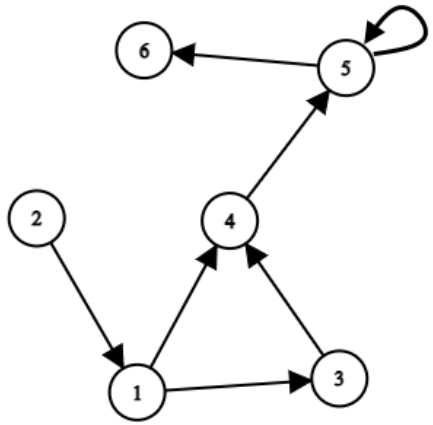
- 主要是介绍一些能生成更好插图的网站，帮助更好的完善题解 PPT 或者是题面的样例说明。
- 函数绘图：https://www.geogebra.org/
- 简单有向图无向图绘图：https://csacademy.com/app/graph_editor/
- 算法与数据结构可视化：https://visualgo.net/zh
- 搜索可视化：https://qiao.github.io/PathFinding.js/visual/

- 这是 CodeForces 的某次比赛的题目，该题使用的插图来自于上文介绍过的简单有向图无向图绘图网站。

For each vertex v output one of four values:

- 0, if there are no paths from 1 to v;
- 1, if there is only one path from 1 to v;
- 2, if there is more than one path from 1 to v and the number of paths is finite;
- -1, if the number of paths from 1 to v is infinite.

Let's look at the example shown in the figure.



- OI-WIKI 出题指南
- 「这一点都不好玩」：一份 ACM/ICPC 区域赛试题是怎么诞生的
- 【杂谈】一次在 CF 出题的经历，CF 出题的流程是怎样的？

Thanks!