简谈贪心

by 冯青同

例1：乘船问题

- 有 n 个人，第 i 个人重量为 wi，每艘船的最大载重量均为 C，且最多只能乘两个人。用最少的船装载所有人
- 考虑把所有人按照重量由小到大排序，然后先考察最轻的人：
- （1）最轻的人不能和任何人共乘，那么一定是每个人各自乘一条船
- （2）最轻的人可以和某些人共乘，那么让最轻的人和可以共乘的人中最重的那一个乘一条船
- （3）接下来，相当于一个规模为 n - 2 的子问题，设原问题的答案为 ans，子问题的答案为 res，那么 ans = res + 1，显然，只需要子问题的答案 res 最小，原问题的答案也是最小

例2：P1803 线段覆盖问题

- 大意：在区间上存在 n 条线段 [ai,bi]，请你选出尽可能多的线段，要求线段两两不能相交
- 数据范围：n <= 1e6
- 一个比较自然的思路：在所有能选的线段里，选择右端点最小的那一条。
- 这个思路为什么是对的呢？
- 因为你选右端点最小的那一条，在接下来的的选择中，可选的线段更多，答案一定不会更小
- 但是实现上有一些问题
- 你当然可以搞些二维的数据结构，比如主席树（）
- 但是另一个可行的思路是：把所有线段按照右端点排序，然后从左到右扫一遍，碰到能选的线段就选上。

例3：P1020 导弹拦截

题目描述

复制Markdown 展开

某国为了防御敌国的导弹袭击，发展出一种导弹拦截系统。但是这种导弹拦截系统有一个缺陷：虽然它的第一发炮弹能够到达任意的高度，但是以后每一发炮弹都不能高于前一发的高度。某天，雷达捕捉到敌国的导弹来袭。由于该系统还在试用阶段，所以只有一套系统，因此有可能不能拦截所有的导弹。

输入导弹依次飞来的高度，计算这套系统最多能拦截多少导弹，如果要拦截所有导弹最少要配备多少套这种导弹拦截系统。

例3：P1020 导弹拦截

- 贪心策略：假设当前有 x 个导弹系统，记第 i 个导弹系统目前拦截的最低的高度为 hi，那么每次读入一颗新的导弹时，设其高度为 H，把它接到满足 hi > H 且 hi 最小的那个系统的后面即可。
- 需要的操作：
- （1）查后继
- （2）删除一个数
- （3）添加一个数
- ~~然后……平衡树！（当然不是~~
- 容易发现在加入新的导弹高度以后，并不会影响先前每一个系统的相对次序（想一想，为什么？）

例4：P1080 国王游戏

题目描述

复制Markdown 展开

恰逢 H 国国庆，国王邀请 n 位大臣来玩一个有奖游戏。首先，他让每个大臣在左、右手上面分别写下一个整数，国王自己也在左、右手上各写一个整数。然后，让这 n 位大臣排成一排，国王站在队伍的最前面。排好队后，所有的大臣都会获得国王奖赏的若干金币，每位大臣获得的金币数分别是：排在该大臣前面的所有人的左手上的数的乘积除以他自己右手上的数，然后向下取整得到的结果。

国王不希望某一个大臣获得特别多的奖赏，所以他想请你帮他重新安排一下队伍的顺序，使得获得奖赏最多的大臣，所获奖赏尽可能的少。注意，国王的位置始终在队伍的最前面。

例4：P1080 国王游戏

- 这是一道恰当地安排顺序，使得某一特定值最小的贪心题。对于这种安排顺序的题，我们有一种通解是：邻项交换法。
- 首先来推导式子，见我手写！
- ……（这里是手写部分）
- 总之，我们发现，对于相邻的两个数而言，满足 ai * bi 更小的那个应该扔到前面。
- 下面来证明这样做的正确性，其实采用冒泡排序的思想即可。
- 每一次扫一遍数组，如果有相邻两个元素逆序，那么交换这两个元素，答案不会更劣。
- 你再仔细一看：欸，这不就是冒泡排序吗？所以最终 ai * bi 一定是有序的。
- 证毕。

例5：P2123皇后游戏

- 给定 n 组数据 ai、bi，ci 采用如下方法定义：
- 问：这 n 组数据可以自由排序，若想使最大的 ci 最小，应该怎样安排顺序？
- （1）推导表达式
- （2）hack 数据
- （3）邻项交换法中，不等号的传递性
- 题解通道：https://www.luogu.com.cn/problem/solution/P2123

$$c_i = \begin{cases} a_1 + b_1 & ,i = 1 \\ \max\left\{c_{i-1}, \sum_{j=1}^{i} a_j\right\} + b_i & ,2 \le i \le n \end{cases}$$

例6：CF1748C Zero-Sum Prefixes

题意翻译

给定一个长为 n 的数列 a，你可以将其中的每个 0 分别改成任意整数。求出你最多能让多少个 k 满足 $a_1 + \cdots + a_k = 0$。

输入格式

Each test contains multiple test cases. The first line contains a single integer t ($1 \le t \le 10^4$) — the number of test cases.

The first line of each test case contains one integer n ($1 \le n \le 2 \cdot 10^5$) — the length of the array a .

The second line of each test case contains n integers $a_1, a_2, \ldots, a_n$ ($-10^9 \le a_i \le 10^9$) — array a .

It is guaranteed that the sum of n over all test cases does not exceed $2 \cdot 10^5$.

例6：CF1748C Zero-Sum Prefixes

- 考虑求一个前缀和数组，只需要使前缀和数组中的 0 尽可能多即可。修改 0 为任意数的操作可以理解为给前缀和数组中的某一个位置加上一个任意数。
- 接下来，我们发现这样一个事实：假设两个相邻的 0 的下标分别为 i 和 j，那么修改 i 这个位置的数为 0 只会影响到 [i,j) 这段区间内的数。
- 为什么呢？
- 假设你将 i 位置的数修改为 x，相当于在前缀和的 i 位置加了 x，你完全可以把 j 位置修改为 -x+y，这样抵消掉了 i 位置的修改，并且可以给后面的数都加上任意值 y。

例7：CF853A Planning

题意翻译

Helen在大都会机场工作，她的任务是安排每天的航班起飞时刻。今天一共有n架飞机将要起飞，第i架飞机将在第i分钟起飞。

大都会机场是大都会最重要的交通枢纽，因此想要原封不动地按照起飞时刻表的时刻起飞是很困难的。今天的情况也是如此：由于技术原因，在今天一开始的k分钟内飞机不允许起飞，因此必须创建一个新的起飞时刻表。

所有的航班必须在第(k+1)分钟到第(k+n)分钟内(包括两端)起飞，而且每分钟仅能有一架飞机起飞。然而，航班起飞的先后顺序可以与最初的时刻表排好的顺序不同，重排的时刻表只有一个限制：飞机不能比它在初始时刻表中起飞的时刻还要早的时刻起飞(即：第i架飞机必须在第i分钟后或第i分钟时起飞)。

Helen知道第i架飞机起飞时刻每延误一分钟机场所需支付的额外花费ci是多少。帮助她找到额外花费最小的方案。

例7：CF853A Planning

- 我们不妨把问题转化成这样：从小到大枚举时间，前 n 分钟每一分钟会加入一架可以飞的飞机，后 n 分钟每分钟可以飞走一架可以飞的飞机，那么每一次时间向后推一，都要使当前所有没有飞的飞机产生一次额外花费。
- 容易想到这样的贪心思路：在第 i 分钟时，选择当前可以飞的飞机中，额外花费最大的那一架起飞，最终的代价会是最小的。
- 考虑使用优先队列来进行维护。枚举时间从第 1 分钟到第 n+k 分钟，每次加入一架能飞的飞机，然后弹出额外花费最大的能飞的飞机。注意每一次转移时间时要加上代价。

例8：CF913D Too Easy Problem

题目描述

你正在准备一场关于调度理论的考试。

这场考试会持续正好 T 毫秒，由 n 道题目组成。

你可以用 t_i 毫秒解决第 i 个问题，或者忽略它并不消耗时间。你也不需要用来在做完一道题之后休息的时间。

不幸的是，你的老师认为一些题目对你来说太简单了。因此，他对于每道题 i 规定了一个整数 a_i，表示题目 i 只在你总共解决了不超过 a_i 个问题（包括问题 i ）的情况下为你的最终成绩加上一分。

正式地，假设你在考试中解决了问题 $p_1, p_2, \cdots, p_k$。那么，你的最终成绩 s 会等于在 1 到 k 之间的满足 $k \leq a_{p_j}$ 的 j 的个数。

你已经意识到这场考试真正的第一道题目已经放在了你面前。因此，你想要选择一组题目来解决，从而最大化你的最终成绩。不要忘记这场考试有时间限制，而你必须有足够的时间来解决所有你选择的题目。如果存在多个最优解，任意输出一组即可。

例8：CF913D Too Easy Problem

- 首先给出一个结论：假设最终答案为 ans，那么 Ai 小于 ans 的题一定不做，而且一定是做了 ans 道题。
- 接下来考虑做法：我想到两种，这里只讲一种贪心的，另一种等二分的时候讲。
- 因为肯定只做 Ai 大于 ans 的题目，那么我们考虑把读入的数据按照 Ai 降序排序，然后枚举一个 i 从 n 到 1，表示我们只考虑 Ai > i 的数据。接下来，我们只需要维护 Ai > i 的数据的 Ti 的前 k 小和，这里使用 priority_queue 实现。标程里有详细解释。
- 注意到还需要输出方案，这个其实比较简单，在 i 和 ans 相等时直接终止循环即可。此时 q（也就是我们的优先）中的元素就是方案。

例9：CF1718A1 Burenka and Traditions

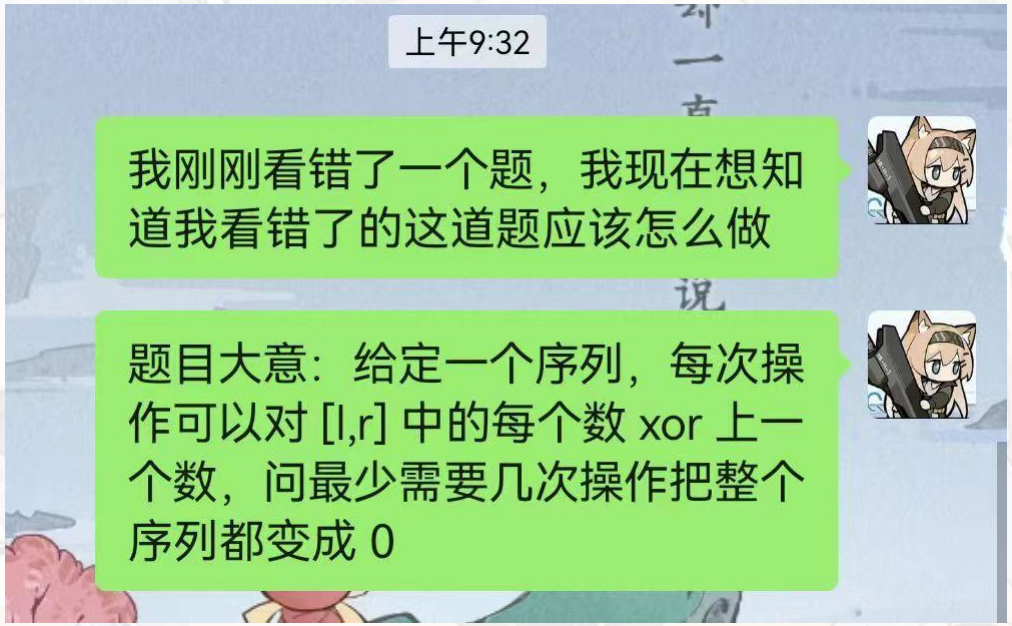
题意翻译

有 T 组数据，对于每一组数据，你有一个长度为 n 的数组 a，每一次操作可以选择一段区间 $[l,r]$，花费 $\left\lceil \frac{r-l+1}{2} \right\rceil$ 秒使区间内的数都异或 x，问最少需要几秒才能把数组中所有的元素都变成零。

例9：CF1718A1 Burenka and Traditions

- 这个题的第一步思路为拆分操作：从代价入手。
- 拆分完以后的操作为：
- （1）对单点异或 x
- （2）对相邻两个数异或 x
- 首先，只使用操作一，肯定能够用 n 的代价完成这个任务，那么只需要考虑如何使用操作二来减少代价
- 假如一段连续子序列的异或和为 0，那么我们可以减少一个代价（具体说明见我手写）
- 并且，只要发现一段子序列的异或和为 0，立刻减少一个代价，这样贪心是正确的

例9彩蛋


上午9:32
我刚刚看错了一个题，我现在想知道我看错了的这道题应该怎么做
题目大意：给定一个序列，每次操作可以对 [l,r] 中的每个数 xor 上一个数，问最少需要几次操作把整个序列都变成 0

例10：P2672推销员

题目描述

阿明是一名推销员，他奉命到螺丝街推销他们公司的产品。螺丝街是一条死胡同，出口与入口是同一个，街道的一侧是围墙，另一侧是住户。螺丝街一共有 N 家住户，第 i 家住户到入口的距离为 S_i 米。由于同一栋房子里可以有多家住户，所以可能有多家住户与入口的距离相等。阿明会从入口进入，依次向螺丝街的 X 家住户推销产品，然后再原路走出去。

阿明每走 1 米就会积累 1 点疲劳值，向第 i 家住户推销产品会积累 A_i 点疲劳值。阿明是工作狂，他想知道，对于不同的 X，在不走多余的路的前提下，他最多可以积累多少点疲劳值。

输入格式

第一行有一个正整数 N，表示螺丝街住户的数量。

接下来的一行有 N 个正整数，其中第 i 个整数 S_i 表示第 i 家住户到入口的距离。数据保证 $S_1 \leq S_2 \leq \cdots \leq S_n < 10^8$。

接下来的一行有 N 个正整数，其中第 i 个整数 A_i 表示向第 i 户住户推销产品会积累的疲劳值。数据保证 $A_i < 1000$。

例10：P2672推销员

- 这个题乍一看不太好做，你需要发现以下结论：
- 对于每一个 x，可能的答案只有两种：选择按 a 排序前 x 大的；选择按 a 排序前 x - 1 大的，然后再选择一个 s 相对较大的（也就是，推销花费的精力少，但是走路花费的多）
- 注意到这里最多只会舍弃一个推销花费少的元素，否则答案一定不会更优（想一想，肯定会有一个多走的路程被另一个抹掉了）
- 然后我们现在的问题变成了：
- （1）维护前 x 大的所有 a 的和，以及这些住户对应的 s 的最大值 ms
- （2）查询一个 s 相对较大的元素，要求是 a + s * 2 最大
- 第一个问题是好维护的，第二个问题就有些麻烦，这里我们详细展开讲第二个问题
- 容易发现，随着 x 变大，ms 一定是递增的，而且我们肯定只会查询 s 大于 ms 的数的 a + s * 2 的最大值，因此维护一个后缀最大值即可。

例11：CF484D Kindergarten

题意翻译

有一组数，你要把他分成若干连续段。每一段的值，定义为这一段 数中最大值与最小值的差。 求一种分法，使得这若干段的值的和最大。 N < 1e6, a[i] < 1e9。

输入输出样例

输入 #1 复制

```
5
1 2 3 1 2
```

输出 #1 复制

```
3
```

输入 #2 复制

```
3
3 3 3
```

输出 #2 复制

```
0
```

例11：CF484D Kindergarten

- 思考一下，大家会发现这样一个贪心结论：最优的划分方法中，每一个子段一定是单调的，否则，划分成两个单调的子段，答案一定不会更差。
- 但是还有个小问题：我们不知道，对于每一个波峰 or 波谷，放在哪里能使答案更优。这似乎不是贪心能做的事，所以我们考虑 dp。
- 设 f(i) 表示划分前 i 个数，能获得的最大的答案是多少。
- 接下来考虑状态转移：
- 如果 i-2,i-1,i 三个数是单调的，那么 f(i)=f(i-1)+abs(A(i)-A(i-1))
- 否则，考虑我们将第 i 个数划分在前一段还是后一段
- 划分在前一段：f(i)=f(i-1)
- 划分在后一段：f(i)=f(i-2)+abs(A(i)-A(i-1))
- 容易发现这里两种选择没有后效性，直接取max即可

例12 Ability To Convert

题意翻译

题目描述

亚历山大正在学习如何把十进制数字转换成其他进制，但是他不懂英文字母，所以他只是把数值按照十进制数字的方式写出来。这意味着他会用 10 代替英文字母 A。这样，他就会把十进制的 475 转换成十六进制的 11311（475=1·16^2+13·16^1+11·16^0）。亚历山大平静的生活着，直到有一天他试着把这些数字转换回十进制数字。

亚历山大记着他总是用较小的数字工作，所以他需要找到在 n 进制的基础下，用他的转换系统得出数字 k 的最小十进制数。 输入输出格式 输入格式：

第一行是一个正整数 n (2<=n<=10^9)，第二行有一个正整数 k ，满足数字 k中包含不超过 60 个数值。第二行整数中的没每一个数字都严格小于 n。

亚历山大保证答案存在且不超过 10^18，数字 k 不含前导 0. 输出格式：

输出一个数字 x —— 问题的答案。

例12 Ability To Convert

- 考虑在 n 进制下，要让转化成的十进制数尽可能小，就是要让划分的位数尽可能少，并且高位尽可能小。
- 那么贪心地想，从小到大划分，让尽可能多的（n 进制下的）位数划分成一位，这样是最好的。因为如果你有某一位可以划分到之前，却没有这样做，那么你后面的划分时，多了一位要划分的数，最终结果一定会变得更大。
- 然后就是模拟了，注意一些前导零之类的细节。

例13：Arthur and Brackets

题意翻译

一个长为2n的括号序列，左括号和右括号都是n个。

对于从左到右的第i个左括号，对与其配对的右括号有如下限制：

与其配对的右括号和这个左括号之间的距离需要在 [Li , Ri] 之间。

输入的第一行给出n（1<=n<=600）。

之后的n行，给出限制：Li、Ri。

如果满足限制的括号序列存在，输出任意一个即可。

否则输出"IMPOSSIBLE"（不包含引号）。

例13：Arthur and Brackets

- 贪心策略：凡是能匹配的，匹配了一定比不匹配更优秀。
- ~~讲完了~~
- ~~大家原谅一下老师23点51分做课件的精神状态~~
- 开玩笑的。其实思路就是这么个思路，具体证明的话，可以这样考虑：假设第 i 个括号在某一时刻可以匹配，那假如你延后了这个匹配，其实你相当于在左括号和右括号之间又插入了一个完整的括号序列，而这个括号序列丢到右括号右边也是完全可以的。
- 接下来就是模拟的过程啦，应该不用我细讲吧~~（其实我是想偷懒~~

例14 分组

题目描述

复制Markdown 展开

小可可的学校信息组总共有 n 个队员，每个人都有一个实力值 a_i。现在，一年一度的编程大赛就要到了，小可可的学校获得了若干个参赛名额，教练决定把学校信息组的 n 个队员分成若干个小组去参加这场比赛。

但是每个队员都不会愿意与实力跟自己过于悬殊的队员组队，于是要求分成的每个小组的队员实力值连续，同时，一个队不需要两个实力相同的选手。举个例子：$[1,2,3,4,5]$ 是合法的分组方案，因为实力值连续；$[1,2,3,5]$ 不是合法的分组方案，因为实力值不连续；$[0,1,1,2]$ 同样不是合法的分组方案，因为出现了两个实力值为 1 的选手。

如果有小组内人数太少，就会因为时间不够而无法获得高分，于是小可可想让你给出一个合法的分组方案，满足所有人都恰好分到一个小组，使得人数最少的组人数最多，输出人数最少的组人数的最大值。

注意：实力值可能是负数，分组的数量没有限制。

例14 分组

- 考虑从小到大加入每一个数，分为两种情况讨论：
- （1）没有任何在最后可以加入这个数的组，那么直接为这个数新增一个组即可
- （2）这个数可以连接在若干组的后面，那么选择这些可以连接的组中，人数最少的那一个
- 具体来说，使用sort+unique去重以后，你就可以将每一个数对应一个唯一的排名，排名至多有10^6 个，为每一个排名都开一个priority_queue即可

小总结

- 1、基于排列顺序的贪心，一般采用邻项交换法，结合冒泡排序的原理进行证明
- 2、基于性质发掘的贪心，一般是要大胆猜结论，然后采用反证法进行证明
- 3、积累了一些维护贪心的方法：排序以后按照顺序处理、使用priority_queue等数据结构、维护前缀或后缀最小值、dp、模拟等等