# 集训五题解

**Problem 1 poj 2631**

简要描述：求树的直径。即求树上2个点，使它们距离最远。

**Description**

Building and maintaining roads among communities in the far North is an expensive business. With this in mind, the roads are build such that there is only one route from a village to a village that does not pass through some other village twice.   
Given is an area in the far North comprising a number of villages and roads among them such that any village can be reached by road from any other village. Your job is to find the road distance between the two most remote villages in the area.

The area has up to 10,000 villages connected by road segments. The villages are numbered from 1.

**Input**

Input to the problem is a sequence of lines, each containing three positive integers: the number of a village, the number of a different village, and the length of the road segment connecting the villages in kilometers. All road segments are two-way.

**Output**

You are to output a single integer: the road distance between the two most remote villages in the area.

**Sample Input**

5 1 6

1 4 5

6 3 9

2 6 8

6 1 7

**Sample Output**

22

树的直径是有固定做法的题目，比较简单。就是先从任意一个点出发做一次树的BFS搜索，找到距离当前点最远的点，再从那个点出发再做一次BFS搜索，再次找到最远点，这次的最远距离就是树的直径了。这里解释一下为什么可以这样。首先树是不存在无圈的图，即任意两点之间有且只有一条路径。我们假设这条最长的路径的两个端点分别为A、B两点，我们选择的点为O。

如果O点在直径上，那么根据直径的定义，离它最远的点一定是A或者B，那么从A或者B点出发找最远的点一定是点B或者A，第二次搜索的最长距离一定是直径。

若O点不在路径AB上，假设离O的最远点是C，那么也就是OC>=OA,OC>=OB。另外因为是树，所有点之间都是存在路径的，C点和路径AB上的点一定有路径，不妨取最近点D，存在路径CD，且D点不是A、B点（A、B点一定是叶子结点，就是说和A、B点相邻的点只有一个，否则AB不是直径）。这种情况下，AB=AD+BD，AC=AD+DO+OC，BC=BD+DO+OC。根据不等式可得，AC+BC>2\*AB，再根据抽屉原理，至少存在AC>AB或者BC>AB,所以AB不是直径，矛盾，假设不成立。所以C一定是A、B两点中其中一点。所以上述方法找到的一定是直径。

这个是我当初遇到这种题别人告诉我用2次BFS我自己想的证明，不是很严谨，大概那个意思，文字描述确实费劲，又不懂画图，觉得不清楚或者要求严谨的可以去查下详细证明。

**Problem 2 xmu 1514**

关于xmu无法提交的话，因为这题代码量也就30行，重在思想。

取数游戏2

**Description**

给定一个由N个数字构成的序列，并且定义一个数字序列中的M值为这个序列中任意两个数字的差值的最大值，一个数字序列中的m值为任意两个数字的差值的绝对值的最小值。现在的问题是，让你移除K个数字，使得剩余的N-K个数字构成的序列的M+m值最小。

**Input**

第一行两个整数N（3<=N<=1,000,000）、K（1<=K<=N-2），为初始给定的序列长度以及要移除的数字个数。

接下来一行N个整数Vi（-5,000,000<=Vi<=5,000,000），代表初始序列的每个数的值。

**Output**

输出为一个整数，代表移除K个数字之后剩余序列M+m的最小值。

**Sample Input**

6 2

-5 8 10 1 13 -1

**Sample Output**

13

**Source**

Xmu

老实说我还真不知道他30行怎么解决的。我不会C++，所以也不会stl，有这方面的问题的还是请教一下陈剑峰学长。这题是去年校赛网络预赛的题，那时我是比赛结束前几分钟才过的。这题求的是使M和m的和最小，仔细思考和容易想到，选的N-K个数字一定是N个数按升序或者降序排列后，连续的N-K个数字，M为排序后选定子序列头尾数字之差，这个易求，m为连续数字之间的差值的最小值，这个是本题的关键。当时我选择的是用RMQ（范围最大/小值问题）然后超内存了……后来优化了一下最后内存擦边地过了。

虽然没有地方提交，苦思冥想还是想出了一个应该可行的方法，而且时间复杂度也会好很多。长度N的序列，删去K个数，剩下N-K个数，对于排序后，这样的连续序列只有K个，而算出每个序列的M都是O（1）的，关键是算出对应的m值，这里我们设排序后每个元素为a[i],d[i]=a[i+1]-a[i],这样得到了n-1个差值，我们要求出每个差值在哪个区间内是最小的，即对于每个d[i]，找到它左边第一个比它大的d值得坐标，右边也是。每个d[i]有其对应的left[i]和right[i]，且d[left[i]]>d[i],d[right[i]]>d[i]。找到这些left和right值可以在O（n）的时间内完成，这里是利用了栈来实现（xoj1534也是利用了这个来做的，是那次校赛的决赛题）。算出每个left和right后，我们枚举这n-1个值，当且仅当right[i]-left[i]+2>=N-K,这时d[i]就是left[i]和right[i]+1这个区间范围内的满足题目要求的连续序列的m值，我们可以填表式的填出K个m值，这样我们就得到了K个序列对应的M值和m值了，题目便可以做了。

这样考虑下来，算法的时间瓶颈反而在排序这个一定存在的操作O(nlogn)，算是比较好的算法了，就是不能提交，不能检验一下正确性，这题也可以问下陈剑峰学长看他怎么做的。

**Problem 3 uva 1203**

<http://uva.onlinejudge.org/index.php?option=com_onlinejudge&Itemid=8&category=497&page=show_problem&problem=3644>

试试使用优先队列解决这题。

Argus

问题描述：你的任务是编写一个称为Argus的系统。该系统支持一个register命令：

Register q\_num period

该命令注册一个触发器，它每period就会产生一次编号为q\_num的事件。

你的任务：模拟前k次事件，多个事件如果同时发生，先处理q\_num小的事件。

Input

The first part of the input are the register instructions to Argus, one instruction per line. You can assume the number of the instructions will not exceed 1000, and all these instructions are executed at the same time. This part is ended with a line of #.

The second part is your task. This part contains only one line, which is one positive integer K (≤ 10000).

Output

You should output the Q\_num of the first K queries to return the results, one number per line.

Sample Input

Register 2004 200

Register 2005 300

#

5

Sample Output

2004

2005

2004

2004

2005

**这题比较简单，只要模拟，再用优先队列(堆)K次即可。我的话是用堆模拟，然后构造一个结构体，记录每个数据的当前时间，每次递增时间，名字，用最小堆（按当前时间建堆）即可。每次取堆顶，输出，更新堆顶数据的当前时间，维护堆。**

**Problem 4 hdu 2147**

问题描述：

就是有一个游戏，在一个n\*m的矩阵中起始位置是（1，m），走到终止位置（n,1）；游戏规则是只能向左，向下，左下方向走，想走到终点的为获胜者。

Input

Input contains multiple test cases. Each line contains two integer n, m (0<n,m<=2000). The input is terminated when n=0 and m=0.

Output

If kiki wins the game printf "Wonderful!", else "What a pity!".

Sample Input

5 3

5 4

6 6

0 0

Sample Output

What a pity!

Wonderful!

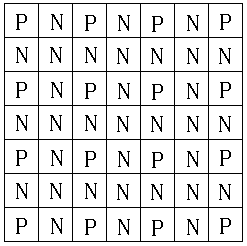
Wonderful!

这里稍微介绍一下一点博弈的知识，博弈问题一般有两种状态，先手必胜状态和先手必败状态，即前者状态下先手的人一定胜利，后者是一定失败。假设A状态能通过一步操作到达B状态，我们认为A是B的父状态，B是A的子状态。

必胜状态N：存在一个子状态是必败状态的状态。（这个人只要这么操作，对面就必败）

必败状态P：所有子状态都是必胜状态。（就是无论这个人怎么操作，下一个状态都是对方一定赢）

这题的（1,1）状态是必败状态，因为到他走的时候，已经在终点了（理解为对面到达了终点才轮到他走），这里的状态可以理解为子问题，即原问题规模为（n，m），状态（i，j）是子问题。



这是求出来的表格，那么这是怎么推出来的？

这种题目有两种做法，一是暴力求出一小部分（打表），然后根据表格找规律，总结出规律。

第二种就是按逻辑推理了，但这种一般比较难，很多题目的规律都很复杂，很难理论分析。

这里有个很好用的SG函数，可以打表，求出每个点的SG函数值，然后找规律（很好用的方法，去年省赛一金牌题，厦大两队金牌队就是这么做的）有兴趣的可以去查看看SG函数。

**Problem 5 poj 1088 经典的滑雪问题。**

滑雪

问题描述：Michael喜欢滑雪百这并不奇怪， 因为滑雪的确很刺激。可是为了获得速度，滑的区域必须向下倾斜，而且当你滑到坡底，你不得不再次走上坡或者等待升降机来载你。Michael想知道载一个区域中最长底滑坡。区域由一个二维数组给出。数组的每个数字代表点的高度。下面是一个例子

1 2 3 4 5

16 17 18 19 6

15 24 25 20 7

14 23 22 21 8

13 12 11 10 9

一个人可以从某个点滑向上下左右相邻四个点之一，当且仅当高度减小。在上面的例子中，一条可滑行的滑坡为24-17-16-1。当然25-24-23-...-3-2-1更长。事实上，这是最长的一条。

Input

输入的第一行表示区域的行数R和列数C(1 <= R,C <= 100)。下面是R行，每行有C个整数，代表高度h，0<=h<=10000。

Output

输出最长区域的长度。

Sample Input

5 5

1 2 3 4 5

16 17 18 19 6

15 24 25 20 7

14 23 22 21 8

13 12 11 10 9

Sample Output

25

拿到题目后，仔细思考便可以得到起始点和终止点一定是极值点（上下左右的点都比他大或者都比他小）。而且我们路径只能从高的数字往低的数字走，所以一定不存在圈。

由此我们可以先枚举每个点，找到若干极大值点，作为我们的起始点，然后从这些点出发BFS搜索，如果周围有比它小的点就走过去。

dp[i][j]表示从这些若干初始点出发，到达该点的最长路径长度，每个点最多到达4次，最多只能从4个方向到达，所以访问次数最多为4\*R\*C。