# Contest 7

# C<D<B<A<G<EFH

| | | |
|---|---|---|
| C | 65 | 80 |
| D | 47 | 115 |
| B | 7 | 73 |
| A | 5 | 73 |
| G | 3 | 52 |
| E | 0 | 1 |

# C. Insomnia

- 输入L, R，输出[L, R]内没有重复数字的数字个数
- 暴力for一遍
- O(n * 位数)的做法就行

# D. Fractal

- This degree of difficulty of the problem is about the level of the final of Advanced C.
- Use recursion.

# B. Naive Chess

- 题目大意：给定一个棋盘和一堆棋子，双方进行博弈，可选择一个棋子向左、向上、或者向左上移动，先移动一个棋子到（0,0）处的人就算赢。
- 传统的Nim游戏？如果这个游戏的定义变成，先把所有棋子移动到（0,0）的人算赢的话，就变成了传统的Nim游戏…但是这个并不是…
- 如何转变成传统的Nim游戏？（1,2）（2,1）定义为结束点即可。

# A. Indian Train

- There are two segments on the plain. The speeds are different among the two segments and the plain. Find the quickest way to travel from an end of the first segment to an end of the second segment.

# A. Indian Train

- Take a guess that the cost function of time is a quadric.
- Then we can use double trichotomy.

# A. Indian Train

- Suppose P is the point where Raj leave AB, and Q is the point where Raj gets on CD.
- =>Sadly, T is not a quadric function about P, Q…

$$T = \frac{AP}{v_1} + \frac{PQ}{v_w} + \frac{QD}{v_2}$$

$$\frac{PQ}{v_w} = \sqrt{(F_2(AP, QD))}$$

$$T = F_1(AP, QD) + \sqrt{(F_2(AP, QD))}$$

# A. Indian Train

- However, we can still get a lot information from the equation.
- Let’ s do some geometry imagination!
- We can proof it much more easily by combine algebra and geometry!

# B. Naive Chess

- 题目大意：给定一个棋盘和一堆棋子，双方进行博弈，可选择一个棋子向左、向上、或者向左上移动，先移动一个棋子到（0,0）处的人就算赢。
- 传统的Nim游戏？如果这个游戏的定义变成，先把所有棋子移动到（0,0）的人算赢的话，就变成了传统的Nim游戏…但是这个并不是…
- 如何转变成传统的Nim游戏？（1,2）（2,1）定义为结束点即可。

# G. Exciting Game　（出数学题好像真的很过分Orz）

- 题目简述：给定a, b, $X\_1$, 素数p. $X\_(i+1)=〖aX〗\_i+b\ (mod\ p)$
- 　　　　　　求最小的n使得$X_n$ = t, 如果无解输出-1. （0 ⩽ a, b, $X\_1$, t ⩽ p-1）

- 解法：　扩展欧几里得 + BSGS
- 　　　先处理特殊情况：若$X_1$ = t, 则n = 1; 若a = 0, 则判断b == t？
- 　　　然后分类讨论：若a=1,　$X\_n$= $X\_1$+(n−1)b, 用EXGCD可以求解
- 　　　若a ⩾ 2, $X\_n$= $a$^($n$−1) $X\_1$+$b$ ($a$^($n$−1)−1)/($a$−1) ($mod\ p$)=$t$
- 　　　设c为a−1的逆元, ($X\_1$+$bc$) $a$^($n$−1)=$bc$+$t$($mod\ p$)
- 　　　EXGCD解出$a$^($n$−1), 再用BSGS算一下n. 复杂度 O(√$p$).

# E. Easy birthday gift

- 题目大意：
- 就是现在定义f(x)表示满足x mod (a*b) == 0的有序正整数对(a, b)的数量, 对于给定的n, 求 f(1) + f(2) + ... + f(n)的值, (1 <= n <= 10^11)

# E. Easy birthday gift

- 首先这个题可以转换一下思路, f(1) + f(2) + … + f(n)实际上就是满足 a*b*c <= n的不同有序数对(a, b, c)的数量
- 那么不妨设 a <= b <= c显然 a <= n^(1/3)
- 那么可以暴力枚举a的值, 然后对于b满足 a <= b <= (n/a)^(1/2)
- 所以暴力枚举a和b的值那么此时c的数量为 n/(a*b) - b + 1个 (n/(a*b) >= b)
- 那么当a = b时, 如果c取b排列有1种, 如果c取 > b这样的c有 n/(a*b) - b 此时相当于排列(a, a, c)有3种
- 当a != b时, 如果c = b此时排列有3种, 如果c取c > b这样的n/(a*b) - b 种c, 相当于排列(a, b, c)有6种
- 所以枚举a, b的值然后判断a和b的关系即可得到此条件下的排列数量

# F. Cutting Polygon

- http://codeforces.com/contest/438/problem/C
- 简单多边形(n<=200)(不一定是凸的)，划分成三角形，三角形顶点都是原多边形的点。求划分方法种数。
- dp[i][j]

# F. Cutting Polygon

- First we label the vertex of polygon from 0 to $n - 1$.
- Then we let $f[i][j]$ be the number of triangulations from vertex $i$ to vertex $j$. (Suppose there is no other vertices and there is an edge between $i$ and $j$)
- If the line segment $(i, j)$ cross with the original polygon or is outside the polygon, $f[i][j]$ is just 0. We can check it in $O(n)$ time.
- Otherwise, we have, $f[i][j] = \sum_{i<k<j} f[i][k] * f[k][j]$ which means we split the polygon into the triangulation from vertex $i$ to vertex $k$, a triangle $(i, k, j)$ and the triangulation from vertex $k$ to vertex $j$. We can sum these terms in $O(n)$ time.

# F. Cutting Polygon

- Finally,the answer is $f[0][n - 1]$. It's obvious that we didn't miss some triangulation. And we use a triangle to split the polygon each time, so if the triangle is different then the triangulation must be different, too. So we didn't count some triangulation more than once.
- So the total time complexity is $O(n^3)$, which is sufficient for this problem.

# H. Naive Network

- 题目大意：动态最小生成树（可以离线）
- LCT？（~~我不会~~）
- 对询问进行分治！
- Solve(l,r)
  - Prepare(l,r)
  - Solve(l,mid)
  - Solve(mid+1,r)
- Solve中处理的时间复杂度要与r-l有关

# H. Naive Network

- Contraction操作
- 把待修改的边标记为-INF，做一遍MST，MST中的非-INF边显然是必须边。
- 已经用必须边连接的点，我们就不需要再进行寻找了，此时点数最大为s+1（s为待修改的边的个数）

# H. Naive Network

- Reduction
- 把待修改的边标记为INF，做一遍MST，不在MST中的非INF边显然永远也不会用到
- 这样的话，边数最大为n+s-1（n为点数，s为询问数）

# H. Naive Network

- 经过这两次操作，我们成功的把图的规模控制在
- 点数不超过s+1
- 边数不超过s+1+s-1=2*s
- s=r-l+1
- 在计算solve(l,r)时，1到l-1的修改都已经修改结束，所以我们就可以放心的递归！
- 整个算法的时间复杂度O(qlog2q)