

# 2020/07/11——2020/07/17周报

## 团队训练

2020.7.12 [2020牛客暑假多校训练营（第一场）](#) prob:4/7/10 rank:56/1115

2020.7.13 [2020牛客暑假多校训练营（第二场）](#) prob:4/9/11 rank:162/1158

## 林星涵

专题

比赛

题目

## 陶吟翔

专题

[LCA问题](#)

比赛

2020.07.11 [Alsing Programming Contest 2020](#) prob:4/5/6 rank:933

题目

- Educational Codeforces Round 91 C - Create The Teams
  - 分类：贪心
  - 题目大意：有 $n$ 个程序员要进行分组，每个程序员有能力值 $a_i$ 。现在要求一个组能力值最低的程序员的能力乘以总人数至少要有 $x$ 。问最多能分几个组
  - 数据范围：多组数据  $T \leq 1000$   $1 \leq n \leq 10^5$   $1 \leq a_i, x \leq 10^9$
  - 解题思路：显然每个组人越少越好，所以我们从大到小开始贪心，能一个人就一个人，不能就看能不能两个人、三个人，以此类推即可，时间复杂度 $O(n \log n)$
  - Comment：比较简单的贪心题
- Educational Codeforces Round 91 D - Berserk And Fireball

- 分类：分类讨论，模拟
- 题目大意：有 $n$ 个士兵排成一排，每个士兵有能力值 $a_i$ 且各不相同，你可以进行两种操作，一种是花费 $x$ 消灭连续的 $k$ 个士兵，一种是花费 $y$ 使两个相邻的士兵中能力值较低的被消灭，问能不能把给出的士兵消灭成特定位置的士兵留下来的排列，如果能，最小化花费
- 数据范围 $1 \leq k \leq n \leq 2 \times 10^5$   $1 \leq x, y \leq 10^9$
- 解题思路：由于保证了士兵能力值各不相同，我们先确定哪些士兵被留下了，然后以区间的形式消灭不留下的。对于一个区间，如果长度小于 $k$ 那么区间两端的值必须有一个大于最大值，否则删不干净，这时只能用第二种操作。如果区间长度大于等于 $k$ 要分类讨论，如果 $y \times k < x$ 那么显然应该用最大值一直删，否则把区间长度删成 $k$ 的倍数然后删光，这时还是要判断能不能用最大值删干净，如果不行就必须删到只剩 $k$ 然后用一次第一种操作
- Comment：题目本身的操作并不是很难，只要从士兵能力值各不相同的条件能够想到以区间为单位删掉不用留下的士兵，接下来就只需要注意细节了
- Codeforces Round #654 (Div. 2) D - Grid-00100
  - 分类：思维，模拟
  - 题目大意：给出 $n$ 和 $k$ 要求构造一个 $n \times n$ 的01矩阵，矩阵中有 $k$ 个1，且 $(\max\{R[i]\} - \min\{R[i]\})^2 + (\max\{C[i]\} - \min\{C[i]\})^2$ 最小 $R[i]$ 表示第 $i$ 行有几个1 $C[i]$ 表示第 $i$ 列有几个1
  - 数据范围：多组数据 $T \leq 100$   $1 \leq n \leq 300$   $1 \leq k \leq n^2$   $\sum n^2 \leq 10^5$
  - 解题思路：显然我们要同时使每一行和每一列的1的个数都尽量平均，首先想到的就是先往从左上到右下的对角线上放，之后多试几次就会发现可以依然按照这个规律来放，先放右上方的 $n-1$ 个，剩下一个放到左下角，然后以此类推可以保证答案最小
  - Comment：一道不错的思维题，代码难度也比较低，不过感觉放在D题有点偏简单（1600分的题）

## 郭衍培

### 专题

### 比赛

### 题目

## 本周推荐

林星涵：

陶吟翔：

郭衍培：

[题目链接](#)

题目大意 $n$ 行 $m$ 列的方格，每行被划分为连续的若干块。在每块中选一个地方填1。设第 $i$ 列有 $q_i$ 个1，

使得 $\sum_{i=1}^n q_i^2$ 最大。求这个最大值

数据范围 $1 \leq m, n \leq 100$

解题思路：大致的思路是区间dp $dp[i][j]$ 表示所有完整包含在 $[i,j]$ 列的块所能做出的贡献，最终答案即为 $dp[1][m]$ 首先证明，在最优策略中，对于 $[i,j]$ 区间，一定存在一列 $k$ 满足所有完整包含在 $[i,j]$ 列且覆盖 $k$ 的块都在 $k$ 填了1，因为我们显然希望1尽量地聚集。所以 $dp[i][j] = \max\{dp[i][k-1] + dp[k+1][j] + num[i][j][k]^2\}$

From:

<https://wiki.cvbbacm.com/> - CVBB ACM Team

Permanent link:

<https://wiki.cvbbacm.com/doku.php?id=2020-2021:teams:hotpot:200711-200717&rev=1594971700> 

Last update: **2020/07/17 15:41**