

高阶程序设计第四次报告

陈德创 19030500217

计算机科学与技术学院

日期：2020 年 5 月 18 日

目录

1	总览	2
2	P1042、乒乓球	2
3	P2670、扫雷游戏	2
4	P1563、玩具谜题	2
5	P4924、魔法少女小 Scarlet	3
5.1	题目复述	3
5.2	解题思路	3
5.3	复杂度分析	3
5.4	结果	3
6	P1067、多项式输出	3
7	P1065、作业调度方案	4
7.1	题目复述	4
7.2	解题思路	4
7.3	复杂度分析	4
7.4	结果	4
8	P1786、帮贡排序	4
9	个人总结	4

1 总览

状态	题号	题目名称	显示算法	标签	难度	通过率
✓	P1042	乒乓球	NOIp普及组	2003	普及-	<div><div></div></div>
✓	P2670	扫雷游戏	NOIp普及组	2015	入门	<div><div></div></div>
✓	P1563	玩具谜题	NOIp提高组	2016	普及-	<div><div></div></div>
—	P1601	A+B Problem (高精)			普及-	<div><div></div></div>
—	P1303	A*B Problem			普及-	<div><div></div></div>
✓	P1009	阶乘之和	NOIp普及组	1998	普及-	<div><div></div></div>
✓	P4924	[1007]魔法少女小Scarlet			普及+/提高-	<div><div></div></div>
✓	P1328	生活大爆炸版石头剪刀布	NOIp提高组	2014	普及-	<div><div></div></div>
✓	P1518	[USACO2.4]两只塔姆沃斯牛 The Tamworth Two	USACO		普及+/提高-	<div><div></div></div>
✓	P1067	多项式输出	NOIp普及组	2009	普及-	<div><div></div></div>
—	P1098	字符串的展开	NOIp提高组	2007	普及+/提高-	<div><div></div></div>
✓	P1065	作业调度方案	NOIp提高组	2006	普及+/提高	<div><div></div></div>
✓	P1786	帮贡排序			普及+/提高	<div><div></div></div>
—	P1591	阶乘数码			普及-	<div><div></div></div>
—	P1249	最大乘积			普及+/提高-	<div><div></div></div>
—	P1045	麦森数	NOIp普及组	2003	普及+/提高-	<div><div></div></div>

2 P1042、乒乓球

循环计数，然后判断一下是不是大于 11 分并且领先了 2 至少分就可以了，因为有两种计数方式，所以要记两次数。注意最后一场 0 : 0 也要输出（所以我第一次 WA 了一半）。

 LetMyself 05-14 11:16:35	Accepted 100	P1042 乒乓球	85ms / 776.00KB / 1.14KB C++
 LetMyself 05-14 11:15:11	Unaccepted 50	P1042 乒乓球	76ms / 912.00KB / 1.22KB C++

3 P2670、扫雷游戏

时间太长了。。。。。其实就是遍历，然后每个格子如果是雷的话就更新周围几个格子的计数。

 LetMyself 2019-10-15 09:10	Accepted 100	P2670 扫雷游戏	31ms / 800.00KB / 481B C++
---	-----------------	------------	----------------------------

4 P1563、玩具谜题

结构体储存小人的朝向和姓名，然后模拟走一遍就可以了。和约瑟夫环差不多，但是简单许多。环可以用取模来模拟，但是取模的时候要注意。我第一次 WA 就是因为我下标从 1 开始但是取模直接取模了而没有处理。建议下标从 0 开始，取模方便。

 LetMyself 02-23 23:11:50	Accepted 100	P1563 玩具谜题	⌚ 436ms / 📦 4.88MB / 📄 4698 C++
 LetMyself 02-23 23:11:20	Accepted 100	P1563 玩具谜题	⌚ 446ms / 📦 4.62MB / 📄 5018 C++
 LetMyself 02-23 23:08:00	Unaccepted 90	P1563 玩具谜题	⌚ 458ms / 📦 4.77MB / 📄 4708 C++

5 P4924、魔法少女小 Scarlet

5.1 题目复述

给定一个 $n*n$ 的矩阵，进行 m 次操作，每次操作将会选择一个坐标 (x,y) 和半径 r 还有方向。每次操作要求对以 (x,y) 为中心的 $2r+1$ 阶方阵进行顺时针或者逆时针 90° 旋转，输出 m 次操作后的矩阵。

5.2 解题思路

模拟。对于每次操作我们是对一个矩阵进行操作，可以分解为对 r 个正方形进行分别进行旋转操作。我们以顺时针旋转为例（逆时针相反就可以了），将正方形分为上边、下边、左边和右边。旋转实际上就是 上边 \rightarrow 右边, 右边 \rightarrow 下边, 下边 \rightarrow 左边, 左边 \rightarrow 上边 的一个变化过程。

为了可以直接在矩阵上操作，我们可以先对上边和右边互换，然后互换完的上边（其中的数据为原来右边的数据）和下边互换，然后第二次互换完的上边（其中的数据为原来下边的数据）和左边互换。就这样实际上就完成了旋转。

需要注意的是对于四个角的处理，以上边和右边的互换为例，对于坐标 (x,y) 和半径 i ，我们可以只对类似于 $[x-i, x+i], y+i$ 和 $x+i, [y+i, x-i]$ 的数据进行互换，即每个角属于一条边，每条边只有一个角。

5.3 复杂度分析

m 次操作，每次对 $r*r$ 的数据进行操作， r 最大是 n ，所以复杂度是 $O(mn^2)$

5.4 结果

 LetMyself 05-14 17:34:52	Accepted 100	P4924 [1007]魔法少女小Scarlet	⌚ 1.61s / 📦 1.57MB / 📄 1.21KB C++
---	-----------------	--------------------------	-----------------------------------

6 P1067、多项式输出

模拟，注意特判挺多的。比如首位不能补正号，其余的如果是正数则要补正号，如果是 ± 1 则不能输出 1，如果是 0 则要则要整项省略，0 次项的输出不要有 x^0 等等。

 LetMyself 2019-06-21 21:24	Accepted 100	P1067 多项式输出	31ms / 928.00KB / 342B C++
 LetMyself 2019-06-21 21:22	Unaccepted 90	P1067 多项式输出	30ms / 788.00KB / 339B C++

7 P1065、作业调度方案

7.1 题目复述

题干过长，这里附上链接[P1065 作业调度方案](#)

7.2 解题思路

这题挺麻烦的，模拟加贪心，模拟题大多都挺麻烦。维护一个 $least[]$ 数组， $least[i]$ 表示第 i 个工件至少什么时候开工，初始化为 1，表示从 1 开工。用 $time[n][max_time]$ 表示机器的可用时间， $time[i][j]$ 表示第 i 个机器第 j 时可用。然后每个工件操作时，从所需机器的 $least[i]$ 开始遍历数组，尝试找到一块可以利用的时间。然后更新 $least[i]$ 就可以了。

7.3 复杂度分析

时间复杂度，由于每次操作时间长度不超过 20，所以 $max_time = 20 * n$ ，每次最差查询整个数组（事实上这种情况几乎不会出现），复杂度为 $O(20mn^2)$ ，常数 20 应该省去，但是 $m, n < 20$ ，所以 20 还是很大的常数的相对于数据规模。数据规模很小，完全可以接受。

7.4 结果

 LetMyself 05-14 18:14:05	Accepted 100	P1065 作业调度方案	50ms / 772.00KB / 1.11KB C++
---	-----------------	--------------	------------------------------

8 P1786、帮贡排序

这题很坑啊，实际上是排两次序，第一次确定职位，第二次确定输出顺序。

 LetMyself 05-14 19:03:03	Accepted 100	P1786 帮贡排序	78ms / 864.00KB / 1.38KB C++
 LetMyself 05-14 18:58:22	Unaccepted 30	P1786 帮贡排序	75ms / 756.00KB / 1.38KB C++

9 个人总结

有一些之前刷过的题，源程序找不到了。

模拟好麻烦。不过模拟也算是最常用的方法了吧。之前就听说学好模拟、枚举和搜索，基本上 noip 省二，运气好省一。其实先模拟、枚举，从简单的暴力中往往可以找到接近正解的思路。

做了挺长时间，主要是细节把控不住，导致一直在 `debug` 还调不出来。看来自己代码能力有待加强。