

2019 CSP-S 模拟题（第一试）

年 月 日

（请选手务必仔细阅读此页内容）

一. 题目概况

中文题目名称	清扫	购物	幸运数
英文题目名称	clean	shopping	prd
可执行文件名	clean	shopping	prd
输入文件名	clean.in	shopping.in	prd.in
输出文件名	clean.out	shopping.out	prd.out
每个测试点时限	1 秒	1 秒	1 秒
内存上限	256MB	256MB	256MB
测试点数目	20	20	20
每个测试点分值	5	5	5
结果比较方式	全文比较	全文比较	全文比较
题目类型	传统	传统	传统

二. 提交源程序文件名

对于 Pascal 语言	clean.pas	shopping.pas	prd.pas
对于 C 语言	clean.c	shopping.c	prd.c
对于 C++语言	clean.cpp	shopping.cpp	prd.cpp

三. 编译命令

对于 Pascal 语言	fpc clean.pas	fpc shopping.pas	fpc prd.pas
对于 C 语言	gcc -o clean clean.c -lm	gcc -o shopping shopping.c -lm	gcc -o prd Prd.c -lm
对于 C++语言	g++ -o clean clean.cpp -lm	g++ -o shopping shopping.cpp -lm	g++ -o prd prd.cpp -lm

四. 注意事项：

1. 文件名（程序名和输入输出文件名）必须使用小写。
2. 选手提交以自己编号命名的文件夹，文件夹内包含 3 个源文件(.c, .cpp, .pas),

并在文件夹下建立 3 个相应的子目录，并将 3 个对应的源程序分别放入对应的子文件夹中，所有名字必须使用小写；例如：

```
001          编号
|---clean
|          |---clean.cpp
|---shopping
|          |---shopping.cpp
|---prd
|          |---prd.cpp
|---clean.cpp
|---shopping.cpp
|---prd.cpp
```

3. C/C++中函数 `main()` 的返回值类型必须是 `int`，程序正常结束时的返回值必须是 `0`。
4. 题目简单，请认真对待，争取三位数。
5. 每道题源代码长度限制均为 50KB。
6. 每道题的数据都有一定梯度。请尽量优化算法，争取拿高分。
7. 评测在 `linux` 系统下全国评测机和 `windows` 下的 `lemon` 分别评测。
8. 编译时不打开任何优化选项。
9. 建议最后 10 分钟不要再编程，检查一下提交的文件夹中的代码是否符合要求，检查文件名，输入输出文件名，数据类型，数据精度，空间限制，赋初值等是否按试卷上的要求来做的，一定要杜绝一切的不小心的人为错误，显然这种错误是致命的。
10. 做题时，审题是关键，必须深入与全面，学过的知识与做过的题都是分析问题的有利武器；编写代码要细致，多写函数，便于调试，只有这样，才能达到你的期望。

细节决定成败！

1、清扫 (clean)

【问题描述】

XX 市为了创建文明城市, 决定扫马路啦…现在 XX 市创卫会将马路的长度分为 len 段, 但觉得一次扫完太木有意思了, 所以要求扫 n 次, 每次扫 $Li \sim Ri$ 的这一段, 由于大家热情很高, 就没有认真统计汇总, 一段马路可能被扫多次, 也可能没有被扫, 所以 XX 市创卫会想知道有哪些位置没有被扫到。

【输入】

输入文件的第一行有两个整数 len, n , 数据间以空格隔开;

接下来 n 行, 每行有两个整数 $Li \ Ri$, 数据间以空格隔开。

【输出】

输出文件仅有一行, 表示没有被扫到的段数。

【输入输出样例】

clean. in	clean. out
6 2 1 2 5 6	2

【样例解释】

只有 3 和 4 没有被扫到。

【数据规模与约定】

30%的数据满足: $1 \leq len, n \leq 1000$

60%的数据满足: $1 \leq len, n \leq 100000$

100%的数据满足: $1 \leq len, m \leq 500000$ 。

2、购物 (shopping)

【问题描述】

果果喜欢购物，他尤其喜欢那种横扫一片商店的快感。最近，他打算对 WH 市商业区的商店实行他疯狂的购物计划。WH 市的商业区就是一条街，这条街上有 n 个商店，果果打算进攻 m 次，每次扫荡第 $L_i \sim R_i$ 个商店，果果会把经过的每个商店扫荡一空（换句话说，就是一个商店不会被算两次），因为连续地扫一片商店是很爽的，所以果果把一次扫荡的 happy 值定义为所有连续的一段被扫空的商店 happy 值之和的平方的和，已被扫空的不再计算，（定义下图所有商店的 happy 值为 1）

如图：

第一次行动：

1	2	3	4	5	6	7	8	9	10	11	12	13	14
---	---	---	---	---	---	---	---	---	----	----	----	----	----

扫荡4-9号商店，得到 $6^2=36$ 的Happy值

第二次行动：

1	2	3	4	5	6	7	8	9	10	11	12	13	14
---	---	---	---	---	---	---	---	---	----	----	----	----	----

扫荡2-12号商店，得到 $2^2+3^2=13$ 的Happy值

现在你不经意间得知了果果的购物计划，而你需要将这些计划排序并求出果果最多获得的 happy 值之和。

【输入】

第一行为 n 和 m ，意义如描述之所示

接下来一行 n 个数第 i 个数表示第 i 个商店的 happy 值

接下来 m 行，每行两个数

$L_i R_i$ 表示果果第 i 次行动要扫荡第 L_i 到第 R_i 个商店

【输出】

一行，包含一个数即为果果最多获得的 happy 值之和

【输入输出样例】

shopping. in	run. out
14 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 4 9 2 12	121

【样例解释】

先扫荡 2~12 号商店, 得到 121happy 值, 然后扫荡 4~9 号商店, 得到 0happy 值, 总 happy 值为 121.

可以证明没有比这个方案更优的方案.

【数据范围】

30%的数据满足: $n \leq 10, m \leq 8$

60%的数据满足: $n \leq 1000, m \leq 1000$

100%的数据满足: $n \leq 5000, m \leq 1000000, \text{happy 值} \leq 100$

3、幸运数（prd）

【问题描述】

果果试图对他的资料进行加密。他认为在十进制下只由 4 和 7 组成的数是“幸运的”，而幸运数就是由一个或者多个“幸运的”数字的积组成的数。比如，47、49 和 112（ $112=4\times 4\times 7$ ）都是幸运数。现在果果的算法中需要这样一个模块，帮助他计算 $[A, B]$ 中有多少个幸运数。作为果果的好友，他希望你能够帮助他实现这个模块。

【输入】

输入文件包含多组测试数据。

第一行是一个正整数 T ，表示测试数据的组数。

每组测试数据包含一行两个整数 A 和 B ，表示待求的区间。

【输出】

对于每组测试数据，在单独的一行内输出一个整数作为答案。

【输入输出样例】

prd. in	prd. out
4	0
1 2	0
88 99	1
112 112	10
1 100	

【样例解释】

$112=4\times 4\times 7$ ，所以 $[112, 112]$ 中幸运数恰有一个。

$[1, 100]$ 中的幸运数为 4, 7, 16, 28, 44, 47, 49, 64, 74 和 77。

【数据范围与约定】

30%的数据满足： $A, B\leq 1000$

100%的数据满足： $T\leq 7777, 1\leq A\leq B\leq 10^{12}$