Public NOIP Round #3

普及组

时间：2022 **年** 10 **月** 22 **日** 8:30 ~ 12:00

题目名称	数字	因子	移除石子	抓内鬼
题目类型	传统型	传统型	传统型	传统型
可执行文件名	`number`	`divisor`	`stone`	`catch`
每个测试点时限	1.0 秒	1.0 秒	1.0 秒	3.0 秒
内存限制	512 MiB	512 MiB	1 GiB	1 GiB
子任务数目	10	4	10	4
测试点是否等分	是	否	是	否

提交源程序文件名

对于 C++ 语言	`number.cpp`	`divisor.cpp`	`stone.cpp`	`catch.cpp`

编译选项

对于 C++ 语言	`-lm -O2`

注意事项：

1. C++ 中函数 main() 的返回值类型必须是 int，值必须为 0。
2. 对于因未遵守以上规则对成绩造成的影响，相关申诉不予受理。
3. 若无特殊说明，输入文件中同一行内的多个整数、浮点数、字符串等均使用一个空格进行分隔。
4. 若无特殊说明，结果比较方式为忽略行末空格、文末回车后的全文比较。
5. 程序可使用的栈空间大小与该题内存空间限制一致。
6. 在终端下可使用命令 `ulimit -s unlimited` 将栈空间限制放大，但你使用的栈空间大小不应超过题目限制。
7. 评测时采用的机器配置为 Intel(R) Xeon(R) Platinum 8272CL CPU @ 2.60GHz (QOJ 与 Public Judge)。使用的操作系统为 Ubuntu 18.04，编译器版本为 g++ 11.1.0 (Ubuntu 11.1.0-1ubuntu1 18.04.1)，上述时限以此为准。
8. **特别注意，在提交时不需要使用文件输入输出。选手应在标准输入中读入数据，并将答案输出至标准输出。**

数字（number）

【题目描述】

dottle 是机器人。

一天，dottle 在黑板上看到了一个十进制整数 n，保证 n 的十进制表示中没有 0。

dottle 认为，3 的倍数是好的，于是他想要擦掉一些数位上的数字来让 n 变为一个好的数字。

擦数字的过程可以看作删掉 n 十进制表示中的一个数字，然后将剩余部分依次拼接，比如 114514 擦掉从左往右的第三位会变成 11514。

注意空的黑板是不合法的，也就是说，假设 n 的十进制表达共有 k 位，执行的删除操作不能超过 $k-1$ 次。

dottle 想要知道，他最少需要擦掉几个数字，才能使 n 变为一个好数，或者告诉他无解。

【输入格式】

一行一个正整数 n。

【输出格式】

若有解，输出一行一个整数，表示答案。

否则，输出 dottle bot。

【样例 1 输入】

```
114514
```

【样例 1 输出】

```
1
```

【样例 1 解释】

删掉从左向右第三个位置上的 4 后数字变为 11514，是一个 3 的倍数。

【样例 2 输入】

```
369
```

【样例 2 输出】

```
0
```

【样例 2 解释】

369 是 3 的倍数。

【样例 3 输入】

```
11
```

【样例 3 输出】

```
dottle bot
```

【样例 3 解释】

注意不能擦掉所有数字，所以无解。

【样例 4 输入】

```
2839592836665555555
```

【样例 4 输出】

```
2
```

【子任务】

本题共 10 个测试点，每个测试点 10 分。对于所有测试点，保证 $1 \leq n \leq 10^{18}$ 且 n 的十进制表达中不含 0。

具体范围如下：

测试点编号	$n \leq$	特殊性质
$1 \sim 2$	9	无
$3 \sim 5$	99	
6	10^5	
$7 \sim 9$	10^{18}	n 的十进制表达长度为 18，且每一位在 $[1,9]$ 之间均匀随机
10		无

因子（divisor）

【题目描述】

今天是 YQH 的生日，她得到了一个长度为 n 的正整数序列 $a_1, a_2, \ldots, a_n$ 作为生日礼物。

然而，YQH 并不对这个序列满意，因为这个序列可能不合法。

具体的，一个序列合法，当且仅当存在一个大于 1 的整数 k，使得序列里每个元素都是 k 的倍数。

为了让 YQH 满意，你需要找到一个 $a_1, a_2, \ldots, a_n$ 的子序列，使得这个子序列是合法的。$b_1, b_2, \ldots, b_m$ 称为 $a_1, a_2, \ldots, a_n$ 的子序列当且仅当，你可以从 $a_1, a_2, \ldots, a_n$ 删去若干个（可以是 0 个）元素后得到 $b_1, b_2, \ldots, b_m$。

符合条件的子序列可能很多，所以 YQH 只想要你找到，总和最大的合法子序列的总和。**注意**，子序列可以取空集，且空集是合法的。

【输入格式】

第一行一个正整数 n。

接下来 n 行，每行一个正整数。第 i 行的数表示 a_{i-1}。

【输出格式】

输出一个整数表示答案。

【样例 1 输入】

```
4
1
1
1
1
```

【样例 1 输出】

```
0
```

【样例 1 解释】

唯一合法的子序列是空集。

【样例 2 输入】

```
6
1
2
3
4
5
6
```

【样例 2 输出】

```
12
```

【样例 2 解释】

一种合法的子序列为 $[2,4,6]$，它们都有因子 2，可以证明没有总和更大的合法子序列。

【样例 3 输入】

```
10
28851
8842
9535
2311
25337
26467
12720
10561
8892
6435
```

【样例 3 输出】

```
56898
```

【样例 4】

见选手目录下 ***divisor/divisor4.in*** 与 ***divisor/divisor4.ans***。

【样例 5】

见选手目录下 ***divisor/divisor5.in*** 与 ***divisor/divisor5.ans***。

【子任务】

子任务编号	$n \leq$	$a_i \leq$	特殊性质	分值
1	18	10^9	无	20
2	1000	10^5		20
3		10^9	A	20
4			无	40

特殊限制 A：保证所有 a_i 都是质数。
对于所有数据，保证 $1 \leq n \leq 1000, 1 \leq a_i \leq 10^9$。

移除石子（stone）

【题目描述】

你正在玩一个名为“移除石子”的小游戏。

平面直角坐标系上有 n 颗石子，放置在整点（横纵坐标均为整数的点）上，第 i 颗石子的坐标为 (x_i, y_i)。保证 n 颗石子的坐标互不相同。保证 n 为偶数。

你需要操控一个机器人移除所有的石子。你要做恰好 $n/2$ 次操作，每次操作中：

- 首先，你在平面上画出一个**正方形**。正方形的边必须与坐标轴平行。正方形的顶点**可以不是**整点。
- 机器人会移走所有在正方形内部（包括边界上）的石子。

因为机器人有两只机械臂，你也不想让机器人太闲或者太累，所以要求每次你画出的正方形里**恰好有两颗**石子。这也意味着，做完所有操作后，所有石子全部被移除了。

石子被移走后就不在这个平面上了，不会对后续操作造成影响，你可以认为它们被丢进了垃圾桶里。

你想知道，是否存在一种合法方案？如果存在的话，请你输出任意一组合法方案。

【输入格式】

本题单个测试点内有多组数据。

第一行一个正整数 T 表示数据组数。

对于每组数据：第一行为一个正整数 n，表示石子的个数。

接下来 n 行，每行两个整数 x_i, y_i，描述一颗石子的坐标。

【输出格式】

对于每组数据：

如果不存在方案，输出 `No`。

否则，第一行输出 `Yes`。

接下来输出 $n/2$ 行，第 i 行四个实数 x_1, y_1, x_2, y_2。$(x_1, y_1), (x_2, y_2)$ 是第 i 次操作时，你画出的正方形的任意两个**相对**的顶点。你需要按照操作的时间顺序输出。

输出的实数应当：

- 用十进制形式输出，不能用科学计数法。
- 至多保留四位小数。

如果有多种方案你可以输出任意一种。`No` `Yes` 对大小写不敏感，也就是 `YES` `nO` 也算对。

【如何知道你的输出是否正确】

如果你熟悉命令行 / 终端操作：下发文件中有两个文件 `testlib.h` 和 `checker.cpp`，将其置于同一目录下编译，然后运行 `checker input output output` 即可，这里 `input` `output` 分别是输入文件和你的输出。

如果你不熟悉命令行：没关系！你只需要严格按照下面的步骤执行就可以了。

1. 先把你要测试的输入数据改名为 `stone.in`，你的输出文件改名为 `stone.out`。
2. 把第一步里的两个文件复制到同一个文件夹（把这个文件夹叫做工作文件夹）里。
3. 下发文件里有四个文件 `testlib.h`，`checker.cpp`，`run_windows.cpp`，`run_linux.cpp`。如果你用的是 Windows 系统，请你删掉 `run_linux.cpp`；Linux 则删掉 `run_windows.cpp`。下面我们都假设你用的是 Windows 系统，如果不是的话，只需要把涉及到 `run_windows.cpp` 的步骤全部改为 `run_linux.cpp` 就可以了。
4. 上面你删掉了一个文件，还会剩下三个文件。你需要把剩下的三个文件复制到工作文件夹里。
5. 在工作文件夹里，编译 `checker.cpp`，如果你使用 Dev-C++ 的话，快捷键是 F9。如果无误，应该会看到在工作文件夹下里生成了文件 `checker.exe`。
6. 在工作文件夹里，编译并运行 `run_windows.cpp`，如果你使用 Dev-C++ 的话，快捷键是 F11。如果这个程序运行之后显示 “OK”，则你的输出是正确的。否则你的输出不正确。注意，这一步的程序运行时间可能比较长，请耐心等待。

【样例 1 输入】

```
1
4
1 1
2 2
5 5
6 6
```

【样例 1 输出】

```
Yes
2 2 5 5.0000
1 1 6 6.000
```

【样例 1 解释】

第一次，我们移走了第二颗石子和第三颗石子。第二次，我们移走了第一颗石子和第四颗石子。

输出不唯一，比如下面的输出也合法：

```
yEs
0.9999 0.9999 2.0001 2.0001
-233 -1.0 233.000 465.00
```

在上面的输出中，第一次，我们移走了第一颗石子和第二颗石子。第二次，我们移走了第三颗石子和第四颗石子。

但是下面的输出不合法：

```
Yes
1 1 2 2
-1e+5 -1e+5 1e+6 1e+6
```

因为不能用科学计数法输出。

下面的输出也不合法：

```
Yes
1 1 5 5
2 2 6 6
```

因为第一次删的时候，正方形内部和边界上一共有 3 个点 $(1,1),(2,2),(5,5)$。

下面的输出也不合法：

```
Yes
1 1 1 2
5 5 6 6
```

因为 $(1,1)$ 和 $(1,2)$ 不可能是任何边平行于坐标轴的正方形的**对角**。

下面的输出也不合法：

```
Yes
1 1 2 2
5 5 6 6.00000
```

因为至多只能输出 4 位小数。

【样例 2 输入】

```
1
4
0 0
100000000 200000000
200000000 100000000
400000000 400000000
```

【样例 2 输出】

```
Yes
100000000 100000000 200000000 200000000
-0.1 -0.100 400000000.0 400000000.0
```

【样例 2 解释】

注意输出 `1e+8` 或 `1e8` 等科学计数法形式的实数是不合法的。

【样例 3】

见选手目录下 ***ex_stone3.in*** 与 ***ex_stone3.ans***。样例 3 满足测试点 5 的性质。

【样例 4】

见选手目录下 ***ex_stone4.in*** 与 ***ex_stone4.ans***。样例 4 满足测试点 8, 9 的性质。

【子任务】

本题 10 个测试点，每个测试点 10 分。

对于所有测试点，保证 $1 \le T \le 60, 2 \le n \le 3\,000, 0 \le x_i, y_i \le 10^9$。保证 n 是偶数，保证石子的坐标互不相同。

表中“数据随机”的含义为石子的横纵坐标均在 $[0, 10^9]$ 随机生成。

测试点编号	$T \leq$	$n \leq$	特殊性质
$1, 2$	1	6	$x_i = 2i, y_i = 2i$
3		3 000	
4		6	$x_i = 0$
5		3000	
$6, 7$		10	数据随机
$8, 9$	20	500	
10	60	3 000	无

抓内鬼（catch）

【题目描述】

UR#24 的题面被内鬼偷走了!

为了找回丢失的题面，uoj 管理员决定和 pjudge 管理员合作，让内鬼无路可逃。

内鬼在一个 n 个点 m 条边的简单无向连通图上行走，他从 1 号点出发，目标是 n 号点。

uoj 和 pjudge 分别抓了 k 和 $n-k$ 个壮丁。图上的每个点会恰好分配一个壮丁，负责盘问来往行人。因为人流量不同，一个人经过第 i 个点需要花费的时间是 t_i 。经过一条边的时间可以忽略不计。

uoj 的壮丁很清楚其他 uoj 的壮丁都是鸽子，pjudge 的壮丁也很清楚其他 pjudge 的壮丁都是鸽子，但他们相互不知道对方是不是鸽子。所以，只有当内鬼经过的一条边的两边的壮丁来自同一个 oj 时，他才会被抓住。

你需要构造一个分配壮丁的方案，使得对于任意一条 1 到 n 的最短路，内鬼走这条路都会被抓住。或者判断无解。

【输入格式】

第一行三个正整数 n, m, k 。

第二行 n 个正整数 $t_1, t_2, \cdots, t_n$ 。

接下来 m 行，每行两个正整数 u_i, v_i ，表示无向图中的一条边。

【输出格式】

如果存在合法方案，那么输出一个长度为 n 的字符串 s ，其中 $s_i \in \{'P', 'U'\}$ 表示第 i 个点的壮丁是来自 pjudge 还是 uoj 。

否则，输出 `impossible` 。

【样例 1 输入】

```
3 2 0
1 1 1
1 2
2 3
```

【样例 1 输出】

```
PPP
```

【样例 1 解释】

uoj 一个壮丁都没有抓到！但是这样就使得每条边的两边的壮丁都来自 pjudge，因此内鬼走任意一条边都会被抓住。

【样例 2 输入】

```
2 1 1
1 1
1 2
```

【样例 2 输出】

```
impossible
```

【样例 2 解释】

uoj 和 pjudge 的壮丁互相认为对方会认真盘查，于是自己当鸽子，结果内鬼跑掉了！

【样例 3 输入】

```
8 9 4
3 3 1 2 2 3 2 1
1 2
1 3
1 4
2 5
3 6
4 7
5 8
6 8
7 8
```

【样例 3 输出】

```
PUPUPPUU
```

【样例 4】

见下发文件中的 ***ex_catch4.in*** 和 ***ex_catch4.ans***。

【样例 5】

见下发文件中的 ***ex_catch5.in*** 和 ***ex_catch5.ans***。

【样例 6】

见下发文件中的 ***ex_catch6.in*** 和 ***ex_catch6.ans***。

【子任务】

本题采用捆绑测试，你需要通过一个子任务的所有测试点才能得到子任务的分数。对于所有数据，保证 $2 \leq n \leq 10^5, 1 \leq m \leq 2 \times 10^5, 0 \leq k \leq n, 1 \leq t_i \leq 10^4$ 。

子任务编号	特殊性质	分值
1	$n \leq 15$	20
2	$k = 1$	30
3	$u_i \in \{1, n\}$ 或 $v_i \in \{1, n\}$	30
4	无	20