CSP-J 模拟

题目名称	切蛋糕	训练	生产线	货币系统
输入/输出文件名	cake.in/out	train.in/out	line.in/out	money.in/out
测试点时限	1 s	1 s	1 s	1 s
内存限制	256MB	256MB	256MB	256MB
分值	100	100	100	100
测试点/子任务个数	10	10	10	20
题目类型	传统型	传统型	传统型	传统型

注意事项

1. **需要建立子文件夹。**
2. 文件名（包括程序名和输入输出文件名）必须使用英文小写。
3. 结果比较方式为忽略行末空格、文末回车后的全文比较。
4. 程序可使用的栈空间大小与该题内存空间限制一致。
5. 编译选项：`-O2 -std=c++14 -lm`。

1. 切蛋糕（cake.cpp）

描述

Alice、Bob 和 Cindy 三个好朋友得到了一个圆形蛋糕，他们打算分享这个蛋糕。

三个人的需求量分别为 a, b, c，现在请你帮他们切蛋糕，规则如下：

1. 每次切蛋糕可以选择蛋糕的任意一条直径，并沿这条直径切一刀（注意切完后不会立刻将蛋糕分成两部分）。
2. 设你一共切了 n 刀，那么你将得到 $2n$ 个扇形的蛋糕（特别地，切了 0 刀被认为是有一个扇形，即整个圆形蛋糕），将这些蛋糕分配给 Alice，Bob 和 Cindy，要求每个扇形蛋糕只能完整地分给一个人。
3. 三人分到的蛋糕面积比需要为 $a:b:c$（不保证是最简比例，且如果 $a:b:c$ 中某个数为 0，表示那个人不吃蛋糕）。

为了完成这个任务，你至少需要切几刀？

格式

输入格式

本题单个测试点包含多组数据。

第一行包含一个整数 T，表示数据组数。

接下来 T 行，每行包含三个整数 a, b, c，表示三人的需求量。

输出格式

输出 T 行，第 i 行的输出表示第 i 组数据中你至少需要切蛋糕的次数。

样例

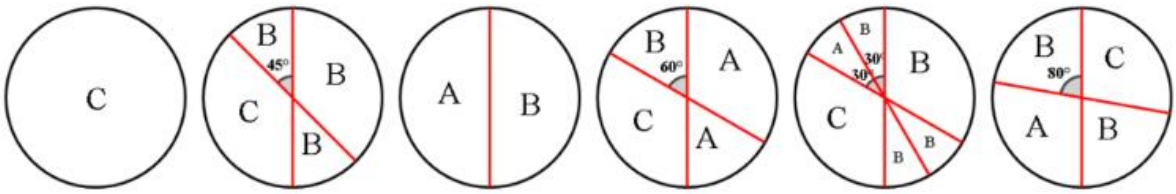
输入样例1

```
6
0 0 8
0 5 3
9 9 0
6 2 4
1 7 4
5 8 5
```

输出样例1

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

样例解释



数据范围

30% 的数据满足：$a = b = 0$。

60% 的数据满足：$a = 0$。

100% 的数据满足：$1 \le T \le 10^4$，$0 \le a, b, c \le 10^8$，保证 $a + b + c > 0$。

2. 训练（train.cpp）

描述

有一队 n 个士兵站成一排。如果把站的位置视为数轴，可以认为第 i 个士兵站在数轴的 a_i 位置上。

接下来，这 n 个士兵会完成 m 个指令。在执行指令的过程中，如果有士兵站立的位置超出了 $[-k, k]$ 这一范围，则该士兵出队。

一共有 3 种不同类型的指令：

`1 x`：全体士兵向数轴正方向移动 x 个单位；

`2 x`：全体士兵向数轴反方向移动 x 个单位；

`3`：查询队伍中还有几个士兵。

现在寻求你的帮助：对于每一个指令 `3`，给出相应的结果。

格式

输入格式

第一行三个正整数 n, m, k，含义如题所述。

第二行 n 个整数，描述每一个士兵的初始位置。

接下来 m 行，每行描述一个指令，指令的格式如题所述。

输出格式

对于每个指令 `3`，输出一行一个整数表示答案。

样例

输入样例1

```
3 4 3
-1 1 2
2 3
3
1 5
3
```

输出样例1

```
2
1
```

输入样例2,3/输出样例2,3

见下发文件。

样例解释

初始时，士兵的位置分别为 `[-1 1 2]`。

第一次操作后，士兵的位置变为 `[-4 -2 -1]`，第一个士兵出队，剩余士兵的位置为 `[-2 -1]`。

第二次操作是询问，答案为 2。

第三次操作后，士兵的位置变为 `[3 4]`，第二个士兵出队，剩余士兵的位置为 `[3]`。

第三次操作是询问，答案为 1。

数据范围

对于 30% 的数据：$1 \leq n, m \leq 5 \times 10^3$。

对于另外 20% 的数据：$1 \leq k \leq 500$。

对于 100% 的数据：$1 \leq n, m \leq 3 \times 10^5$，$1 \leq k, x \leq 2 \times 10^9$，$-k \leq a_i \leq k$，保证士兵所在的位置互不相同。

3. 生产线（line.cpp）

描述

工厂中有一条生产线可以用来制作着各类工艺品，每种工艺品都有其独特的价值。由于这条生产线很特殊，所以每次启动生产线时，制作出来的工艺品的价值总是**一段从小到大的连续整数**。

同时，工厂的仓库存储这些生产出来的工艺品时，存储模式类似于“栈”。也就是说，每制作出一个工艺品，就会将这个工艺品存入仓库，当需要从仓库中取出一个工艺品进行售卖时，会选择最后放入的那个工艺品。

现在，我们要进行 n 次操作，操作分为两种类型：

1. `1 l r`：启动生产线，依次制作出价值为 $l \sim r$ 的工艺品，并将这些工艺品依次放入仓库。
2. `2 k`：有一位客户要购买 k 件工艺品，此时会从仓库中依次取出 k 个工艺品来售卖。保证 k 小于仓库中的工艺品总数。

对于每个操作 2，请输出此次售卖的工艺品的总价值。

格式

输入格式

第一行有一个整数 n，表示操作的个数。

接下来 n 行描述一组询问。第一个整数 op 表示询问的种类，若为 1 则为操作 1，为 2 则为操作 2。

- 对于操作 1，接下来有两个整数 l, r，含义如题面所示。
- 对于操作 2，接下来有一个整数 k，含义如题面所示。

输出格式

对于每个操作 2，输出一行一个整数表示答案。

样例

输入样例1

```
6
1 1 14
2 5
1 14 19
1 1 9
2 8
2 10
```

输出样例1

```
60
44
124
```

输入样例2/输出样例2

见下发文件。

数据范围

- 对于前 30% 的数据，$n, l, r \leq 100$。
- 对于另外 20% 的数据，$l = r$。
- 对于另外 20% 的数据，$k \leq 10$。
- 对于 100% 的数据，$1 \leq n \leq 5 \times 10^5$，$0 \leq l \leq r \leq 10^6$，$1 \leq k \leq 10^{12}$。

4. 货币系统（money.cpp）

描述

在网友的国度中共有 n 种不同面额的货币，第 i 种货币的面额为 $a[i]$，你可以假设每一种货币都有无穷多张。为了方便，我们把货币种数为 n、面额数组为 $a[1..n]$ 的货币系统记作 (n,a)。

在一个完善的货币系统中，每一个非负整数的金额 x 都应该可以被表示出，即对每一个非负整数 x，都存在 n 个非负整数 $t[i]$ 满足 $a[i]\times t[i]$ 的和为 x。然而， 在网友的国度中，货币系统可能是不完善的，即可能存在金额 x 不能被该货币系统表示出。例如在货币系统 $n=3, a=[2,5,9]$ 中，金额 $1,3$ 就无法被表示出来。

两个货币系统 (n,a) 和 (m,b) 是等价的，当且仅当对于任意非负整数 x，它要么均可以被两个货币系统表出，要么不能被其中任何一个表出。

现在网友们打算简化一下货币系统。他们希望找到一个货币系统 (m,b)，满足 (m,b) 与原来的货币系统 (n,a) 等价，且 m 尽可能的小。他们希望你来协助完成这个艰巨的任务：找到最小的 m。

格式

输入格式

输入文件的第一行包含一个整数 T，表示数据的组数。

接下来按照如下格式分别给出 T 组数据。 每组数据的第一行包含一个正整数 n。接下来一行包含 n 个由空格隔开的正整数 $a[i]$。

输出格式

输出文件共有 T 行，对于每组数据，输出一行一个正整数，表示所有与 (n,a) 等价的货币系统 (m,b) 中，最小的 m。

样例

输入样例1

```
2
4
3 19 10 6
5
11 29 13 19 17
```

输出样例1

```
2
5
```

样例解释

在第一组数据中，货币系统 $(2,[3,10])$ 和给出的货币系统 (n,a) 等价，并可以验证不存在 $m<2$ 的等价的货币系统，因此答案为 2。

在第二组数据中，可以验证不存在 $m<n$ 的等价的货币系统，因此答案为 5。

数据范围

对于 100% 的数据，满足 $1\le T\le 20, n, a_i\ge 1$。

测试点	n	a_i
$1\sim 3$	$=2$	≤ 1000
$4\sim 6$	$=3$	≤ 1000
$7\sim 8$	$=4$	≤ 1000
$9\sim 10$	$=5$	≤ 1000
$11\sim 13$	≤ 13	≤ 16
$14\sim 16$	≤ 25	≤ 40
$17\sim 20$	≤ 100	≤ 25000