C5 - Statement

A 计算函数值 2024

问题描述

给定若干实数 x，计算下面的值：

$$f(x) = \sin(x) \cdot \ln(|x| + 1)$$

输入格式

不定行输入，每行一个实数 x ($-1000 \leq x \leq 1000$，输入时保留到小数点后两位)

保证不超过 1000000 组。

输出格式

不定行输出。对于每行输入，输出一行，一个实数 y，保留三位小数。

样例输入

```
99.45
0.00
3.14
```

样例输出

```
-4.068
0.000
0.002
```

Hint

建议使用 `math.h` 库中以下几个函数：

- `sin(x)` 返回弧度角 x 的正弦
- `fabs(x)` 返回浮点数 x 的绝对值
- `log(x)` 返回 x 的自然对数（底数为 e 的对数）

Author：Moca

B 小懒獭与函数

问题描述

小懒獭发现了一种特殊的函数 `f(x)`，它的定义如下：

$$f(x) = (\text{popcount}(x) + x^2) \mod 10000$$

其中，$\text{popcount}(x)$ 表示 x 的二进制表示中 1 的个数。

小懒獭对 $f(f(f(x)))$ 的结果非常感兴趣，请你帮忙计算它的值。

输入格式

不定行输入，每行一个自然数 x ($0 \leq x \leq 10^5$)

保证不超过 10000 组。

输出格式

不定行输出。对于每行输入，输出一行，表示 $f(f(f(x)))$ 对应的结果。

样例输入

```
1
2
3
```

样例输出

```
27
733
5381
```

Author：Moca

C 斐波那契数列 (easy version)

题目描述

如果每对大兔子每月能生殖一对小兔子，每对小兔子需要花一个月长成大兔子，那么在第一个月有 0 对大兔子，1 对小兔子的情况下，兔子每个月的数量会形成下面这个数列：

$$1, 1, 2, 3, 5, 8, 13, 21, \cdots$$

这个数列被称为斐波那契数列，又称兔子数列。

现在，改变第一个月的大兔子数量和小兔子数量，请你判断第 n 个月时总共有多少对兔子?

输入

第一行两个非负整数 a, b，表示在第一个月里，有 a 对大兔子，b 对小兔子，$0 \leq a \leq 10^9, 0 \leq b \leq 10^9$。

第二行一个正整数 t，表示数据组数，$1 \leq t \leq 10$。

接下来 t 行，每行一个正整数 n，表示询问第 n 个月时，兔子总数有多少对，$1 \leq n \leq 30$。

输出

对于每组数据，输出一行一个正整数，表示第 n 个月时的兔子总数（对）

输入样例 1

```
1 0
3
1
4
2
```

输出样例 1

```
1
5
2
```

输入样例 2

```
114514 1919810
3
1
2
10
```

输出样例 2

```
2034324
2148838
115781296
```

样例解释

对于第二组样例，第一个月大小兔子数量分别为 $114514, 1919810$，总数 2034324。第二个月大小兔子数量分别为 $2034324, 114514$，总数 2148838。

Hint

根据今天刚学的递归函数和全局变量来解决这个问题吧！

或者，你当然可以使用效率更高的方法。

D 卡皮巴拉的素数位置

题目描述

卡皮巴拉乐园中有 n 只卡皮巴拉，水豚饲养员正准备给他们喂食。第 i 只卡皮巴拉处于 p_i $(0 \leq i < n)$ 位置，如果 p_i 是素数，水豚饲养员则会准备 2^i 根玉米。所以最后总共会准备 $\sum 2^i$ $(i \in i|p_i是素数)$ 根玉米。水豚饲养员想知道需要准备的玉米数量是否是素数。

输入

第一行，1 个整数 n，表示卡皮巴拉数量，满足 $1 \leq n \leq 40$。

接下来 n 行，表示 n 个卡皮巴拉的位置 p_i，满足 $2 \leq p_i \leq 10^{10}$。

输出

对于每只卡皮巴拉，输出一行，如果其位置是素数，则输出 `Yes`，否则输出 `No`。

然后输出一行，表示总共需要准备的玉米数量（保证大于1）。

最后输出一行，表示玉米数量是否是素数，是素数，则输出 `Yes`，否则输出 `No`。

输入样例1

```
3
2
3
3
```

输出样例1

```
Yes
Yes
Yes
7
Yes
```

HINT

可以参考课件 **P5-函数** 中的素数判断方法。请注意本题数据范围。

Author: Capybara

E 摩卡与汉诺獭

题目描述

摩卡为水獭们设计了一个小游戏来帮助它们理解课上学习的递归思想。

水獭们排成了不同的队列，每只水獭都有一个不同的编号，现在共有三支队列 A、B、C：

- 初始时，所有水獭都站在队列 A 中，按照从小到大的编号从队首到队尾排列（编号从 1 到 n）。
- 摩卡的目标是将所有水獭从队列 A 移动到队列 C，但每次操作只能移动队首的水獭，且必须遵循以下规则：
  i. 编号较大的水獭不能在编号较小的水獭之前。
  ii. 将水獭移动到一个新的队列后，水獭会站在这个队列的队首，即之前来到这的所有水獭之前。

请帮助摩卡用最少的操作将所有水獭从队列 A 移动到队列 C。

输入格式

一个整数 n，表示水獭的数量，满足 $1 \leq n \leq 18$。

输出格式

对于 Moca 的每次操作，输出一行：

`Moca move otter {x} from queue {q1} to queue {q2}`

其中 x 是一个整数，表示水獭的编号，`q1`、`q2` 是一个字符，表示队列的编号（`A`、`B`、`C`）。

样例输入

```
3
```

样例输出

```
Moca move otter 1 from queue A to queue C
Moca move otter 2 from queue A to queue B
Moca move otter 1 from queue C to queue B
Moca move otter 3 from queue A to queue C
Moca move otter 1 from queue B to queue A
Moca move otter 2 from queue B to queue C
Moca move otter 1 from queue A to queue C
```

Hint

可以参考 PPT 例 5 - 15

Author：Moca

F 完美日期

题目描述

显而易见，每一个日期都可以表示为一个 8 位正整数，其中前 4 位表示年，第 $5,6$ 位表示月，第 $7,8$ 位表示日。例如， 1926 年 8 月 17 日可以表示为 19260817 。

对于一个日期，如果它对应的八位数是完全平方数，那么我们就认为这是一个完美日期。例如， 2024 年国庆节对应的 $20241001 = 4499^2$ ，所以这天就是一个完美日期。现在已知两个日期 y_1 年 m_1 月 d_1 日和 y_2 年 m_2 月 d_2 日，请你输出它们之间所有的完美日期（包含这两个日期）。

输入格式

多组数据输入，每组数据一行，两个 8 位正整数，分别表示对应的日期。数据保证两个日期均合法，都在 1600 年 1 月 1 日和 9999 年 12 月 31 日之间，并且第一个日期不在第二个日期之后。

保证输入的数据不超过 20 组。

输出格式

对于每组数据，输出一行一个整数，输出范围内完美日期的个数 n；

样例输入

```
20150101 20721231
20250101 20251231
```

样例输出

```
3
0
```

Hint

你可以实现下面两个函数完成功能的封装：

1. 这个 8 位数能否对应一个日期；
2. 这个 8 位数是否是完全平方数。

如果你的程序运行超时，请尝试优化（比如连续跳过一段不存在的日期）。提示：

1. 每个月都不存在 32 号到 99 号；
2. 每一年都只有 12 个月，不存在 13 月到 99 月。

`G` shtog 的电子日历

题目描述

shtog 想开发一款电子日历系统。在开发过程中，他需要实现这样的一个功能：给出一个年份和月份，输出该年该月的日期表。可是贪玩的 shtog 打算国庆节出去度假，于是便将这个需求托付给了你，请你编写程序完成这一功能。

输入

只有一行，包含以一个空格隔开的两个整数：年份y（$1900 \leq y \leq 2099$），月份m（$1 \leq m \leq 12$）

输出

输出为月历表。月历表第一行为星期表头，如下所示：

```
Sun Mon Tue Wed Thu Fri Sat
```

剩下各行依次是当月各天的日期，从1日开始到31日（30日或29日或28日）。每个日期应与日期表头保持右对齐，空白部分均用空格填充

输入样例

```
2020 2
```

输出样例

```
Sun Mon Tue Wed Thu Fri Sat
                        1
  2   3   4   5   6   7   8
  9  10  11  12  13  14  15
 16  17  18  19  20  21  22
 23  24  25  26  27  28  29
```

HINT

注意考虑闰年2月份有29天，平年有28天，一个年份y是闰年等价于：**y能被4整除且不能被100整除**或者**y能被400整除**

可以参考，1900年的1月1日为周一，据此循环推算某天的周几。

或者可以使用`Zeller`公式来直接计算某年某月某日是星期几，具体可参考C4上机赛D题以及C5对应PPT课件例题“万年历”。

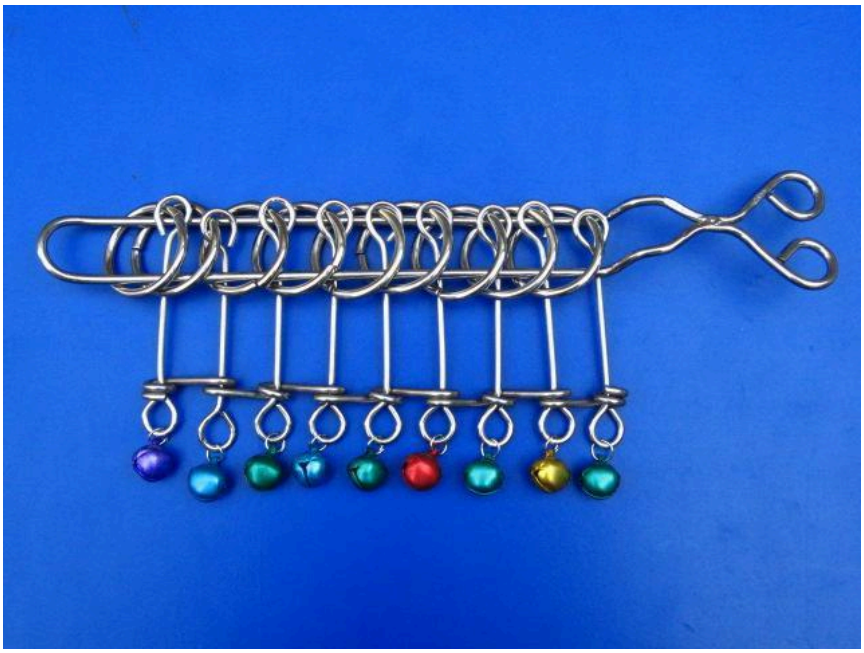
AUTHOR：shtog

H 九连环

第一名（计算本题罚时）做出本题的课程内24级非免修同学，经验证为独立完成后可以获得出题人赠送的九连环一个。

题目描述

「James Eugene Raynor」送给「Zeratul」了一个九连环，但「Zeratul」始终玩不明白九连环，请你帮帮他。



如图，九连环由一个架子和九个环组成，可以进行如下操作：

- 第一个环可以从架上取下或放到架上；
- 第 $k(k>1)$ 个环可以被取下（放上），如果第 $k-1$ 个环是它前面唯一一个在架上的环。

那么对于环数更少或更多的 n 连环，游戏方式是类似的。「Zeratul」想知道，最快将一个所有环都在架上的 n 连环拆下来需要怎么做，请你帮帮他。

为了帮助没有玩过九连环的同学们理解这道题，现将此题用数学语言重新翻译（二进制表示的最低位称为第 0 位，次低位称为第 1 位，以此类推）：

输入一个正整数 n，你将获得一个所有位全是 1 的 n 位二进制数 $status$，你可以对其进行如下操作之一：

- 将 $status$ 二进制表示的最低位取反；
- 将 $status$ 二进制表示的第 $k(k>0)$ 位翻转，如果 $status$ 的二进制表示的最后一个 1 在第 $k-1$ 位。

现在请你求出使用最少次操作将 $status$ 的值变为 0 的方法。

输入格式

一个正整数 $n(1\le n\le 15)$，表示环数。

输出格式

共 p 行，其中 p 为最小步数。

每行一个自然数 $status$，表示执行完这一步操作后连环的状态（即 $status$ 的十进制值）。 $status$ 的 n 位二进制表示中，对于每一位而言， 1 表示该环在架上， 0 表示该环在架下。

显然，最后一个自然数应该是 0 。

输入样例

```
3
```

输出样例

```
6
2
3
1
0
```

样例解释

第一步，将第 1 个环取下，$status = (110)_2 = 6$；

第二步，将第 3 个环取下，$status = (010)_2 = 2$；

第三步，将第 1 个环放上，$status = (011)_2 = 3$；

第四步，将第 2 个环取下，$status = (001)_2 = 1$；

第五步，将第 1 个环取下，$status = (000)_2 = 0$。

Hint

显然，将一个被拆下来的连环全部放上架子的操作和将其从架子上取下的操作是反向的。

想要取下 n 连环，可以先取下 $n-2$ 连环，再取下第 n 个环，再放上 $n-2$ 连环，再取下 $n-1$ 连环。可以证明，这是最快的方法。

Author: Protoss

I 摩卡与艺术家水獭

题目描述

著名的艺术家水獭正在创作一组独特的艺术雕塑柱，这些柱子将用于即将举行的艺术展览。

他可以使用三种不同的木块材料：**浅色薄木块**、**深色薄木块**和**深色厚木块**。其中，薄木块的厚度为 1 cm，而厚木块的厚度为 k cm。这些木块可以堆叠成不同高度的展示柱，放置在透明的展示柜中，让来参观的水獭们欣赏。

为了使雕塑作品符合艺术家的高标准，水獭必须遵循以下创作规则：

1. 每根雕塑柱至少由一个木块组成。
2. 每根雕塑柱的总高度不超过 l cm。
3. 雕塑柱的顶部和底部必须是深色木块。
4. 木块的堆叠必须是**深浅交替**的，也就是说浅色木块上方必须放置深色木块，深色木块上方则必须放置浅色木块。

为了当给定最大高度 l 和厚木块厚度 k 时，他总共可以创作出多少种符合要求的雕塑柱组合，水獭向他的好朋友，会编程的 Moca 寻求帮助。

输入格式

两个整数 l 和 k，含义如题目所示，其中 $1 \leq l \leq 100$，$2 \leq k \leq 100$

输出格式

输出一个整数，表示合法的方案数。

样例输入

```
5 3
```

样例输出

```
6
```

说明

在该样例中，当最大高度为 5 cm，厚木块厚度为 3 cm 时，共有 6 种符合条件的雕塑柱组合（下面用红色数字表示白色木板）：

1. $[1, \color{red}{1}, 1, \color{red}{1}, 1]$
2. $[1, \color{red}{1}, 1]$
3. $[1]$
4. $[1, \color{red}{1}, 3]$
5. $[3, \color{red}{1}, 1]$
6. $[3]$

J 相信后人的智慧

描述

《相信后人的智慧》是一款策略游戏，它由开始阶段和行动阶段以及结算阶段组成。

在开始阶段中，我们会随机生成一个自然数 m 作为行动阶段的回合数量，以及随机生成一个由大于 1 的正整数组成的数列 $a_1, a_2, \cdots, a_m$ 分别作为行动阶段各回合的幸运数字，注意 m 和 $a_1, a_2, \cdots, a_m$ 被称为基础设定；然后，你会获得 n 个尚未解决的历史遗留问题，并且此时你的信任值是 0。

在行动阶段中，对于不大于 m 的正整数 k，在第 k 个回合中，你必须将当前尚未解决的历史遗留问题划分为 a_k 份，其中每一份都由自然数个当前尚未解决的历史遗留问题组成，然后你必须解决除了最少的那一份以外的全部的当前尚未解决的历史遗留问题，其中每解决一个历史遗留问题会使你的信任值增加 e^k 。

在结算阶段中，每个当前尚未解决的历史遗留问题会使你的信任值增加 e^{m^2+1}；最后，信任值最高的玩家会获得胜利。

输入

一个不大于 10^9 的自然数 n，表示你在开始阶段获得的尚未解决的历史遗留问题的数量。

输出

一个自然数，表示有多少种基础设定使得在最优策略下你在结算阶段有不少于一个尚未解决的历史遗留问题。

样例

输入 1

```
4
```

输出 1

```
5
```

说明 1

只有如下 5 个基础设定使得在最优策略下结算阶段有不少于一个尚未解决的历史遗留问题：

- $m = 0$；在最优策略下，结算阶段有 4 个尚未解决的历史遗留问题从而信任值增加 $4\mathrm{e}^1$，最后信任值为 $4\mathrm{e}^1 \approx 10.873$ 。
- $m = 1$，$a_1 = 2$；在最优策略下，在第一回合中 $4 = 2 + 2$ 从而信任值增加 $2\mathrm{e}^1$，结算阶段有 2 个尚未解决的历史遗留问题从而信任值增加 $2\mathrm{e}^2$，最后信任值为 $2\mathrm{e}^1 + 2\mathrm{e}^2 \approx 20.215$ 。

- $m = 1$，$a_1 = 3$；在最优策略下，在第一回合中 $4 = 1 + 1 + 2$ 从而信任值增加 $3\mathrm{e}^1$，结算阶段有 1 个尚未解决的历史遗留问题从而信任值增加 $1\mathrm{e}^2$，最后信任值为 $3\mathrm{e}^1 + 1\mathrm{e}^2 \approx 15.544$。
- $m = 1$，$a_1 = 4$；在最优策略下，在第一回合中 $4 = 1 + 1 + 1 + 1$ 从而信任值增加 $3\mathrm{e}^1$，结算阶段有 1 个尚未解决的历史遗留问题从而信任值增加 $1\mathrm{e}^2$，最后信任值为 $3\mathrm{e}^1 + 1\mathrm{e}^2 \approx 15.544$。
- $m = 2$，$a_1 = 2, a_2 = 2$；在最优策略下，在第一回合中 $4 = 2 + 2$ 从而信任值增加 $2\mathrm{e}^1$，在第二回合中 $2 = 1 + 1$ 从而信任值增加 $1\mathrm{e}^2$，结算阶段有 1 个尚未解决的历史遗留问题从而信任值增加 $1\mathrm{e}^5$，最后信任值为 $2\mathrm{e}^1 + 1\mathrm{e}^2 + 1\mathrm{e}^5 \approx 161.239$。

其它基础设定不可以使得在最优策略下结算阶段有不少于一个尚未解决的历史遗留问题，例如：

- $m = 1$，$a_1 = 5$；在最优策略下，在第一回合中 $4 = 0 + 1 + 1 + 1 + 1$ 从而信任值增加 $4\mathrm{e}^1$，结算阶段没有尚未解决的历史遗留问题从而信任值增加 $0\mathrm{e}^2$，最后信任值为 $4\mathrm{e}^1 \approx 10.873$。
- $m = 3$，$a_1 = 3, a_2 = 3, a_3 = 3$；在最优策略下，在第一回合中 $4 = 1 + 1 + 2$ 从而信任值增加 $3\mathrm{e}^1$，在第二回合中 $1 = 0 + 0 + 1$ 从而信任值增加 $2\mathrm{e}^2$，在第三回合中 $0 = 0 + 0 + 0$ 从而信任值增加 $0\mathrm{e}^3$，结算阶段没有尚未解决的历史遗留问题从而信任值增加 $0\mathrm{e}^{10}$，最后信任值为 $3\mathrm{e}^1 + 1\mathrm{e}^2 + 0\mathrm{e}^3 + 0\mathrm{e}^{10} \approx 15.544$。

输入 2

```
10
```

输出 2

```
19
```

输入 3

```
1000
```

输出 3

```
48614
```

输入 4

```
1000000000
```

输出 4

```
1149582312706970
```

提示

注意时间限制。

如果你想要通过全部的测试数据，那么你的程序在 duck.ac 中的运行时间应该不超过 100 毫秒。