实验舱
青少年编程
走近科学 走进名校

蛟龙四班

算法复杂度分析、枚举

Mas

Part-1

C++语法回顾

数据类型	定义标识符	占字节数	数值范围	表示范围
短整型	short [int]	2(16位)	-32768～32767	-2^{15}～$2^{15}-1$
整型	[long] int	4(32位)	-2147483648～2147483647	-2^{31}～$2^{31}-1$
长整型	long [int]	4(32位)	-2147483648～2147483647	-2^{31}～$2^{31}-1$
超长整型	long long [int]	8(64位)	-9223372036854775808～9223372036854775807	-2^{63}～$2^{63}-1$
无符号整型	unsigned [int]	2(16位)	0～65535	0～$2^{16}-1$
无符号短整型	unsigned short [int]	2(16位)	0～65535	0～$2^{16}-1$
无符号长整型	unsigned long [int]	4(32位)	0～4294967295	0～$2^{32}-1$
无符号超长整型	unsigned long long	8(64位)	0～18446744073709551615	0～$2^{64}-1$

数据类型	定义标识符	数值范围	占字节数	有效位数
单精度实型	float	-3.4E-38～3.4E+38	4(32位)	6～7位
双精度实型	double	-1.7E+308～1.7E+308	8(64位)	15～16位
长双精度实型	long double	-3.4E+4932～1.1E+4932	16(128位)	18～19位
布尔变量	bool	真true或假false之一	1(8位)	

IO流

重定向

在一般情况下，我们都会从控制台（中进行输入stdin和输出stdout，通过文件重定向可以从文件进行输入和输出

- C/CPP
  - 从Xxx.in文件中读入：

    ```
    freopen("xxx.in", "r", stdin);
    ```

  - 向xxx.out文件输出：

    ```
    freopen("xxx.out", "w", stdout);
    ```

快速读入

- 一般情况下，C/CPP读取数据都是以字节流的形式读取，速度较慢
  - C++快读(字符流读入)

```
int read()
{
    int x = 0, f = 1;
    char ch = getchar();
    while (ch < '0' || ch > '9')
    {
        if (ch == '-')
            f = -1;
        ch = getchar();
    }
    while (ch >= '0' && ch <= '9')
    {
        x = (x << 1) + (x << 3) + (ch ^ 48);
        ch = getchar();
    }
    return x * f;
}
```

Part-2

算法复杂度

算法复杂度

- **TLE：**
  - Time Limit Exceeded 程序运行超过了时间限制
- **MLE：**
  - Memory Limit Exceeded 程序运行时使用了超过内存限制的空间

- 算法的复杂度是用来衡量算法好坏的一个指标，常用时间复杂度和空间复杂度，它们一般都是关于输入数据量的函数，例如O(n),O(n^2),O(log n)

时间复杂度简单分析

- 时间复杂度只关心算法中最耗时的部分,舍去常数部分,通常用简单的函数O来表示

- 这段代码复杂度为O(n)

```
for(i=1;i<=n;i++)
    ……(内部没有循环)
```

- 这段代码复杂度为O(n*m)

```
for(i=1;i<=n;i++)
   for(j=1;j<=m;j++)
       ……(内部没有循环)
```

- 这段代码复杂度为O(logn)

```
while(n)
    n /= 2;
```

时间复杂度简单分析

- 时间限制：1000ms能干些什么？
- 数量级小于等于5e8基本上可以认为在1000ms不会超时
- 如果数量级5e8 ~ 1e9 中，有一定可能会超时

时间复杂度	1000ms处理数据量
O(n)	<=5e8
O(n^2)	<=10000
O(n^3)	<= 500
O(n!)	<= 11
O(2^n)	<=25
O(nlogn)	< 1e6
O(logn)	<=1e16
1	\

空间复杂度

- 空间限制：256MB能干些什么？
- 256MB = 2^8MB = 2^18KB = 2^28 B
- 4B = 1个int (32位)
- 256MB = 2^26个int = 67,108,864 个 int ≈ $6*10^7$个int
- 结论：256MB的内存空间最多大约能开的int数组长度为**60000000**

在函数内声明的基本数据类型都分配在栈上，在函数外声明的数据类型都分配在堆上
主流OJ对于栈内存大小限制为128MB，如果需要分配较多空间，建议写在函数外

Part-3

枚举

枚举

枚举就是根据提出的问题，一一列出该问题的所有可能的解。

在逐一列出的过程中，检验每个可能解是否是问题的真正解，如果是就采纳这个解，如果不是就继续判断下一个。

枚举法一般比较直观，容易理解，但由于要检查所有的可能解，因此**运行效率较低**。

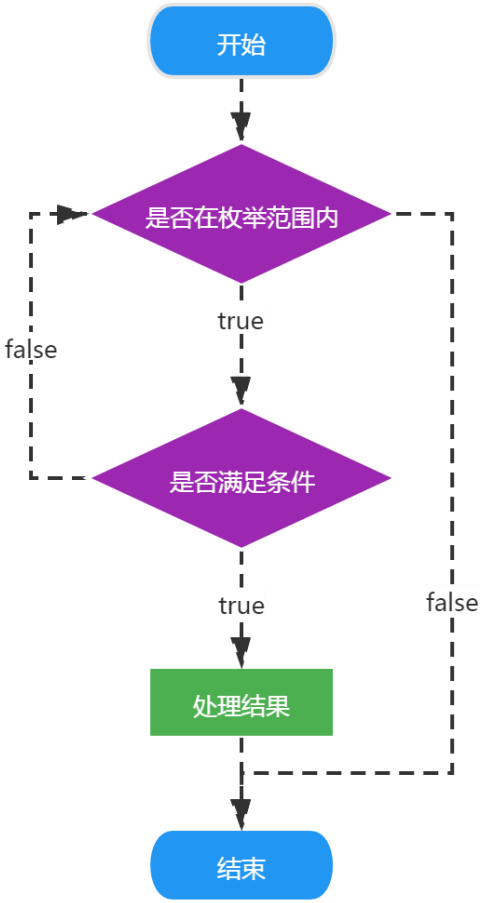
能够用枚举法解决的题目往往是最简单的一类题目。

这种题目具有以下特点：

1. 解枚举范围是有穷的
2. 检验条件是确定的

枚举题解题解题套路

- **确定枚举范围**
- **写出条件判断表达式**



#87 合数的因子

【题目描述】

输入一个正整数，输出所有的因子。

【输入格式】

一个整数。

【输出格式】

若干个整数，每个整数后面一个空格，最后不换行。

【输入样例】

```
12
```

【输出样例】

```
1 2 3 4 6 12
```

```cpp
#include <bits/stdc++.h>
using namespace std;
int main()
{
    int n;
    cin >> n;
    for (int i = 1; i <= n; i++)
        if (n % i == 0)
            cout << i << " ";
    return 0;
}
```

范围: 小于1的所有自然数，也就是1 ~ n

条件: 能把n 除尽的数，也就是n % i == 0

#104 知识竞赛

【题目描述】

某次知识竞赛共有 25 题，评分标准如下：答对一题得 8 分，答错 1 题倒扣 5 分，不答题不得分也不扣分。小明答题得分是 60 分，问小明答对、答错、不答对各有多少题？

【输入格式】

输入二个整数 n ， m ，分别是题目数量和得分分数。

【输出格式】

输出三个整数，分别是答对、答错、不答的题目数。

【输入样例】

```
20 55
```

【输出样例】

```
10 5 5
```

- 如何枚举?
- 需要枚举那些变量?
- 时间复杂度是多少?

知识竞赛

枚举变量	含义	范围
i	答对题目数量	0~n
j	答错题目数量	0~n
k	不答题目数量	0~n

直接枚举 i、j、k

条件：

```
i+j+k == n && 8*i-5*j == m
```

时间复杂度O(n^3)

变量	含义	范围
i	答对题目数量	0~n
j	答错题目数量	0~n-i
k	不答题目数量	0~n-i-j

枚举 i、j、k，去掉无意义的情况

条件：

```
i+j+k == n && 8*i-5*j == m
```

时间复杂度O(n^3)减少了无意义的枚举

变量	含义	范围
i	答对题目数量	0~n
j	答错题目数量	0~n-i

枚举 i、j，直接根据条件计算k

条件：

```
8*i-5*j == m
```

时间复杂度O(n^2)

#901 铺地毯

题目描述

为了准备一个独特的颁奖典礼，组织者在会场的一片矩形区域（可看做是平面直角坐标系的第一象限）铺上一些矩形地毯。一共有 n 张地毯，编号从 1 到 n。现在将这些地毯按照编号从小到大的顺序平行于坐标轴先后铺设，后铺的地毯覆盖在前面已经铺好的地毯之上。

地毯铺设完成后，组织者想知道覆盖地面某个点的最上面的那张地毯的编号。注意：在矩形地毯边界和四个顶点上的点也算被地毯覆盖。

输入输出格式

输入格式：

输入共 $n+2$ 行

第一行，一个整数 n，表示总共有 n 张地毯

接下来的 n 行中，第 $i+1$ 行表示编号 i 的地毯的信息，包含四个正整数 a,b,g,k，每两个整数之间用一个空格隔开，分别表示铺设地毯的左下角的坐标 (a,b) 以及地毯在 x 轴和 y 轴方向的长度

第 $n+2$ 行包含两个正整数 x 和 y，表示所求的地面的点的坐标 (x,y)

输出格式：

输出共 1 行，一个整数，表示所求的地毯的编号；若此处没有被地毯覆盖则输出 -1

#874 四方分解

【描述】

有一个定理：对于任意一个正整数 n，可以分解为不超过 4 个自然数的平方和。

比如

$$25 = 1^2 + 2^2 + 2^2 + 4^2 = 3^2 + 4^2 = 4^2 + 3^2 = 5^2$$

显然有 4 种方法，但是 $3^2 + 4^2$，$4^2 + 3^2$ 算 1 种

所以 25 能分解的方法共有 3 种

【输入格式】

一个正整数 $\mathrm{n}\ (1 \leq n \leq 32768)$

【输出格式】

一个整数，代表分解方案数

【样例输入】

```
25
```

【样例输出】

```
3
```

直接枚举四个数：时间复杂度O(n^4)

可以更高效吗?

#1025 最大公约数和最小公倍数问题

题目描述

输入二个正整数 x_0，y_0（$2 \le x_0 \le 1000000$，$2 \le y_0 \le 1000000$），求出满足下列条件的 P，Q 的个数。

条件：

1. P，Q 是正整数；
2. 要求 P，Q 以 x_0 为最大公约数，以 y_0 为最小公倍数。

试求：

满足条件的所有可能的两个正整数的个数。

输入

每个测试文件只包含一组测试数据，每组两个正整数 x_0 和 y_0 $(2 \le x_0 < 1000000，2 \le y_0 \le 1000000)$。

输出

对于每组输入数据，输出满足条件的所有可能的两个正整数的个数。

样例输入

```
3 60
```

样例输出

```
4
```

```
for (int p = x; p <= y; p++)
    for (int q = x; q <= y; q++)
    {
        int g = __gcd(p, q), l = p / g * q;
        if (g == x && l == y)
            ans++;
    }
```

枚举 p,q，p,q 范围为 x ~ y,分别求出最大公约数和最小公倍数

令 N = y – x,T = min(p,q)

求最大公约数时间复杂度为O(logT)

总时间复杂度近似为 O(N*N* logT)

最大公约数和最小公倍数问题

性质

p*q = gcd(p,q) * lcm(p,q)

p*q = x*y

枚举 p， 枚举范围为 x ~ y , 步长为 x , 根据性质直接求出 q

令 N = (y - x) / x, T = min(p,q)

时间复杂度为 O(N*logT)

```
for (int q, p = x; p <= y; p += x)
{
    if (x * y % p != 0)
        continue;
    q = x * y / p;
    if (__gcd(p, q) == x)
        ans++;
}
```

#1090 两数之和

题目描述

给定一个长度为 $n(n \leq 50000)$ 的有序数组，请你求出两个元素之和为 sum 的下标

输入描述

第一行两个整数 n、sum

接下来一行 n 个整数

输出描述

两个整数，表示第一个元素的下标和第二个元素的下标(下标从 0 开始)

如果无解则输出 $ERROR$

输入样例

```
6 30
1 3 5 9 15 25
```

输出样例

```
2 5
```

两重循环枚举，一层确定a[i],另一层查找k-a[i]是否存在，总时间复杂度O(n*n)

借助数组有序，查找可用二分查找(O(logn))。总时间复杂度O(n*logn)

是否有更高效的解法?

#1603 整除的数(加强版)

题目描述

$1、2、3.....n$ 这 $n(0 < n \leq 10^{18})$ 个数中有多少个数可以被正整数 b 整除。

输入

第一行包含一个整数 $T(1 \leq T \leq 10^5)$

每组数据占一行，每行给出两个正整数 $n(0 < n \leq 10^{18})$ 、 $b(1 \leq b \leq 10^{18})$ 。

输出

输出每组数据相应的结果。

样例输入

```
3
2 1
5 3
10 4
```

样例输出

```
2
1
2
```

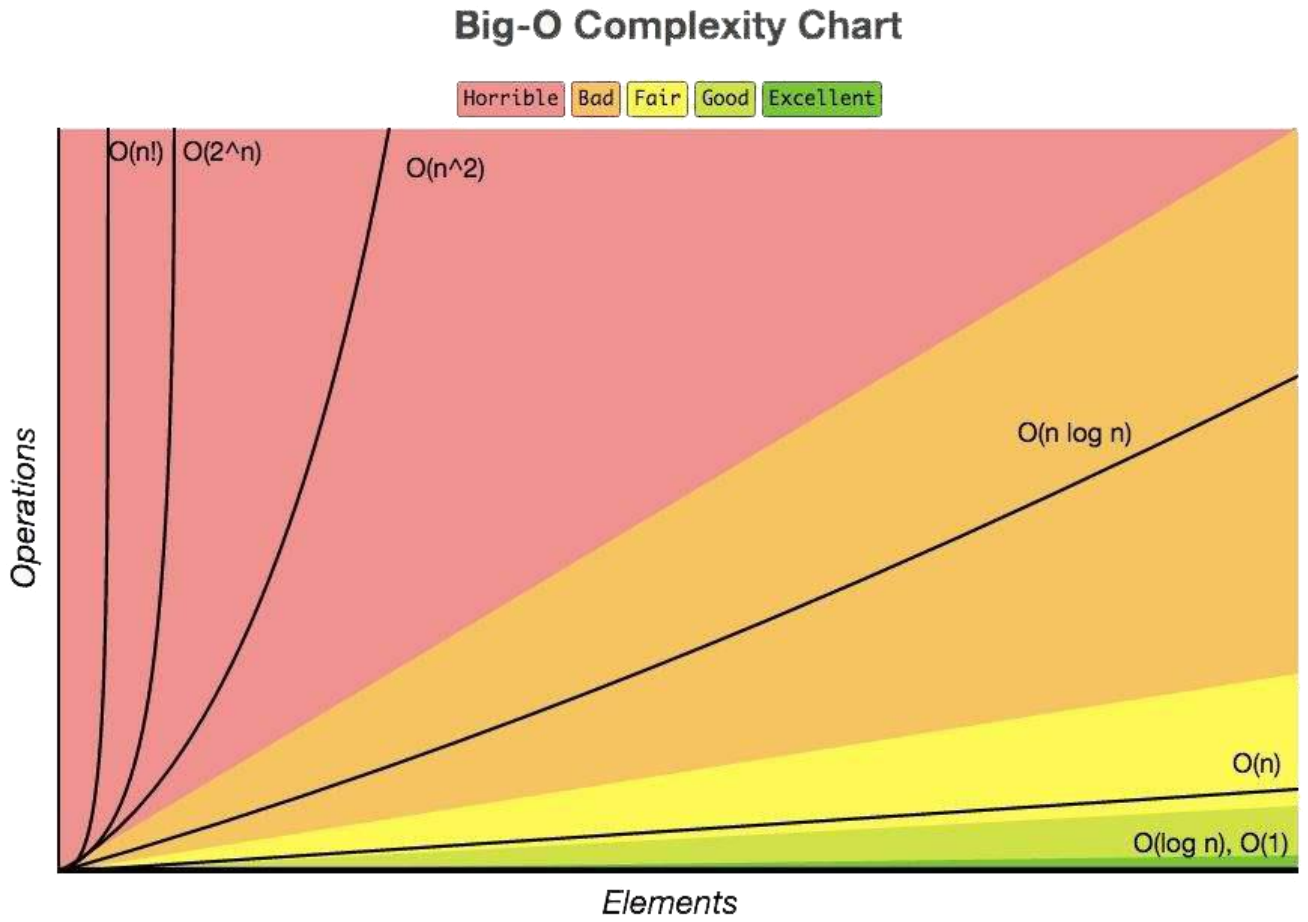
对于每一次询问，从1~n枚举能被b整除的数

总时间复杂度O(T*n)

是否有更高效的解法?

枚举优化的方法

- 尽可能减少循环的次数
- 尽可能减少枚举的变量(减少循环的层数)
- 用空间换时间
- 对数据进行预处理



谢谢观看