朋辈辅导直播

程序设计之
时间复杂度与空间复杂度简介

主讲人：佟振宇

北航学业与发展支持中心朋辈辅导师
宇航学院2021级本科生
2022程序设计基础课程助教

直播时间：2022年9月24日 14:00-15:00

北京航空航天大学 BEIHANG UNIVERSITY

学业与发展支持中心

内容简介

时间复杂度

算法的时间复杂度是一个函数，它定性描述该算法的运行时间

空间复杂度

空间复杂度是对一个算法在运行过程中临时占用存储空间大小的量度

目录页 CONTENTS PAGE

Part 1: 时间复杂度

引入

问题引入

为什么要了解时间复杂度?

在之前的学习，比如E1-J中，有不少同学的代码出现了TLE的结果，因而无法得到满分。随着大家学习的进步，TLE的出现频率可能会越来越高，甚至和WA一起成为最常见的Unaccepted类型。

Time Limit Exceed (TLE)

为什么会TLE？通俗的讲，就是程序需要得到答案的运行时间超过了题目的时间限制，从而在达到时间限制时，没有输出一个答案。这样，就会因为TLE而不通过本题。

如何克服TLE的问题?

了解什么是时间复杂度，掌握时间复杂度更优的算法。

根据数据范围选择具有合适时间复杂度的算法。

概念简介

在计算机科学中，**时间复杂性**，又称**时间复杂度**，算法的**时间复杂度**是一个函数，它定性描述该算法的运行时间。这是一个代表算法输入值的字符串的长度的函数。时间复杂度常用大O符号表述，不包括这个函数的低阶项和首项系数。使用这种方式时，时间复杂度可被称为是渐近的，亦即考察输入值大小趋近无穷时的情况。

感到迷茫？看不懂？

简单介绍：时间复杂度是一个函数，一般与 n，m 等与问题有关的变量组成，用来表示在 n=某数 的情况下程序大致进行的运算次数。

大O符号与大家在数学分析中学到的意义类似。常见的时间复杂度有：

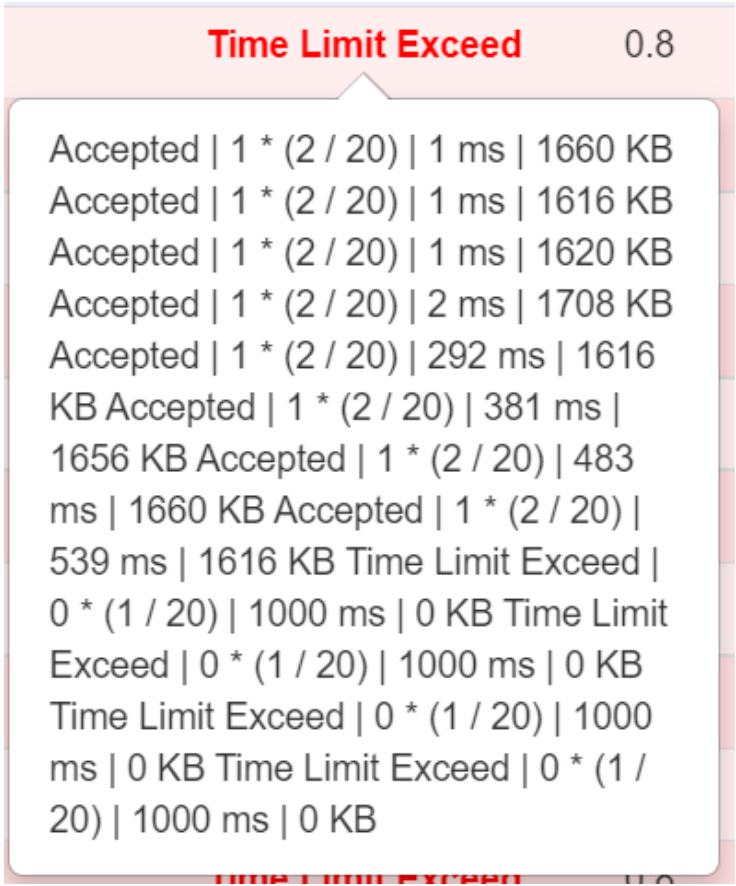
$$O(n), O(n^2), O(nlogn), O(\sqrt{n}), \dots$$

我们可以用实际的数字，从而得到一个程序大致的运算次数。
比如 n=10000 带入，时间复杂度为$O(n)$的程序运算次数大致为10000，而时间复杂度为$O(n^2)$的程序运算次数大致为100000000.

概念简介

一般而言，计算机在1s内能进行的运算次数大约为10的8次方，而OJ平台上对程序运行时间的限制为1000ms，因此通过计算时间复杂度我们能够估计程序的运算次数，进而判断程序是否有可能超时（TLE）。

如果一道题输入数据n范围是$n \leq 10^7$，则这道题的程序只能使用时间复杂度为$O(n)$的算法，而不能选用时间复杂度为$O(n^2)$的算法。



如何计算

1 找到循环语句

由于循环以外的语句运算次数一般都很小，相比于循环中的语句可以忽略不计，因此一般我们只考虑循环次数最多的循环。

2 判断循环次数

循环次数一般根据循环的条件以及循环的嵌套层数来判断，
如右图几种最简单的情况。
有些一眼看不出来的情况需要你理解这一段代码是如何运行的，从而判断其中语句的执行次数。
还有一些常见算法的时间复杂度可以当作结论记住。如二分法、辗转相除法、冒泡排序、快速排序等等。

3 计算时间复杂度

一般而言，相互独立的循环语句（无嵌套关系）中循环次数最高的就是程序的时间复杂度。

```
for(i=0; i<n; i++){
    //some codes
}
```

$O(n)$

```
for(i=0; i<n; i++){
    for(j=0; j<m; j++){
        for(k=0; k<t; k++){
            //some codes
        }
    }
}
```

$O(nmt)$

若n=m=t则为$O(n^3)$

```
for(i=1; i*i<=n; i++){
    //some codes
}
```

$O(\sqrt{n})$

```
//辗转相除法
while(b){
    int t=a%b;
    a=b;
    b=t;
}
```

$O(logn)$

例题分析

C 蛋糕店打工

时间限制：1000ms 内存限制：65536kb

通过率：807/1040 (77.60%) 正确率：807/2911 (27.72%)

题目描述

小羊学姐在蛋糕店找了一份兼职，她每天的具体工作是：蛋糕店一共有 n 种品牌的蛋糕（每种品牌都有无限盒），品牌按照固定顺序排列，且第 i 种品牌的蛋糕每盒中含有的蛋糕块数规格是： a_i 块/盒。小羊每天要接待 m 位客户，每位客户有且仅能买两盒蛋糕，客户会告诉小羊他想要的两盒蛋糕分别是哪两个品牌（交代品牌的序号），小羊需要在结账时告诉客户他买的两盒蛋糕里面一共有多少块？请你帮助小羊学姐快速计算。

输入

第一行有两个正整数，分别为 n 和 m

第二行有 n 个正整数，分别是 a_1 , a_2 , a_3 ,..., a_n

接下来有 m 行，每行两个正整数，分代表两盒蛋糕的品牌序号

（所有同一行输入的正整数之间都用空格隔开）

输出

一共 m 行输出，每行一个正整数

C2-C

读入n个数，存入数组中。

查询m次，每次讲两个数字相加并输出

因此时间复杂度为$O(n+m)$.

D 星幽测试

时间限制：1000ms 内存限制：65536kb

通过率：506/829 (61.04%) 正确率：506/2036 (24.85%)

题目描述

七海正在帮学姐测试新生的星幽能力值。为了获知当年的整体情况，根据老师要求，她需要计算出所有 N 名新生的"整均值"，"非方差"，能力值处于"整均值"以上（不含等于）的人数以及按照输入顺序前 K 人能力的"整均值"。

输入

共两行，第一行两个数，为 N 和 K ，意义如上。

第二行 N 个数，代表每位同学的能力值 x_i。

附注：整均值为先计算所有人总和，再将总和除以人数，结果向上取整的值。非方差指的是，先计算所有人和整均值之差的平方值的总和，再除以人数，结果向下取整的值。

输出

输出四个数，代表所有新生的整均值，非方差，能力值处于均值以上的人数以及按照输入顺序前 K 人能力的整均值。

C2-D

N个数据，K<=N，虽然做法中先用一个循环算出了整均值，又用一个循环算出了非方差，总运算次数可能为几倍的n，但是时间复杂度作为对运算次数的估计，我们一般忽略常系数，因此时间复杂度为$O(n)$.

例题分析

H 最大公约数x3

时间限制：1000ms 内存限制：65536kb

通过率：344/702 (49.00%) 正确率：344/1930 (17.82%)

题目描述

求出三个数字 a, b, c 的最大公约数

输入

第一行三个数字，如题目描述所述

输出

三个数字的最大公约数

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>

int main(){
        int a,b,c;
        scanf("%d%d%d",&a,&b,&c);
        int gcd=a;
        if(gcd<b)gcd=b;
        if(gcd<c)gcd=c;
        for(;gcd>=1;gcd--){
                if(a%gcd==0&&b%gcd==0&&c%gcd==0)break;
        }
        printf("%d",gcd);
        return 0;
}
```

C1-H

设 a, b, c 不超过正整数M，则右图做法中，循环要进行M-gcd(a, b, c)次，一般情况下gcd(a, b, c)与M相比过于小，因此可估计循环次数为M次，因此该算法时间复杂度为$O(M)$。

原题中，a, b, c的范围是0 ≤ a, b, c ≤ 1000，即M为1000。因此$O(M)$的算法不会超时。如果题目数据范围改为 a, b, c 在 int 范围内，即M为$2^{31}-1$，数量级是10的9次方，则$O(M)$的算法肯定会超时，需要时间复杂度更优的算法。

H 最大公约数x3

时间限制：1000ms 内存限制：65536kb

通过率：344/702 (49.00%) 正确率：344/1930 (17.82%)

题目描述

求出三个数字 a, b, c 的最大公约数

输入

第一行三个数字，如题目描述所述

输出

三个数字的最大公约数

C1-H

右图做法中，运用了两次辗转相除法得出了答案，辗转相除法的时间复杂度为$O(logn)$，即使数据范围是 a, b, c 在int范围内，甚至long long范围内，$M = 2^{63} - 1$，$log(M) = 63$，不可能超时。

```c
#include <stdio.h>

int main(){
        int a, b, c;
        scanf("%d%d%d", &a, &b, &c);
        while(a){
                int d=b%a;
                b=a;
                a=d;
        }
        while(c){
                int d=b%c;
                b=c;
                c=d;
        }
        printf("%d", b);
        return 0;
}
```

优化算法

J 回文日期

时间限制：1000ms 内存限制：65536kb

通过率：97/162 (59.88%) 正确率：97/569 (17.05%)

题目描述

Uanu在学滋芬时被突然打开的传送门带到了蓝星，但当Uanu想回去时传送门却已经关闭了。探索过程中发现这个星球的一切都是回文（对称）的，所以Uanu猜测只有是回文日期时传送门才会打开。

已知该星球的年月日的周期长度与地球相同，使用八位数字表示一个日期：前 4 位代表年，接下来 2 位代表月，最后 2 位代表日。如：**20200202**、**20211202** 都是对称（回文）日期，而**20220925** 不是回文日期。

由于Uanu的滋芬作业还没完成，他会在最近一次传送门打开时回家。请你帮Uanu计算一下，从到达蓝星开始，他需要在蓝星待多少天（期间无传送门打开）。

输入格式

本题存在多组数据，对于每组数据：

一行，一个 8 位整数n表示蓝星今天的日期，保证数据为合法日期。

输出格式

对于每组数据分别输出答案：

一行，两个整数表示，下一次传送门打开日期和从达到蓝星开始（上一次传送门打开日期一定存在）他一共需要在蓝星待多少天。数字以空格隔开。

如果下次传送门打开日期超过**99999999**，只输出一行 `N0, mG g0D!`

输入样例

```
20200202
66810404
95180628
```

输出样例

```
20211202 669
70011007 259698
N0, mG g0D!
```

E2-J

法一：一天一天往前找，找到上一个回文日期；一天一天往后找，找到下一个回文日期。

平均循环次数大约为100000000/366=273224

法二：把所有回文日期都找出来，共366个，存到一个数组里，然后遍历数组找到>date的最小回文日期和<=date的最大回文日期。

几遍366次的循环，运算次数应为366的数倍。

法三：一年最多只有一个回文日期，一年一年往上去找>date的回文日期，往下找<=date的回文日期。

平均循环次数大约为10000/366=27.3

计算两个回文日期的间隔天数：

一天一天算，平均循环次数大约为273224次

一年一年算，然后按照月算，按照日算，运算基本不会超过100次。

优化算法

G void学数学

时间限制：1500ms 内存限制：65536kb

通过率：274/493 (55.58%) 正确率：274/1408 (19.46%)

题目描述

离散数学老师说："我们有两个集合A和B,那么A和B的对称差就是$(A-B)\cup(B-A)$"

比如$A=\{1,2,3,4\}$,$B=\{2,5,6\}$

那么$A-B=\{1,3,4\}$,$B-A=\{5,6\}$。 $A-B$就是将A中属于B的部分去掉 ,$B-A$就是将B中属于A的部分去掉。

A和B的对称差就是$\{1,3,4,5,6\}$，是上面两个集合的并

然后老师就留下了T道课后作业，每一道都要求出A、B两个集合的对称差 。但是void很笨，他不会算，所以他想请你帮帮他。

输入

第一行一个整数T，代表作业的数量（数据组数）

对于每一组数据

第一行为一个整数n，表示集合A的基数，也就是集合A中有多少个元素

第二行有n个整数，表示集合A中的元素

第三行为一个整数m，表示集合B的基数

第四行有m个整数，表示集合B中的元素

输出

输出共T行，每一行从小到大输出若干个整数，整数之间以一个空格隔开，表示A和B的对称差中的所有元素

C3-G

法一：遍历A中的每个数，遍历A中数a时，遍历B中的每个数看是否存在B中数b=a，若存在则不输出a，否则应当输出a。再相同方法遍历B，遍历的时候将应当输出的数存到一个数组里，将数组排序，然后输出。

时间复杂度：$O(nm)$，太慢了！还没算排序耗时。

法二：利用哈希表，由于集合中元素数据范围在5000以内，用数组h[5005]统计每个数出现的次数。遍历A中的每个数a，h[a]++；遍历B中的每个数b，h[b]++。然后遍历数组h，如果h[i]==1则输出。

时间复杂度： $O(n+m+\mathrm{M})$，M为哈希表大小（数据范围），本题中为5000.

缺点：若元素数据范围M过大，则内存不够建立数组h。

法三：双指针法，不过多介绍。

时间复杂度：若集合中元素从小到大给出，则为$O(n+m)$；若乱序给出，也可达到$O(nlogn+mlogm)$

优点：空间复杂度为$O(n+m)$

优化算法

J 有理数

时间限制：1000ms 内存限制：65536kb

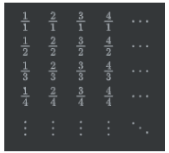
通过率：65/297 (21.89%) 正确率：65/876 (7.42%)

题目背景

在离散数学（2）中，大家将会学到有关无穷集的大小的比较的问题。可以证明，有理数集和整数集是等势的。为证明有理数集和整数集等势，我们只需要证明正有理数集和正整数集等势。这道题与其中的一个证明方法有关。

题目描述

为此，我们考虑给所有的正有理数进行一个"编号"。考虑构造以下的无穷方阵（显然这个方阵涵盖所有正有理数）：



然后，我们按照从右上到左下的对角线依次为这些有理数"编号"，依次是：

第一个对角线	第二个对角线		第三个对角线			第四个对角线				第五个对角线					...
1/1	2/1	1/2	3/1	2/2	1/3	4/1	3/2	2/3	1/4	5/1	4/2	3/3	2/4	1/5	...

$$\frac{1}{1},\frac{2}{1},\frac{1}{2},\frac{3}{1},\frac{2}{2},\frac{1}{3},\frac{4}{1},\frac{3}{2},\frac{2}{3},\frac{1}{4},\frac{5}{1},\frac{4}{2},\cdots$$

但是，其中一些分数是重复的（如$\frac{2}{2}$和$\frac{1}{1}$），因此我们要删除所有的形如$\frac{a}{b}$但$\gcd(a,b)\neq 1$的分数。最终分数与"编号"对应关系如下：

编号	1	2	3	4	5	6	7	8	9	10	11	...
分数	$\frac{1}{1}$	$\frac{2}{1}$	$\frac{1}{2}$	$\frac{3}{1}$	$\frac{1}{3}$	$\frac{4}{1}$	$\frac{3}{2}$	$\frac{2}{3}$	$\frac{1}{4}$	$\frac{5}{1}$	$\frac{1}{5}$	...

本题的目标是，给出a和b的值，求出正分数$\frac{a}{b}$的编号（数据保证$\gcd(a,b)=1$）。

输入输出说明

输入说明

一行，两个正整数a和b，保证$\gcd(a,b)=1$。

输出说明

一行，一个正整数x，表示$\frac{a}{b}$的编号是x。

数据范围与提示

gcd函数：表示最大公约数。ppt例1-9的代码可用，但是建议使用更快速的方法计算。

对于40%的数据，保证$a+b\leq 200$。这部分数据可以直接参考题意写双重循环+gcd验证，且gcd验证可以直接使用ppt例1-9的代码，直到找到题述分数为止，然后直接输出遍历到的gcd=1的有效分数的个数，即为编号。

对于80%的数据，保证$a+b\leq 4\times 10^3$。这部分数据也可以直接参考题意写双重循环+gcd验证，但是需要考虑更快速的gcd代码。

对于100%的数据，保证$a+b\leq 10^6$。这部分数据超出了第一次课的要求，仅留给有数论或信竞基础的同学思考。其他同学也可以在学习更多知识后再来思考本题。

参考代码

采用双重循环+gcd验证的同学可以参考以下代码（仅为部分代码）

```
int a, b, x = 0, flag = 0;
scanf("%d%d", &a, &b);
for(int s = 2; ; s++) // s 表示 i+j 的和
{
    for(int j = 1; j < s; j++)
    {
        int i = s - j; // 遍历到分数 i/j
        int g; // g = gcd(i,j)

        // do something

        if(g == 1) x++; // 说明这是一个符合要求的分数
        if(i == a && j == b) // 找到了！
        {
            flag = 1; break; // flag置1，退出循环
        }
    }
    if(flag) break;
}
printf("%d\n", x);
```

Author: Toby

E1-J

设a+b≤M。

双重循环法总的循环次数与M^2成正比（忽略低阶项），时间复杂度为$O(M^2)$。

求gcd若采取枚举法的话，时间复杂度为$O(M)$，总时间复杂度为$O(M^3)$。

求gcd若采取辗转相除法，时间复杂度为$O(logM)$，总时间复杂度为$O(M^2logM)$。

这道题的满分做法请参考题解，时间复杂度需要为$O(MlogM)$。

J 不可分解的01串

时间限制：1000ms 内存限制：65536kb

通过率：3/34 (8.82%) 正确率：3/132 (2.27%)

题目描述

求长度为 n 的 01 字符串中不可分解字符串的个数。

“不可分解”指不能写成两个或多个相同字符串的拼接。

- `100100100` == `100` $*3$ ，可分解
- `1101` ，不可分解

由于结果可能很大，请输出答案对 998244353 取模的结果。

输入

一个正整数 n .

输出

一行一个整数，表示答案对 998244353 取模的结果。

数据范围

n 为正整数

对于95%的数据， $n \leq 2 \times 10^6$

对于5%的数据， $n \leq 10^{15}$

C3-J

拿到95%的分数，至少需要$O(nlogn)$的算法。

拿到100%的分数，需要$O(\sqrt{n})$的算法。

Part 2: 空间复杂度

概念简介

与时间复杂度类似，空间复杂度是对一个算法在运行过程中临时占用存储空间大小的量度。

相比于时间限制比较严格，现在的空间限制较为宽松，一般在恰当的范围之内，不会发生 Memory Limit Exceed(MLE) 的问题。

但是，仍然有必要向同学们普及程序占用的空间的相关知识。

我们计算程序占用的内存空间时，一般忽略零散的变量，而关注数组。
我们按照数组的长度，乘以它所对应的单个变量类型的大小，求和之后就可以得到程序占用的内存空间。

计算内存

例：

```
//int为4B（32位）,long long为8B(64位)
//char为1B（8位）
//float为4B,double为8B
////////////////////////////
int a[1000000];
double b[10000];
char c[100];
```

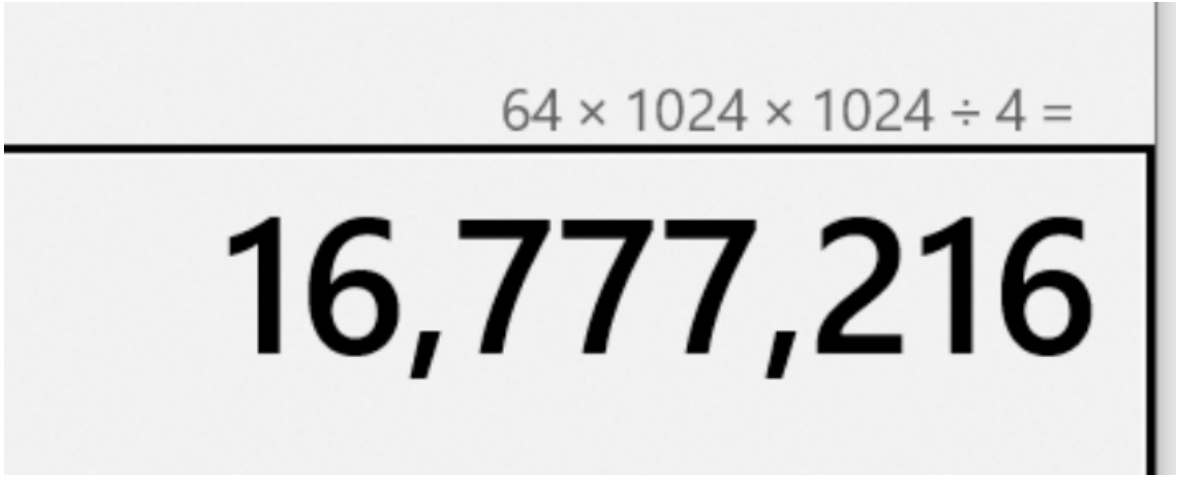
上述代码占用的内存为：$1000000 * 4 + 10000 * 8 + 100 * 1 = 4080100\text{B}$

当然对于这个结果，我们一般转化成MB来粗略估计，4080100B转化成MB大约是3.89MB（转化关系为除以两次1024）。

而题目限制如果是65536KB，也就是64MB，程序占用的内存没有超过空间限制，不会MLE。

计算内存

那么我们也可以计算：
如果全开int类型的数组，可以开多大?



假设空间限制仍然是64MB，那么我们最多可以开这么大的数组。
当然，一般同学们无需用到这么大的数组，10的6次方数量级大小的数组已经算的上很大了。

重点！！！
不建议同学们在函数中开过大的数组，大数组请放到函数外面，成为全局变量，否则会导致Runtime Error(RE)。
一般来说，函数内部开的数组大小我们不建议超过10的5次方，超过这个范围的数组都可以开在函数外面。

致谢

本文题目来自北航OJ系统(accoding.buaa.edu.cn)，2022程序设计基础课程的上机题目。

部分内容来自2022程设助教张宇阳给同学们的参考资料《时间和空间复杂度初步》。感谢支持!

感谢聆听！！！

佟振宇　2022.9.24

北京航空航天大學 BEIHANG UNIVERSITY

学业与发展支持中心

朋辈辅导直播

程序设计之
时间复杂度与空间复杂度简介

主讲人：佟振宇

北航学业与发展支持中心朋辈辅导师
宇航学院2021级本科生
2022程序设计基础课程助教

直播时间：2022年9月24日 14:00-15:00

学生反馈二维码

微博扫码关注