

# C 语言程序设计

考试分级题库(2 小时版本)

2019-11-11

# 说 明

《C 语言程序设计》课程的考试形式为上机考试，考试时间 2 小时，满分 100 分。考试题的题型为编程题，数量为 8 道题。考试题目全部由计算机从本题库自动抽取组成(D 类除外)。A 类题目抽取 4 道，每题 13 分；B 类题目抽取 2 道，每题 11 分；C 类题目抽取 1 道，每题 13 分；D 类题目抽取 1 道，每题 13 分。

本考试题库适用于使用以下教材的课程作为期末考试之用，试题难度能够匹配教材的知识点和教学目标，可以有效检查学生的学习效果。

✓ 主教材：《C 语言程序设计案例教程》，吴绍根 黄达峰 编著；

清华大学出版社，2018。ISBN：978-7-302-50602-7。

✓ 配套教材：《C 语言程序设计案例教程—习题解答》，黄达峰 吴绍根 编著；

清华大学出版社，2018。ISBN：978-7-302-50582-2。

本课程配套的学习网站 <http://www.StartFromC.com> 。

可以通过微信公众号“从 c 开始”获取更多的 c 语言学习资料。



# 目 录

第 1 部分 A 类题目——简单 .....	4
第 2 部分 B 类题目——中等 .....	17
第 3 部分 C 类题目——较难 .....	30
第 4 部分 D 类题目——课外 .....	36

## 第 1 部分 A 类题目——简单

- 题目-01. 【练习 3- 6】求 3 个数的最小值
- 题目-02. 【练习 3- 9】奇偶数判定
- 题目-03. 【练习 3- 13】输出正整数
- 题目-04. 【练习 3- 14】计算前n 个正整数之和
- 题目-05. 【练习 3- 17】素数判定
- 题目-06. 【练习 3- 18】鸡兔同笼
- 题目-07. 【练习 4- 4】三角形判定
- 题目-08. 【练习 4- 5】直角三角形判定
- 题目-09. 【练习 4- 16】阶乘之和
- 题目-10. 【练习 4- 17】斐波那契数列
- 题目-11. 【练习 5- 11】九九乘法表
- 题目-12. 【练习 6- 1】水仙花数
- 题目-13. 【练习 6- 6】循环字母字符串

## 第 2 部分 B 类题目——中等

- 题目-14. 【练习 4- 6】字符分类
- 题目-15. 【练习 4- 7】成绩评定
- 题目-16. 【练习 5- 8】个人所得税
- 题目-17. 【练习 6- 9】完美数
- 题目-18. 【练习 7- 3】逆序字符串
- 题目-19. 【练习 7- 15】百分制转换等级制
- 题目-20. 【练习 7- 16】平年闰年判断

题目-21. 【练习 8- 9】线段中点

题目-22. 【练习 8- 10】转换日期

题目-23. 【练习 10- 3】读取大写字符串

### 第 3 部分 C 类题目——较难

题目-24. 【练习 6- 4】公因数与公倍数

题目-25. 【练习 6- 16】二进制数 1 的个数

题目-26. 【练习 6- 19】分解质因数

题目-27. 【练习 7- 5】约瑟夫问题

题目-28. 【练习 9- 5】棋盘布局判断

### 第 4 部分 D 类题目——课外

课外样题 1. 统计进位

课外样题 2. 分数拆分

# 第 1 部分 A 类题目——简单

## 题目-01. 【3-6】求 3 个数的最小值

### 问题描述

编写 C 语言程序，从键盘读入 3 个整数并输出这 3 个数中的最小值到屏幕。

### 输入格式

一共 3 行数据，每行包含一个整数。

### 输出格式

一个整数，行末没有换行符。

### 数据规模与约定

每个整数  $n$  的值约定为  $-10\,000\,000 \leq n \leq 10\,000\,000$ 。

### 样例输入

```
123
-234
345
```

### 样例输出

```
-234
```

```
#include<stdio.h>
int main()
{
    int a;
    int b;
    int c;

    scanf ("%d",&a);
    scanf ("%d",&b);
    scanf ("%d",&c);

    if (a<b){
        printf ("%d",a);
    }else if (b<c){
        printf ("%d",b);
    }else{
        printf ("%d",c);
    }

    return 0;
}
```

## 题目-02. 【3-9】奇偶数判定

### 问题描述

编写C语言程序，从键盘读入一个整数，然后判断该整数是奇数或者偶数，并输出结果到屏幕。如果是奇数，则输出“odd”；如果是偶数，则输出“even”。

### 输入格式

一共 1 行数据，包含一个整数。

### 输出格式

输出“odd”或“even”，行末没有换行符。

### 数据规模与约定

每个整数n的值约定为  $1 \leq n \leq 10\,000\,000$ 。

### 样例输入

135
-----

### 样例输出

odd
-----

```
#include <stdio.h>
int main () {
    int a;

    scanf ("%d",&a);

    if(a%2) {
        printf("odd");
    }else{
        printf ("even");
    }
    return 0;
}
```

## 题目-03. 【3-13】输出正整数

### 问题描述

编写C语言程序，从键盘读入一个整数  $n$ ，然后按照从小到大的顺序输出前  $n$  个正整数到屏幕(不输出 2 的倍数、3 的倍数以及 5 的倍数)，每个整数后面都有一个空格。

### 输入格式

一共 1 行数据，包含一个整数  $n$ 。

### 输出格式

输出 1 行，包含若干个整数，每个整数后面都有一个空格，行末没有换行符。

### 数据规模与约定

整数  $n$  的值约定为  $1 \leq n \leq 100$ 。

### 样例输入

59
----

### 样例输出

1 7 11 13 17 19 23 29 31 37 41 43 47 49 53 59
---

```
#include <stdio.h>
int main ()
{
    int n;
    int i=0;
    scanf ("%d",&n);
    while (i<n){
        i++;
        if ((i%2==0) || (i%3==0) || (i%5==0)) {
            continue;
        }
        printf ("%d ",i);
    }
    return 0;
}
```



## 题目-04. 【3-14】计算前 $n$ 个正整数之和

### 问题描述

编写 C 语言程序，从键盘读入一个整数  $n$ ，然后计算前  $n$  个正整数之和，并输出到屏幕。

### 输入格式

一共 1 行数据，包含一个整数  $n$ 。

### 输出格式

输出 1 行，包含一个整数，行末没有换行符。

### 数据规模与约定

整数  $n$  的值约定为  $1 \leq n \leq 10\,000$ 。

### 样例输入

100
-----

### 样例输出

5050
------

```
#include <stdio.h>
int main ()
{
    int n;
    int i=1;
    int sum=0;

    scanf ("%d",&n);

    while (i<=n){
        sum=sum+i;
        i++;
    }
    printf ("%d",sum);
    return 0;
}
```

## 题目-05. 【3-17】素性测试

### 问题描述

质数(Prime number)，又称素数，指在大于 1 的整数中，除了 1 和该数自身外，无法被其他整数整除的数(或者说是只有 1 与该数本身两个正因数的数)。大于 1 的整数若不是素数，则称之为合数。

编写 C 语言程序，从键盘读入一个整数 n，然后判断 n 是素数或者合数。如果 n 是素数，则输出信息 “prime number”；如果 n 是合数，则输出信息 “composite number”。

### 输入格式

一共 1 行数据，包含一个整数n。

### 输出格式

输出 1 行信息，行末没有换行符。

### 数据规模与约定

整数n的值约定为  $2 \leq n \leq 1\,000\,000$ 。

### 样例输入

31
----

### 样例输出

prime number
--------------

```
#include <stdio.h>
int main ()
{
    int n;
    int i=1;

    scanf ("%d",&n);

    while (i<=n){
        if (n%i==0){
            printf ("composite number");
            break;
        }
        i++;
    }

    if(i==n){
        printf ("prime number");
    }
    return 0;
}
```

## 题目-06. 【3-18】鸡兔同笼问题

### 问题描述

“鸡兔同笼”问题是中国古代的数学名题之一。大约在 1500 年前，《孙子算经》中就记载了这个有趣的问题。书中是这样叙述的：

今有雉兔同笼，上有三十五头，下有九十四足，问雉兔各几何？

这 4 句话的意思是：有若干只鸡兔同在一个笼子里，从上面数，有 35 个头，从下面数，有 94 条腿。问笼中各有多少只鸡和兔？

编写 C 语言程序，从键盘读入代表头的总数量的整数 head 以及代表腿的总数量的整数 leg，然后计算鸡和兔的数量并输出结果到屏幕。如果有多个解，则只需要输出一个解即可。如果无解，则输出信息“Error”。这里约定鸡和兔的数量都是不少于一只。

### 输入格式

一共 1 行数据，包含 2 个整数 head 和 leg，之间使用一个空格分隔。

### 输出格式

输出 1 行信息，行末没有换行符。

如果有解，则包含 2 个整数，分别表示鸡和兔的数量，之间使用一个空格分隔

如果无解，则包含信息“Error”。

### 数据规模与约定

整数 head 的值约定为  $2 \leq n \leq 10\,000$ 。

整数 leg 的值约定为  $6 \leq n \leq 10\,000$ 。

### 样例输入

35 94
-------

### 样例输出

23 12
-------

```
#include <stdio.h>
int main ()
{
    int c=1;
    int r;
    int h;
    int l;
    scanf ("%d %d",&h,&l);

    while (c<h){
        r = h - c;
        if (c * 2+r * 4==l){
            break;
        }
        c++;
    }
}
```

```
    if (c==h) {  
        printf ("Error");  
    }else{  
        printf ("%d %d", c, r);  
    }  
  
return 0;  
}
```

## 题目-07. 【4-4】三角形判定

### 问题描述

编写C语言程序，从键盘读入 3 个整数(使用空格分隔)作为 3 条线段长度，如果这 3 条线段能够组成一个三角形，那么输出信息 “yes” 到屏幕，否则就输出 “no”。

### 输入格式

一共 1 行数据，包含 3 个代表线段长度的整数a、b、c。

### 输出格式

输出 1 行信息，行末没有换行符。

### 数据规模与约定

整数a 的值约定为  $1 \leq a \leq 100\,000$ 。

整数b 的值约定为  $1 \leq b \leq 100\,000$ 。

整数c 的值约定为  $1 \leq c \leq 100\,000$ 。

### 样例输入

3 8 4
-------

### 样例输出

no
----

```
#include <stdio.h>
int main () {
    int e1;
    int e2;
    int e3;

    scanf ("%d %d %d",&e1,&e2,&e3);
    if ((e1+e2>e3)&&
        (e1+e3>e2)&&
        (e2+e3>e1)) {
        printf ("yes");
    }else{
        printf ("no");
    }
    return 0;
}
```

## 题目-08. 【4-5】直角三角形判定

### 问题描述

编写C语言程序，从键盘读入 3 个整数(使用空格分隔)作为 3 条线段长度，如果这 3 条线段能够组成一个直角三角形，那么输出信息 “yes” 到屏幕，否则就输出 “no”。

### 输入格式

一共 1 行数据，包含 3 个代表线段长度的整数a、b、c。

### 输出格式

输出 1 行信息，行末没有换行符。

### 数据规模与约定

整数a 的值约定为  $1 \leq a \leq 100\,000$ 。

整数b 的值约定为  $1 \leq b \leq 100\,000$ 。

整数c 的值约定为  $1 \leq c \leq 100\,000$ 。

### 样例输入

3 4 5
-------

### 样例输出

yes
-----

```
#include <stdio.h>
int main () {
    int e1;
    int e2;
    int e3;

    scanf ("%d %d %d",&e1,&e2,&e3);
    if ((e1*e1+e2*e2==e3*e3) ||
        (e1*e1+e3*e2==e2*e2) ||
        (e2*e2+e3*e3==e1*e1)) {
        printf ("yes");
    }else{
        printf ("no");
    }
    return 0;
}
```

## 题目-09. 【4-16】阶乘之和

### 问题描述

编写 C 语言程序，从键盘读入一个整数  $n$ ，然后计算不超过  $n$  的所有正整数的阶乘  $n!$  之和，并输出结果到屏幕。

### 输入格式

一共 1 行数据，包含一个整数  $n$ 。

### 输出格式

输出 1 行，包含一个整数，行末没有换行符。

### 数据规模与约定

整数  $n$  的值约定为  $1 \leq n \leq 20$ 。

### 样例输入

13
----

### 样例输出

6749977113
------------

```
#include <stdio.h>
int main()
{
    int n;
    int ls;
    int i;
    scanf("%d",&n);
    long long s=0,t=1;
    for(i=1;i<=n;i++){
        t=1;
        for(ls=1;ls<=i;ls++)
        {
            t*=ls;
        }
        s+=t;
    }
    printf("%lld",s);
    return 0;
}
```

## 题目-10. 【4-17】斐波那契数列

### 问题描述

斐波那契(Fibonacci)数列，又称黄金分割数列：该数列的第一项是 0，第二项是 1，从第三项起每一项都是前两项之和。

编写 C 语言程序，从键盘读入一个整数  $n$ ，然后输出斐波那契数列的前  $n$  项到屏幕，项与项之间使用空格分隔。

### 输入格式

一共 1 行数据，包含一个整数  $n$ 。

### 输出格式

输出 1 行，项与项之间使用空格分隔，即用  $n-1$  个空格分隔  $n$  项数据，行末没有空格也没有换行符。

### 数据规模与约定

整数  $n$  的值约定为  $1 \leq n \leq 93$ 。

### 样例输入

10
----

### 样例输出

0 1 1 2 3 5 8 13 21 34
------------------------

```
#include <stdio.h>
int main()
{
    int i;
    int n;
    int t1 = 0;
    int t2 = 1;
    int nextTerm;
    scanf("%d", &n);
    for (i = 1; i <= n; ++i)
    {
        printf("%d", t1);
        if(i!=n)printf(" ");
        nextTerm = t1 + t2;
        t1 = t2;
        t2 = nextTerm;
    }
    return 0;
}
```

对一半



## 题目-11. 【5-11】九九乘法表

### 问题描述

编写C语言程序，输出三角形的九九乘法表，要求左对齐。

### 输入格式

一共 1 行数据，包含一个整数n。

### 输出格式

使用英文大写字母“X”表示乘号，不要使用中文乘号。

使用换码序列 '\t' 实现每列数据左对齐。

每行行末都有一个换行符。

### 数据规模与约定

整数n 的值约定为  $1 \leq n \leq 9$ 。

### 样例输入

6

### 样例输出

```
1X1=1
1X2=2   2X2=4
1X3=3   2X3=6   3X3=9
1X4=4   2X4=8   3X4=12  4X4=16
1X5=5   2X5=10  3X5=15  4X5=20  5X5=25
1X6=6   2X6=12  3X6=18  4X6=24  5X6=30  6X6=36
```

```
#include<stdio.h>
int main () {
    int x,y,i;
    int m;
    scanf ("%d",&i);
    for (x=1;x<=i;x++){
        for (y=1;y<=x;y++){
            printf ("%dX%d=%d\t",y,x,x*y);
        }
        printf ("\n");
    }
    return 0;
}
```

## 题目-12. 【6-1】水仙花数

### 问题描述

一个  $n$  位整数，如果等于它的  $n$  个数字的  $n$  次方之和，则该  $n$  位数称为  $n$  位水仙花数。

■ 一个三位数，如果等于它的三个数字的三次方之和，则该三位数称为三位水仙花数；

■ 一个四位数，如果等于它的四个数字的四次方之和，则该四位数称为四位水仙花数；

例如，153 是其中一个三位水仙花数：

$$153 = 1^3 + 5^3 + 3^3 = 1 + 125 + 27$$

编写 C 语言程序，输出所有三位水仙花数。每行输出一个水仙花数。

### 输入格式

无。

### 输出格式

每行一个水仙花数和一个换行符。

### 数据规模与约定

无

### 样例输入

无
---

### 样例输出

153
370
371
407

```
#include <stdio.h>
int main()
{
    int hun, ten, ind, n;
    for( n=100; n<1000; n++ )
    {
        hun = n / 100;
        ten = (n-hun*100) / 10;
        ind = n % 10;
        if(n == hun*hun*hun + ten*ten*ten + ind*ind*ind)
            printf("%d\n", n);
    }
    return 0;
}
```

## 题目-13. 【6-6】循环字母字符串

### 问题描述

编写 C 语言程序，从键盘读入一个英文字母，如果是大写字母，则从该字母开始按升序输出大写形式的字母表，在输出最后一个字母 Z 后，继续从字母 A 输出，直到 26 个大写字母全部输出完毕；如果是小写字母，则从该字母开始按升序输出小写形式的字母表，在输出最后一个字母 z 后，继续从字母 a 输出，直到 26 个小写字母全部输出完毕。

### 输入格式

一共 1 行数据，包含一个 ASCII 字符。

### 输出格式

输出 1 行信息，行末没有换行符：

如果输入的字符是大写字母，则输出大写的循环字母表；

如果输入的字符是小写字母，则输出小写的循环字母表；

如果输入的字符不是字母，则输出信息 “error”。

### 数据规模与约定

所输入的字符，其 ASCII 十进制编号  $n$  满足  $33 \leq n \leq 126$ 。

### 样例输入

F
---

### 样例输出

FGHIJKLMNOPQRSTUVWXYZABCDE
----------------------------

```
#include <stdio.h>
int main ()
{
    char le;
    int i;
    scanf ("%c",&le);
    if ((le>='a') && (le<='z')){
        for (i=0;i<26;i++){
            printf ("%c",'a'+(le-'a'+i)%26);
        }
    }else if ((le>='A') && (le<='Z')){
        for (i=0;i<26;i++){
            printf ("%c",'A'+(le-'A'+i)%26);
        }
    }else {
        printf ("error");
    }
    return 0;
}
```

## 第 2 部分 B 类题目——中等

### 题目-14. 【4-6】字符分类

#### 问题描述

编写C语言程序，从键盘读入一个字符，然后输出该字符对应的 ASCII 编号的十进制数值和十六进制数值(后面显示字母'H'加以区分)到屏幕，并判断该字符的类型：

- 如果该字符是大写字母，则输出 “uppercase”；
- 如果该字符是小写字母，则输出 “lowercase”；
- 如果该字符是数字，则输出 “digital”；
- 如果该字符不是以上 3 种类型，则输出 “other”。

#### 输入格式

一共 1 行数据，包含 1 个字符。

#### 输出格式

输出 1 行信息，包含 1 个十进制数值、1 个十六进制数值以及相应的类型信息，三者之间使用一个空格分隔，行末没有空格也没有换行符。

#### 数据规模与约定

所输入的字符，其ASCII十进制编号  $n$  满足  $33 \leq n \leq 126$ 。

#### 样例输入

6
---

#### 样例输出

54 36H digital
----------------

```
#include<stdio.h>
int main()
{
    char ch;
    scanf("%c",&ch);
    if(!((ch>='a' &&ch<='z') || (ch>='A' &&ch<='Z') || (ch>='1' && ch<='9'))){
        printf("other");
    }else if (ch>='a' &&ch<='z'){
        printf ("%d %xH lowercase",ch,ch);
    }else if (ch>='1' && ch<='9'){
        printf ("%d %xH digital",ch,ch);
    }else if(ch>='A' &&ch<='Z'){
        printf ("%d %xH uppercase",ch,ch);
    }
}
```

## 题目-15. 【4-7】成绩评定

### 问题描述

某学校对学生的评价标准如下(假设只有语文、数学和英语 3 门课程, 分数是 100 分制的整数):

- 三门课的平均分不低于 80, 且至少有一门课不低于 90, 则评为“优秀”;
- 每一门课都不低于 75, 则评为“良好”;
- 三门课的平均分不低于 60, 且至多只有一门课低于 60, 则评为“合格”;
- 如果不是“优秀”、“良好”、“合格”之一, 则评为“不合格”。

编写C 语言程序, 从键盘读入 3 门课程成绩, 然后输出相应的评价等级。如果评为“优秀”, 则输出信息“excellent”; 如果评为“良好”, 则输出信息“good”; 如果评为“合格”, 则输出信息“pass”; 如果评为“不合格”, 则输出信息“fail”。评定原则是“就高不就低”, 即如果同时满足优秀和良好, 则评为优秀。

### 输入格式

一共 1 行数据, 包含 3 个代表 3 门课程分数的整数a、b、c, 之间使用一个空格分隔。

### 输出格式

输出相应的信息, 行末没有换行符:

如果评为“优秀”, 则输出信息“excellent”;

如果评为“良好”, 则输出信息“good”;

如果评为“合格”, 则输出信息“pass”;

如果评为“不合格”, 则输出信息“fail”。

### 数据规模与约定

整数a 的值约定为  $0 \leq a \leq 100$ 。

整数b 的值约定为  $0 \leq b \leq 100$ 。

整数c 的值约定为  $0 \leq c \leq 100$ 。

### 样例输入

50 60 70
----------

### 样例输出

pass
------

```
#include <stdio.h>
int main ()
{
    int chinese;
    int math;
    int english;

    scanf ("%d",&chinese);
    scanf ("%d",&math);
    scanf ("%d",&english);

    if (((chinese+math+english) >= (80 * 3)) &&
        ((chinese>=90) || (math>=90) || (english>=90))) {
```

```

        printf ("excellent");
    }else if ((chinese>=75) && (math>=75) && (english>=75)){
        printf ("good");
    } else if (((chinese+ math +english)>=(60 * 3)) &&
        (((chinese>=60) && (math>=60)) ||
        ((chinese>=60) && (english>=60)) ||
        ((math>=60) && (english>=60)))){
        printf ("pass");
    }else {
        printf ("fail");
    }
    return 0;
}

```

## 题目-16. 【5-8】个人所得税

### 问题描述

假设某地区个人所得税的缴纳方式如下：

月总收入在 1600 元以下(含 1600 元)不需要缴纳个人所得税。月总收入在 1600 元以上，那么需要缴税的部分为：月总收入-1600，简称“应税收入”，且分级逐级计算：

- 应税收入在 500 元内(含 500 元)的部分，税率为 5%；
- 应税收入在 500 元~2000 元内(含 2000 元)的部分，税率为 10%；
- 应税收入 2000 元~5000 元内(含 5000 元)的部分，税率为 15%；
- 应税收入 5000 元~10000 元内(含 10000 元)的部分，税率为 20%；
- 应税收入在 10000 元以上的部分，税率为 30%。

例如，某职工的当月的总收入为 7000 元，那么他应缴的个人所得税计算如下：

- (1) 应税收入 = 月总收入 - 1600 = 7000 - 1600 = 5400 (元)
- (2) 500 元内的所得税 = 500 \* 5% = 25 (元)
- (3) 500 元~2000 元内的所得税 = (2000 - 500) \* 10% = 150 (元)
- (4) 2000 元~5000 元内的所得税 = (5000 - 2000) \* 15% = 450 (元)
- (5) 5000 元~10000 元内的所得税 = (5400 - 5000) \* 20% = 80 (元)
- (6) 应缴纳的个人所得税共计 = 25 + 150 + 450 + 80 = 705 (元)

编写 C 语言程序，从键盘读入月总收入，然后计算应缴的个人所得税。

### 输入格式

一共 1 行数据，包含一个整数n。

### 输出格式

输出 1 行，包含对应的所得税金额，保留到小数点后 2 位。行末没有换行符。

### 数据规模与约定

整数n 的值约定为  $0 \leq n \leq 10\,000\,000$ 。

### 样例输入

7000
------

### 样例输出

705.00
--------

```
#include <stdio.h>
int main ()
{
    double to;
    double ta;
    double tax;

    scanf ("%lf",&to);

    ta=to-1600;
    if (ta>10000){
        tax = 500 * 0.05 + (2000 - 500) * 0.2;
```

```

        tax += (5000-200) * 0.15 + (10000-5000) * 0.2;
        tax += (ta-10000) * 0.3;
    }else if (ta>5000){
        tax = 500 * 0.05 + (2000-500) * 0.1;
        tax += (5000-2000) * 0.15+(ta-5000) * 0.2;
    }else if (ta>2000){
        tax = 500 * 0.05 + (2000-500) * 0.1;
        tax += (ta-2000) * 0.15;
    }else if (ta>500){
        tax = 500 * 0.05+(ta-500) * 0.1;
    }else if (ta>0){
        tax = ta * 0.05;
    }else {
        tax=0;
    }
    printf ("%0.2f", tax);
    return 0;
}

```



## 题目-17. 【6-9】完美数

### 问题描述

正整数 $n$ 的所有小于 $n$ 的正因数之和如果等于 $n$ 本身，则称 $n$ 是完美数(Perfect Number)。例如，6和28都是完美数，因为：

■  $6 = 1 + 2 + 3$

■  $28 = 1 + 2 + 4 + 7 + 14$

编写C语言程序，从键盘读入一个整数 $n$ ，如果 $n$ 是完美数，则输出信息“yes”，否则输出信息“no”。

### 输入格式

一共1行数据，包含一个整数 $n$ 。

### 输出格式

输出信息“yes”或者“no”，行末没有换行符。

### 数据规模与约定

整数 $n$ 的值约定为  $1 \leq n \leq 10\,000$ 。

### 样例输入

28
----

### 样例输出

yes
-----

```
#include<stdio.h>
int main ()
{
    int n;
    int i;
    int sum=0;
    scanf ("%d",&n);
    for (i=1;i<n;i++){
        if (n%i==0){
            sum=sum+i;
        }
    }
    if (sum==n){
        printf ("yes");
    }else {
        printf ("no");
    }
    return 0;
}
```

## 题目-18. 【7-3】逆序字符串

### 问题描述

编写C语言程序，从键盘读取不超过 10 个字符，然后逆序输出。

### 输入格式

一共 1 行数据，包含一个字符串s。

### 输出格式

逆序输出字符串s的内容，行末没有换行符。

### 数据规模与约定

输入的字符串s不包含空格和制表符。

所输入的字符，其ASCII十进制编号 n 满足  $33 \leq n \leq 126$ 。

### 样例输入

```
helloworldhowareyou
```

### 样例输出

```
dlrowolleh
```

```
#include <stdio.h>
#define N 10
int main ()
{
    char str[N];
    int i;
    for(i=0;i<N;i++){
        scanf ("%c",&str[i]);
        if (str[i]=='\n'){
            break;
        }
    }
    while(--i>=0){
        printf ("%c",str[i]);
    }
    return 0;
}
```

## 题目-19. 【7-15】百分制转换等级制

### 问题描述

某学校使用等级制评定学生成绩，原来的百分制按照如下规则转换为等级制：95 分及以上评为 A，85 分及以上评为B，70 分及以上评为 C, 60 分及以上评为D，60 分以下评为 E。

(1) 编写一个 C 语言函数，函数名字是 `to_grade`，返回值是 `char` 类型，参数列表有一个 `int` 类型变量 `score` 作为形式参数。

函数 `to_grade()` 的功能是按照给定的百分制整数分数 `score` 计算出相应的等级，并把该等级返回。如果整数变量 `score` 不满足条件 “ $0 \leq score \leq 100$ ”，则函数 `to_grade()` 返回值是 ' '（即空格）。函数 `to_grade()` 不允许从键盘读取数据，也不允许输出数据到屏幕。

函数 `to_grade()` 对应的函数原型如下：

```
char to_grade(int score);
```

(2) main 函数调用函数 `to_grade()` 的测试代码如下：

```
#include <stdio.h>
char to_grade(int score);
int main()
{
    int score;
    char grade;
    scanf("%d", &score);
    grade = to_grade(score);
    if(grade != ' '){
        printf("%c", grade);
    }else{
        printf("error");
    }
    return 0;
}
// 你编写的代码将会嵌入到这里
```

### 输入格式

一共 1 行数据，包含整数 `score`。

### 输出格式

无。

### 数据规模与约定

无。

### 样例输入

```
80
```

### 样例输出

```
C
```

```

#include <stdio.h>
char to_grade(int score);
int main ()
{
    int score;
    char grade;
    scanf ("%d",&score);
    grade=to_grade(score);
    if(grade!=' '){
        printf ("%c",grade);
    }
    return 0;
}
char to_grade(int score)
{
    char grade;
    if ((score>100)|| (score<0)) {
        grade=' ';
    }else if (score>95){
        grade='A';
    }else if (score>85){
        grade='B';
    }else if (score>70){
        grade='C';
    }else if (score>60){
        grade='D';
    }else if (score>0){
        grade='E';
    }
    return grade;
}

```

## 题目-20. 【7-16】平年闰年判断

### 问题描述

平年与闰年的判断标准如下：

- 如果年份是 100 的倍数，且能被 400 整除，则该年份是闰年；
- 如果年份不是 100 的倍数，且能被 4 整除，则该年份是闰年；
- 如果以上都不满足，则该年份为平年。

(1) 编写一个 C 语言函数，函数名字是 `is_leap`，返回值是 `int` 类型，参数列表有一个 `int` 类型变量 `year` 作为形式参数。

函数 `is_leap()` 的功能是根据给定的 `year` 值来判断该年份是平年或闰年，如果是闰年则返回整数 1，即逻辑值“真”，如果是平年则返回整数 0，即逻辑值“假”。函数 `is_leap()` 不允许从键盘读取数据，也不允许输出数据到屏幕。

函数 `is_leap()` 对应的函数原型如下：

```
int is_leap(int year);
```

(2) main 函数调用函数 `is_leap()` 的测试代码如下：

```
#include <stdio.h>
int is_leap(int year);
int main()
{
    int year;
    scanf("%d", &year);
    if((year < 1900)|| (year >
        9999)){ printf("error");
    }else
        if(is_leap(year)){ p
            rintf("366");
        }else{
            printf("365");
        }
    return 0;
}
```

### 输入格式

一共 1 行数据，包含整数 `year`。

### 输出格式

无。

### 数据规模与约定

每个整数 `year` 的值约定为  $1900 \leq year \leq 9999$ 。

### 样例输入

2020

## 样例输出

366

```
#include <stdio.h>
int is_leap(int year);
int main ()
{
    int year;
    int days=365;
    scanf ("%d",&year);
    if (is_leap(year)) {
        days++;
    }
    printf ("%d",days);
    return 0;
}
int is_leap(int year)
{
    int is_leap=0;
    if ((year%400==0) || ((year%100 !=0) && (year%4==0))) {
        is_leap=1;
    }
    return is_leap;
}
```

## 题目-21. 【8-9】线段中点

### 问题描述

现有结构体定义如下：

<pre>struct point{     double x;     double y; }; typedef struct point Point;</pre>
(1) 编写 2 个 C 语言函数 <code>input_point()</code> 和 <code>middle()</code> ，分别对应的函数原型如下：
<pre>void input_point(Point * a, Point * b); Point middle(Point a, Point b);</pre>

函数 `input_point()` 的功能是从键盘读入 2 个点的坐标分别保存到指针 `a` 和 `b` 所指向的 `Point` 类型的变量。

函数 `middle()` 的功能是计算以点 `a` 和点 `b` 为端点的线段 `ab` 的中点的坐标保存到 `Point` 类型的变量，并作为返回值返回。该函数不允许从键盘读取数据，也不允许输出数据到屏幕。

(2) `main` 函数调用函数 `input_point()` 和 `middle()` 的测试代码如下：

```
#include <stdio.h>  
#include <math.h>  
struct point{  
    double x;        // 点的 x 坐标  
    double y;        // 点的 y 坐标  
};  
typedef struct point Point;  
void input_point(Point * a, Point * b);  
Point middle(Point a, Point b);  
int main()  
{  
    Point a;        // 点 a  
    Point b;        // 点 b  
    Point m;        // 线段 ab 的中点  
    input_point(&a, &b);  
    m = middle(a, b);  
    printf("%f %f", m.x, m.y);  
    return 0;  
}  
// 你编写的代码将会嵌入到这里
```

### 输入格式

一共 2 行数据：

第 1 行包含点 `a` 的 `x` 坐标和 `y` 坐标，之间使用空格分隔；

第 2 行包含点 `b` 的 `x` 坐标和 `y` 坐标，之间使用空格分隔。



## 输出格式

无。

## 数据规模与约定

x 坐标的值约定为  $-100\,000 \leq x \leq 100\,000$ 。

y 坐标的值约定为  $-100\,000 \leq y \leq 100\,000$ 。

## 样例输入

1.2	3.4
5.6	7.8

## 样例输出

3.400000	5.600000
----------	----------

```
#include<stdio.h>
#include <math.h>
struct point{
    double x;
    double y;
};
typedef struct point Point;

void input_point(Point * a, Point * b);
Point middle(Point a, Point b);

int main ()
{
    Point a;
    Point b;
    Point m;

    input_point (&a,&b);
    m=middle(a,b);
    //printf ("a(%f,%f)、b(%f,%f)",a.x,a.y,b.x,b.y);
    printf ("%f,%f",m.x,m.y);
    return 0;
}

void input_point(Point * a,Point * b)
{
    scanf ("%lf %lf",&a->x,&a->y);
    scanf ("%lf %lf",&b->x,&b->y);
}

Point middle (Point a,Point b)
{
    Point m;
    m.x=(a.x+b.x)/2;
    m.y=(a.y+b.y)/2;
    return m;
}
```

## 题目-22. 【8-10】转换日期

### 问题描述

现有结构体定义如下：

```
struct date{
    int month;    // 月
    int day;      // 日
};
typedef struct date Date;
```

(1) 编写一个C 语言函数：to\_date( )，对应的函数原型如下：

```
Date to_date(int n);
```

函数 to\_date( ) 的功能是根据给定的整数 n，计算一年中的第 n 天是几月几日，然后保存到 Date 类型的变量并作为返回值返回。假定年份是平年，即 2 月有 28 天。该函数不允许从键盘读取数据，也不允许输出数据到屏幕。

(2) main 函数调用函数to\_date ( )的测试代码如下：

```
#include <stdio.h>
struct date{
    int month;    // 月
    int day;      // 日
};
typedef struct date Date;
Date to_date(int n);
int main()
{
    Date date;
    int n;
    scanf("%d", &n);
    date = to_date(n);
    printf("%d %d", date.month, date.day);
    return 0;
}
// 你编写的代码将会嵌入到这里
```

### 输入格式

一共 1 行数据，包含一个整数n。

### 输出格式

无。

### 数据规模与约定

整数n 的值约定为  $1 \leq n \leq 365$ 。

### 样例输入

60

### 样例输出

3 1

```
#include <stdio.h>
struct date{
    int month;
    int day;
};
typedef struct date Date;
Date to_date(int n);

int main ()
{
    Date date;
    int n;
    scanf ("%d",&n);
    date=to_date(n);
    printf ("%d %d",date.month,date.day);
    return 0;
}

Date to_date(int day_of_year)
{
    Date date = {1,1};
    int days[12]={
        31,28,31,30,31,30,31,31,30,31,30,31
    };
    date.day=day_of_year;
    while (date.day>days[date.month-1]){
        date.day-=days[date.month-1];
        date.month++;
    }
    return date;
}
```

## 题目-23. 【10-3】读取大写字符串

### 问题描述

(1) 编写一个C 语言函数: `input_upper_string()`, 对应的函数原型如下:

```
void input_upper_string(char * str, int n);
```

函数 `input_upper_string()` 的功能: 从键盘输入长度不超过 `n-1` 个字符的一句话保存到指针变量 `str` 所指向的字符数组, 并检查每一个字符, 如果该字符是小写字母, 则转换为大写字母。

(2) `main` 函数调用函数 `input_upper_string()` 的测试代码如下:

```
#include <stdio.h>
#define LENGTH 20
#define SIZE (LENGTH + 1)
void input_upper_string(char * str, int n);
int main()
{
    char str[SIZE];
    input_upper_string(str, SIZE);
    printf("%s", str);
    return 0;
}
// 你编写的代码将会嵌入到这里
```

### 输入格式

一共 1 行数据, 包含一个字符串 `s`。

### 输出格式

无。

### 数据规模与约定

字符串 `s` 的长度 `n` 的值约定为  $1 \leq n \leq 1000$ 。

### 样例输入 1

```
I am learning C programming language.
```

### 样例输出 1

```
I AM LEARNING C PROG
```

### 样例输入 2

```
A1b2c3d4E5
```

### 样例输出 2

```
A1B2C3D4E5
```

```

#include <stdio.h>
#define LENGTH 20
#define SIZE (LENGTH + 1)
void input_upper_string(char * str, int n);
int main()

{
    char str[SIZE];
    input_upper_string(str, SIZE);
    printf("%s", str);
    return 0;
}
void input_upper_string(char * str,int n)
{
    char ch;
    int i;
    for (i=0;i<n-1;i++){
        scanf ("%c",&ch);
        if ((ch>='a') && (ch<='z')){
            ch=ch-('a' - 'A');
        }
        if (ch=='\n'){
            break;
        }else{
            str[i]=ch;
        }
    }
    str[i]='\0';
}

```

## 第 3 部分 C 类题目——较难

### 题目-24. 【6-4】公因数与公倍数

#### 问题描述

最大公因数(Greatest Common Divisor, 简称 GCD), 也称最大公约数、最大公因子, 指两个或多个整数共有约数中最大的一个。整数  $m$  和  $n$  的最大公约数记为  $\text{GCD}(m, n)$ 。

最小公倍数(Least Common Multiple, 简称 LCM)是指两个或多个整数共有的倍数中除了 0 以外最小的一个。整数  $m$  和  $n$  的最小公倍数记为  $\text{LCM}(m, n)$ 。

整数  $m$ 、 $n$ 、 $\text{GCD}(m, n)$  以及  $\text{LCM}(m, n)$  的关系是:

$$m \times n = \text{GCD}(m, n) \times \text{LCM}(m, n)$$

编写 C 语言程序, 从键盘读入两个整数  $m$  和  $n$ (使用空格分隔), 然后输出  $m$  和  $n$  的最大公约数和最小公倍数到屏幕。

#### 输入格式

一共 1 行数据, 包含 2 个整数  $m$  和  $n$ , 之间使用一个空格分隔。

#### 输出格式

输出 2 行, 第一行为最大公约数, 第二行为最小公倍数, 每行行末都有一个换行符。

#### 数据规模与约定

整数  $m$  的值约定为  $1 \leq m \leq 1\,000\,000\,000$ 。

整数  $n$  的值约定为  $1 \leq n \leq 1\,000\,000\,000$ 。

#### 样例输入

```
32 48
```

#### 样例输出

```
16
96
```

```
#include <stdio.h>
int main ()
{
    long long m, n, a, b, r;
    scanf("%lld %lld", &m, &n);
    if (m < n) {
        r = m; m = n; n = r;
    }
    a = m; b = n; r = a % b;
    while (r != 0) {
        a = b; b = r; r = a % b;
    }
    printf ("%lld\n", b);
    printf ("%lld", m * n / b);
}
```

## 题目-25. 【6-16】二进制数 1 的个数

### 问题描述

编写 C 语言程序，从键盘读入一个整数  $n$ ，然后统计整数  $n$  所对应的二进制数中'1'的数量，并输出到屏幕。

### 输入格式

一共 1 行数据，包含一个整数  $n$ 。

### 输出格式

输出一个整数，行末没有换行符。

### 数据规模与约定

整数  $n$  的值约定为  $0 \leq n \leq 2\,147\,483\,647$ 。

### 样例输入

12345
-------

### 样例输出

6
---

```
#include <stdio.h>
int main ()
{
    int n;
    int co=0;
    scanf("%d",&n);
    while(n>=1){
        if (n%2==1){
            co++;
        }
        n/=2;
    }
    printf ("%d",co);
}
```

## 题目-26. 【6-19】分解质因数

### 问题描述

分解质因数：任何一个合数都可以写成若干个质数(素数)相乘的形式。例如：

■  $2000 = 2 \times 2 \times 2 \times 2 \times 5 \times 5 \times 5$

■  $2005 = 5 \times 401$

■  $2010 = 2 \times 3 \times 5 \times 67$

编写C语言程序，从键盘读入一个整数  $n$ ，然后把  $n$  分解质因数，并输出到屏幕。

### 输入格式

一共 1 行数据，包含一个整数  $n$ 。

### 输出格式

输出一行信息，行末没有空格，也没有换行符：

如果  $n$  是素数，则输出  $n$ ；

如果  $n$  是合数，则按从小到大的顺序输出  $n$  的所有质因数，之间使用空格分隔。

### 数据规模与约定

整数  $n$  的值约定为  $2 \leq n \leq 100\ 000$ 。

### 样例输入

2010
------

### 样例输出

2 3 5 67
----------

```
#include <stdio.h>
int main ()
{
    int n;
    int i=2;
    scanf("%d",&n);
    while (i<n){
        if (n%i==0){
            n/=i;
            if (n>1){
                printf ("%d ",i);
                continue;
            }
            if(n==1){
                printf ("%d ",i);
            }
        }
        i++;
    }
    if(i>=n){
        printf ("%d ",n);
    }
    return 0;
}
```



## 题目-27. 【7-5】约瑟夫问题

### 问题描述

约瑟夫问题(Josephus Problem)，又称约瑟夫环：n 个人围成一圈，对其顺时针编号为 1~n，然后从第 1 个人开始顺时针方向报数，第 1 个人报数 1，第 2 个人报数 2，依次类推，凡报数k 的人则出列，接着下一个人重新从 1 开始报数，如此反复。

例如 n=8，k=3，则出列的顺序是：3、6、1、5、2、8、4、7。

编写 C 语言程序，从键盘读入两个整数 n 和 k(使用空格分隔)，然后输出出列的顺序到屏幕。

### 输入格式

一共 1 行数据，包含 2 个整数n 和 k，之间使用空格分隔。

### 输出格式

输出 n 行，每行一个整数和一个换行符。

### 数据规模与约定

整数n 的值约定为  $1 \leq n \leq 100$ 。

整数k 的值约定为  $1 \leq k \leq 100$ 。

### 样例输入

10 3
------

### 样例输出

3
6
9
2
7
1
8
5
10
4

```
#include <stdio.h>
#define N 100
int main ()
{
    int man[N]={0};
    int n;
    int k;
    int co=0;
    int out=0;
    int i=0;
    scanf ("%d %d",&n,&k);
    while (out<n){
        if (man[i]==0){
```

```

        co++;
    }
    if (co==k) {
        man[i]=1;
        out++;
        co=0;
        printf ("%d\n", i+1); //if(i!=n)printf("\n");
    }
    i=(i+1)%n;
}
return 0;
}

```

## 题目-28. 【9-5】棋盘布局判断

### 问题描述

**N 皇后问题(NQP, N-Queen Problem):** 在  $N$  行  $N$  列的国际象棋棋盘上放置  $N$  个皇后, 使其不能互相攻击。

由于国际象棋中的皇后可以在同一行、或同一列、或同一斜线(两个方向的斜线)上行走, 因此在同一行、或同一列、或同一斜线上不能放置多于一个皇后。例如, NQP 在  $N$  等于 5 时的一个解如下图所示。

	A	B	C	D	E
1				♥	
2		♥			
3					♥
4			♥		
5	♥				

我们可以使用如下所示的二维数组 `chess` 保存棋盘的布局:

```
#define N 5
int chess[N][N] = {0};    // 0: 没有放置皇后, 1: 有放置皇后
```

函数 `input_chessboard()` 的功能是从键盘输入棋盘的布局。由于在同一行中只能放置一个皇后, 因此我们可以简化输入的格式: 每一行的输入信息只需要明确指出第几列放置皇后即可。

(1) 编写 C 语言函数 `is_valid()`, 对应的函数原型如下:

```
int is_valid(int chess[][N], int n);
```

函数 `is_valid()` 的功能是对二维数组 `chess` 所对应的棋盘布局进行有效性检查: 如果放置在  $N$  行  $N$  列的国际象棋棋盘上的  $N$  个皇后, 任何两个皇后都没有互相攻击, 即  $N$  个皇后放置在  $N$  行  $N$  列的国际象棋棋盘上使得任意一行、任意一列、以及任意一斜线上都没有多于一个皇后, 那么该函数返回逻辑值“真”, 表示二维数组 `chess` 所对应的棋盘布局是  $N$  皇后问题的一个解, 否则返回逻辑值“假”。该函数不允许从键盘读取数据, 也不允许输出数据到屏幕。

```
#include <stdio.h>
#define N 5
void input_chessboard(int chess[][N], int n);
int is_valid(int chess[][N], int n);
int main()
{
    int chess[N][N] = {0};    // 0: 没有放置皇后, 1: 有放置皇后
    input_chessboard(chess, N);
    if(is_valid(chess, N)){
```

(2) main 函数调用函数 input\_chessboard ( ) 和 is\_valid ( ) 的测试代码如下:

```

        printf("yes");
    }else{
        printf("no");
    }
    return 0;
}
void input_chessboard(int chess[][N], int n)
{
    int row = 0;
    int col;
    while(row < n){
        scanf("%d", &col);
        if((col >= 1)&&(col <=
            n)){ chess[row][col - 1] =
                1; row ++;
        }
    }
}
// 你编写的代码将会嵌入到这里

```

### 输入格式

一共 1 行数据，包含 5 个整数 a、b、c、d、e，之间使用空格分隔：

整数 a 表示在棋盘的第 1 行放置皇后的列编号；

整数 b 表示在棋盘的第 2 行放置皇后的列编号；

整数 c 表示在棋盘的第 3 行放置皇后的列编号；

整数 d 表示在棋盘的第 4 行放置皇后的列编号；

整数 e 表示在棋盘的第 5 行放置皇后的列编号。

### 输出格式

无。

### 数据规模与约定

整数 a 的值约定为  $1 \leq a \leq 5$ 。

整数 b 的值约定为  $1 \leq b \leq 5$ 。

整数 c 的值约定为  $1 \leq c \leq 5$ 。

整数 d 的值约定为  $1 \leq d \leq 5$ 。

整数 e 的值约定为  $1 \leq e \leq 5$ 。

### 样例输入

```
4 2 5 3 1
```

### 样例输出

```
yes
```

```

#include <stdio.h>
#define N 5

```

```

void input_chessboard(int chess[][N],int n);
//void show_chessboard(int chess[][N],int n);
int is_valid(int chess[][N],int n);

int main ()
{
    int chess[N][N]={0};
    input_chessboard(chess,N);
    //show_chessboard(chess,N);
    if (is_valid(chess,N)){
        printf ("yes");
    }else{
        printf ("no");
    }
    //printf ("%d皇后问题的一个解。 \n",N);
    return 0;
}

void input_chessboard(int chess[][N],int n)
{
    int row=0;
    int col;
    while (row<n){
        //printf ("第%d行皇后的位置: ",row+1);
        scanf("%d",&col);
        if ((col>=1) && (col<=n)){
            chess[row][col-1]=1;
            row++;
        }
    }
}

/*
void show_chessboard(int chess[][N],int n)
{
    char queen='\3';
    int row,col;
    printf (" ");
    for (col=0;col<N;col++){
        printf ("%4c",'A'+col);
    }
    printf ("\n");
    for(row=0;row<n;row++){
        printf ("%2c",' ');
        for (col=0;col<N;col++){
            printf ("+---");
        }
        printf ("+\n");
        printf ("%d",row+1);
        for(col=0;col<N;col++){
            if(chess[row][col]==0){
                printf ("|%3c",' ');
            }else{
                printf ("|%2c",queen);
            }
        }
        printf ("|\n");
    }
}

```

```

    printf ("%2c", ' ');
    for (col=0;col<N;col++){
        printf ("+---");
    }
    printf ("\n\n");
}
*/
int is_valid(int chess[][N],int n)
{
    int row,col;
    int i;
    for(row=0;row<n;row++){
        for (col=0;col<N;col++){
            if(chess[row][col]==1){
                for (i=0;i<N;i++){
                    if((i !=col) && (chess[row][i])==1){
                        return 0;
                    }
                }
                for (i=0;i<n;i++){
                    if ((i !=row) && (chess[i][col])==1){
                        return 0;
                    }
                }
                for (i=1;i<n;i++){
                    if((row+i>=0) && (row+i<n)&&
                       (col+i>=0) && (col+i<N)){
                        if(chess[row+i][col+i]==1){
                            return 0;
                        }
                    }
                    if((row-i>=0) && (row-i<n) &&
                       (col-i>=0) && (col-i<N)){
                        if(chess[row-i][col-i]==1){
                            return 0;
                        }
                    }
                    if ((row+i>=0)&&(row+i<n)&&
                       (col-i>=0) && (col-i<N)){
                        if(chess[row+i][col-i]==i){
                            return 0;
                        }
                    }
                    if((row-i>=0) && (row-i<n)&&
                       (col+i>=0) && (col+i<N)){
                        if(chess[row-i][col+i]==1){
                        }
                    }
                }
            }
        }
    }
    return 1;
}

```

## 第 4 部分 D 类题目——课外

[说明]：为了增加考试的区分度，本类题目不统一设置题库，考试时候由命题小组自由出题。本类题目的难度原则上稍微大于 C 类题目。以下 2 道课外样题不属于题库的一部分，仅用于了解 D 类题目的难度。

### 课外样题 1. 统计进位

#### 问题描述

很多小学生在学习加法时，发现“进位”特别容易出错。你的任务是计算两个 3 位数在相加时需要多少次进位。你编制的程序应当可以连续处理多组数据，直到读到两个整数 0（这是输入结束标记）。

#### 输入格式

每一行包含两个以空格分隔的 3 位数  $m$  和  $n$ 。如果  $m=0$  且  $n=0$ ，则输入结束。

#### 输出格式

每行输出  $m$  和  $n$  相加时需要进位的次数，以及一个换行符。

#### 数据规模与约定

$m$  的值约定为  $100 \leq n \leq 999$ 。

$n$  的值约定为  $100 \leq m \leq 999$ 。

#### 样例输入

123	456
555	555
123	594
0	0

#### 样例输出

0
3
1

```
#include<stdio.h>
int main()
{
    int a,b,count,num,i;
    while(scanf("%d%d",&a,&b)!=EOF) {
        if(a==0&&b==0)
            break ;
        count=0,num=0;
        for(i=0;i<3;i++) {
            if((a%10+b%10+count)>=10) {
                count=(a%10+b%10+count)/10;
                num++;
            }
        }
    }
}
```



```
                a/=10;
                b/=10;
            }
        printf("%d\n", num);
    }
    return 0;
}
```

还是有一点细节上的问题没有解决

## 课外样题 2. 分数拆分

### 问题描述

编写程序，输入一个正整数 $m$ ，找到所有的正整数  $x$  和  $y$  (其中  $x \geq y$ )，使得  $1/m = 1/x + 1/y$ 。

### 输入格式

第 1 行包含一个整数  $n$ ，代表有  $n$  组测试数据。

接下来的  $n$  行每行包含一个正整数  $m$ 。

### 输出格式

按顺序输出对应每行的  $m$  找到所有满足条件  $1/m = 1/x + 1/y$  的组合。

如果某个  $m$  值找到多种组合，则按照  $x$  由大到小的顺序输出。

### 数据规模与约定

$n$  的值约定为  $1 \leq n \leq 10$ 。

$m$  的值约定为  $2 \leq m \leq 100$ 。

### 样例输入

```
2
3
12
```

### 样例输出

```
1/3=1/12+1/4
1/3=1/6+1/6
1/12=1/156+1/13
1/12=1/84+1/14
1/12=1/60+1/15
1/12=1/48+1/16
1/12=1/36+1/18
1/12=1/30+1/20
1/12=1/28+1/21
1/12=1/24+1/24
```

```
#include<stdio.h>
int main () {
    int k, x, y, n;
    scanf ("%d", &n);
    while (n--) {
        scanf ("%d", &k);
        for (x=(k+1); x<=2*k; x++) {
            y=(k*x)/(x-k);
            if ((k*x)%(x-k)==0) {
                y=(k*x)/(x-k);
                printf ("1/%d=1/%d+1/%d\n", k, y, x);
            }
        }
    }
}
//还是差了点意思
```