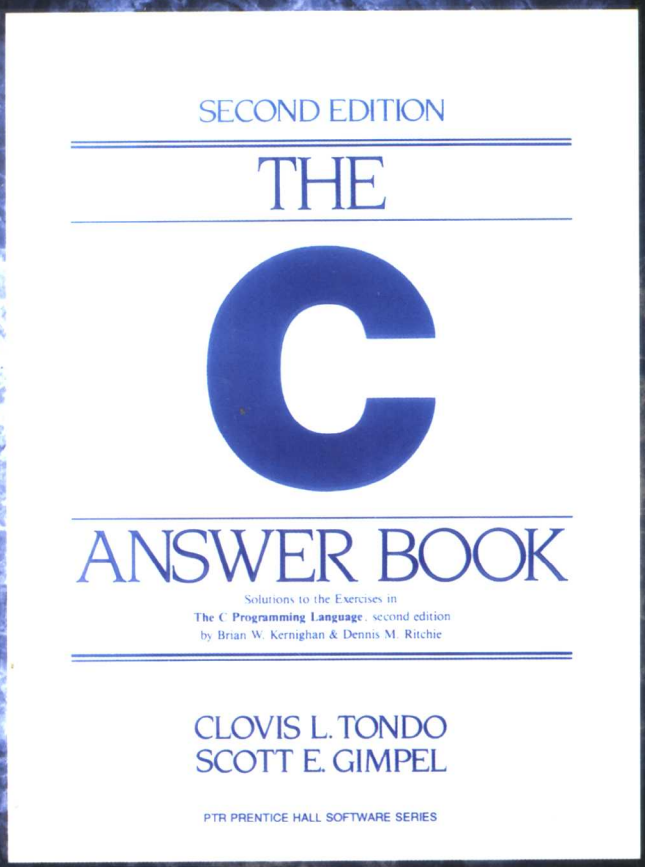
计 算 机 科 学 丛 书

原书第2版

C程序设计语言（第2版·新版）习题解答

（美）Clovis L. Tondo
Scott E. Gimpel 著 杨涛 等译



The C Answer Book
Second Edition

机械工业出版社
China Machine Press

计 算 机 科 学 丛 书

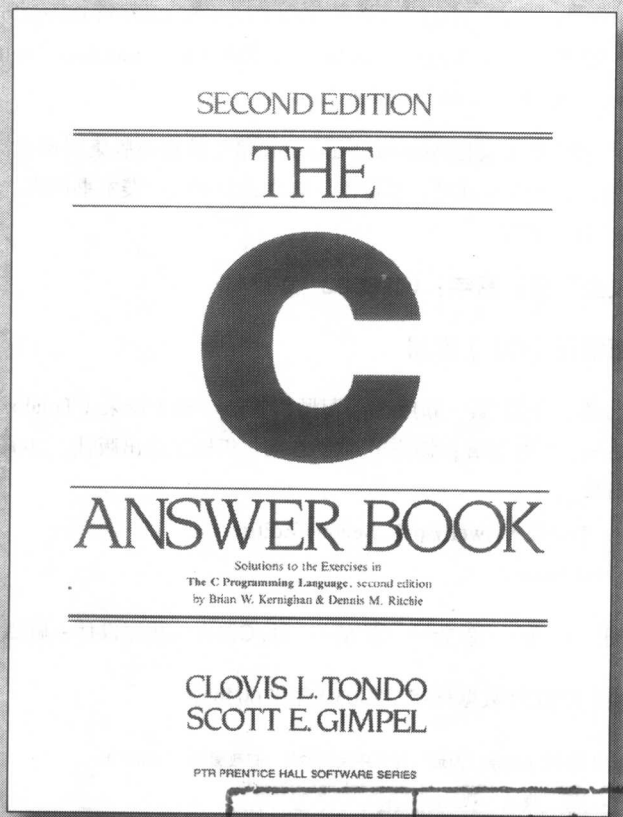
原书第2版

C程序设计语言

（第2版·新版）

习题解答

（美）Clovis L. Tondo
Scott E. Gimpel 著 杨涛 等译



The C Answer Book
Second Edition

机械工业出版社
China Machine Press

本书对Brain W. Kernighan和Dennis M. Ritchie所著的《The C Programming Language》（第2版）的所有练习题都进行了解答。K&R的原著是C语言方面的经典教材，而这本与之配套的习题解答将帮助您更加深入地理解 C语言并掌握良好的C语言编程技能。本书有关练习题都是用K&R原著中当时已经介绍过的语言结构来解答的，对每道练习题的答案要点都给予了清晰的解释，实用性强。适合于大专院校师生作为计算机专业或非计算机专业C语言教学的辅助教材，也可以作为从事计算机相关软硬件开发的技术人员的参考书。

本书版权登记号：图字：01-2003-1002

图书在版编目（CIP）数据

C程序设计语言（第2版·新版） 习题解答 /（美）汤朵（Tondo, C. L.），（美）吉米拜尔（Gimpel, S. E.）著；杨涛等译. - 北京：机械工业出版社，2004.1

（计算机科学丛书）

书名原文：The C Answer Book, Second Edition

ISBN 7-111-12943-1

Ⅰ. C… Ⅱ. ①汤… ②吉… ③杨… Ⅲ. C语言 - 程序设计 - 解题 Ⅳ. TP312-44

中国版本图书馆CIP数据核字（2003）第076046号

机械工业出版社（北京市西城区百万庄大街22号 邮政编码 100037）

责任编辑：庞 燕

北京瑞德印刷有限公司印刷 新华书店北京发行所发行

2004年1月第1版第1次印刷

787mm × 1092mm 1/16 · 9.25印张

印数：0 001-5 000册

定价：15.00元

凡购本书，如有倒页、脱页、缺页，由本社发行部调换

本社购书热线电话：（010）68326294

出版者的话

文艺复兴以降，源远流长的科学精神和逐步形成的学术规范，使西方国家在自然科学的各个领域取得了垄断性的优势；也正是这样的传统，使美国在信息技术发展的六十多年间名家辈出、独领风骚。在商业化的进程中，美国的产业界与教育界越来越紧密地结合，计算机学科中的许多泰山北斗同时身处科研和教学的最前线，由此而产生的经典科学著作，不仅擘划了研究的范畴，还揭橥了学术的源变，既遵循学术规范，又自有学者个性，其价值并不会因年月的流逝而减退。

近年，在全球信息化大潮的推动下，我国的计算机产业发展迅猛，对专业人才的需求日益迫切。这对计算机教育界和出版界都既是机遇，也是挑战；而专业教材的建设在教育战略上显得举足轻重。在我国信息技术发展时间较短、从业人员较少的现状下，美国等发达国家在其计算机科学发展的几十年间积淀的经典教材仍有许多值得借鉴之处。因此，引进一批国外优秀计算机教材将对我国计算机教育事业的发展起积极的推动作用，也是与世界接轨、建设真正的世界一流大学的必由之路。

机械工业出版社华章图文信息有限公司较早意识到“出版要为教育服务”。自1998年开始，华章公司就将工作重点放在了遴选、移译国外优秀教材上。经过几年的不懈努力，我们与Prentice Hall，Addison-Wesley，McGraw-Hill，Morgan Kaufmann等世界著名出版公司建立了良好的合作关系，从它们现有的数百种教材中甄选出Tanenbaum，Stroustrup，Kernighan，Jim Gray等大师名家的一批经典作品，以“计算机科学丛书”为总称出版，供读者学习、研究及庋藏。大理石纹理的封面，也正体现了这套丛书的品位和格调。

“计算机科学丛书”的出版工作得到了国内外学者的鼎力襄助，国内的专家不仅提供了中肯的选题指导，还不辞劳苦地担任了翻译和审校的工作；而原书的作者也相当关注其作品在中国的传播，有的还专诚为其书的中译本作序。迄今，“计算机科学丛书”已经出版了近百个品种，这些书籍在读者中树立了良好的口碑，并被许多高校采用为正式教材和参考书籍，为进一步推广与发展打下了坚实的基础。

随着学科建设的初步完善和教材改革的逐渐深化，教育界对国外计算机教材的需求和应用都步入一个新的阶段。为此，华章公司将加大引进教材的力度，在“华章教育”的总规划之下出版三个系列的计算机教材：除“计算机科学丛书”之外，对影印版的教材，则单独开辟出“经典原版书库”；同时，引进全美通行的教学辅导书“Schaum's Outlines”系列组成“全美经典学习指导系列”。为了保证这三套丛书的权威性，同时也为了更好地为学校和老师们服务，华章公司聘请了中国科学院、北京大学、清华大学、国防科技大学、复旦大学、上海交通大学、南京大学、浙江大学、中国科技大学、哈尔滨工业大学、西安交通大学、中国人民大学、北京航空航天大学、北京邮电大学、中山大学、解放军理工大学、郑州大学、湖北工学院、中国国家信息安全测评认证中心等国内重点大学和科研机构在计算机的各个领域的著名学者组成“专家指导委员会”，为我们提供选题意见和出版监督。

这三套丛书是响应教育部提出的使用外版教材的号召，为国内高校的计算机及相关专业

的教学度身订造的。其中许多教材均已为M. I. T.，Stanford，U.C. Berkeley，C. M. U. 等世界名牌大学所采用。不仅涵盖了程序设计、数据结构、操作系统、计算机体系结构、数据库、编译原理、软件工程、图形学、通信与网络、离散数学等国内大学计算机专业普遍开设的核心课程，而且各具特色——有的出自语言设计者之手、有的历经三十年而不衰、有的已被全世界的几百所高校采用。在这些圆熟通博的名师大作的指引之下，读者必将在计算机科学的宫殿中由登堂而入室。

权威的作者、经典的教材、一流的译者、严格的审校、精细的编辑，这些因素使我们的图书有了质量的保证，但我们的目标是尽善尽美，而反馈的意见正是我们达到这一终极目标的重要帮助。教材的出版只是我们的后续服务的起点。华章公司欢迎老师和读者对我们的工作提出建议或给予指正，我们的联系方法如下：

电子邮件：hzedu@hzbook.com
联系电话：（010）68995264
联系地址：北京市西城区百万庄南街1号
邮政编码：100037

专家指导委员会

（按姓氏笔画顺序）

尤晋元	王　珊	冯博琴	史忠植	史美林
石教英	吕　建	孙玉芳	吴世忠	吴时霖
张立昂	李伟琴	李师贤	李建中	杨冬青
邵维忠	陆丽娜	陆鑫达	陈向群	周伯生
周克定	周傲英	孟小峰	岳丽华	范　明
郑国梁	施伯乐	钟玉琢	唐世渭	袁崇义
高传善	梅　宏	程　旭	程时端	谢希仁
裘宗燕	戴　葵			

前言 The C Answer Book

这本习题解答对Brian W. Kernighan和Dennis M. Ritchie所著的《The C Programming Language》（第2版，Prentice Hall, 1988）（以下简称为“教材”）中所有的练习题都进行了解答。教材的中文版《C程序设计语言（第2版·新版）》已由机械工业出版社华章公司于2003年11月出版。

在美国国家标准协会（American National Standards Institute，ANSI）推出C语言的ANSI标准之后，Kernighan和Ritchie两位作者对《The C Programming Language》的第1版进行了修订，所以我们也根据ANSI标准和K&R的《The C Programming Language》（第2版）对有关习题解答进行了修订。

K&R所著的《The C Programming Language》（第2版）是C语言方面的经典教材，而这本与之配套的习题解答将帮助您更加深入地理解C语言并掌握良好的C语言编程技巧。您可以通过教材学习C语言，独立地解答书中的练习题，再钻研本书给出的习题答案。有关习题都是用教材中当时已经介绍过的语言结构来解答的，这样做的目的是为了使这本习题解答能够与教材中的教学内容保持同步。在学习到更多的C语言知识之后，相信大家能够给出更好的解决方案。例如，下面这条语句是在教材第15页介绍的：

```
if (表达式)
    语句-1
else
    语句-2
```

所以我们对出现在此之前的习题将不使用这条语句进行解答；但出现在教材第13页上的习题1-8、1-9、和1-10如果使用了这条语句，其解答将得到很大的改进。有时我们在解答中也列出使用了当时尚未介绍到的C语言知识的解决方案。

本书中的习题解答都进行了解释。我们将假设读者都已经读过了教材中有关习题出现之前的内容。我们不打算重复教材已经介绍过的内容，但会把各习题解答的要点指出来。

单凭阅读和学习其语法结构并不能真正掌握一门程序设计语言，必须进行编程实践——亲自编写一些程序并研究一些别人写的程序。我们的目标是：利用C语言良好的特性使程序模块化，充分利用库函数并以格式化的风格编写程序，这些将有助于大家清楚地了解程序的逻辑流程。我们希望这本书能够帮助大家成为C语言的高手。

我们要感谢以下朋友对本书的出版所给予的帮助：Brian Kernighan、Don Kostuch、Bruce Leung、Steve Mackey、Joan Magrabi、Julia Mistrello、Rosemary Morrissey、Andrew Nathanson、Sophie Papanikolaou、Dave Perlin、Carlos Tondo、John Wait和Eden Yount。

Clovis L. Tondo

参与本书翻译工作的人员还有杨晓云、王建桥、胡建平、张玉亭、靳蓝。

目 录 C PROGRAMMING LANGUAGE

第1章 导 言

练习1-1 （《C程序设计语言（第2版·新版）》即教材第3页）

在你自己的系统中运行“hello, world”程序。再有意去掉程序中的部分内容，看看会得到什么出错信息。

```
#include  <stdio.h>

main()
{
      printf("hello, world");
}
```

上面这个例子省略了换行符（\n），这将使光标停留在输出信息的末尾。

```
#include <stdio.h>

main()
{
      printf("hello, world\n")
}
```

第二个例子省略了printf()后面的分号。C程序的语句必须以分号结尾（参见教材第5页）。因此，对于本例，编译器将识别出少了一个分号并给出相应的出错信息。

```
#include <stdio.h>

main()
{
      printf("hello, world\n');
}
```

在第三个例子里，\n后面的双引号"被错写为单引号'。于是，这个单引号及其后面的右括号和分号将被看做是整个输出字符串的一部分。编译器将把这种情况视为一个错误，会报告说缺失了一个双引号；在右花括号前缺失了一个右圆括号；字符串过长；字符串中带有换行符。

练习1-2 （教材第3页）

做个实验，当printf函数的参数字符串中包含\c（其中c是上面的转义字符序列中未曾列出的某一个字符）时，观察一下会出现什么情况？

```
#include <stdio.h>

main()
{
      printf("hello, world\y");
      printf("hello, world\7");
      printf("hello, world\?");
}
```

参考手册（参见教材第169页）中提到："如果\后面紧跟的字符不在以上指定的字符中，则行为是未定义的。"

上面这段代码的执行结果与具体的编译器相关。一种可能出现的结果是：

```
hello, worldyhello, world<BELL>hello, world?
```

其中，<BELL>是ASCII值等于7的字符所产生的一声短蜂鸣。在\的后面，可以用最多3个八进制数字（参见教材第29页）来代表一个字符，而\7在ASCII字符集中代表的是一声短蜂鸣。

练习1-3 （教材第8页）

请修改温度转换程序，使之能在转换表的顶部打印一个标题。

```
#include <stdio.h>

/* print Fahrenheit-Celsius table
    for fahr = 0, 20, . . ., 300; floating-point version */
main()
{
      float fahr, celsius;
      int lower, upper, step;

      lower = 0;   /* lower limit of temperature table   */
      upper = 300; /* upper limit                        */
      step  = 20;  /* step size                          */

      printf("Fahr Celsius\n");
      fahr = lower;
      while (fahr <= upper) {
            celsius = (5.0/9.0) * (fahr-32.0);
            printf("%3.0f   %6.1f\n", fahr, celsius);
            fahr = fahr + step;
```

在循环语句之前增加的printf("Fahr Celsius\n");语句将在温度转换表的顶部产生一个表头。为了让输出内容与这个表头对齐，我们还在%3.0f和%6.1f之间增加了两个空格。上面这个程序中的其余语句与教材第6页中给出的代码完全一致。

练习1-4 （教材第8页）

编写一个程序打印摄氏温度转换为相应华氏温度的转换表。

```
#include <stdio.h>

/* print Celsius-Fahrenheit table
```

```
    for celsius = 0, 20, ..., 300; floating-point version  */
main()
{
     float fahr, celsius;
     int lower, upper, step;

     lower = 0;         /* lower limit of temperature table */
     upper = 300;       /* upper limit                      */
     step  = 20;        /* step size                        */

     printf("Celsius  Fahr\n");
     celsius = lower;
     while (celsius <= upper) {
          fahr = (9.0*celsius) / 5.0 + 32.0;
          printf("%3.0f  %6.1f\n", celsius, fahr);
          celsius = celsius + step;
      }
}
```

本程序将输出一个摄氏温度（0～300）到华氏温度的转换表。华氏温度是用以下语句计算得到的：

```
fahr = (9.0*celsius) / 5.0 + 32.0
```

本题的解题思路与打印华氏温度到摄氏温度的对照表程序（见教材第6页）是相同的。整型变量lower、upper、step分别对应于变量celsius的下限、上限、步长。程序先把变量celsius初始化为它的下限，再在while循环中把对应的华氏温度计算出来。然后，程序打印出这组摄氏温度和华氏温度的值，并按步长递增变量celsius的值。while循环将一直执行直到变量celsius超出其上限为止。

练习1-5 （教材第9页）

修改温度转换程序，要求以逆序（即按照从300度递减到0度的顺序）打印温度转换表。

```
#include <stdio.h>

/* print Fahrenheit-Celsius table in reverse order        */
main()
{
     int fahr;

     for (fahr = 300; fahr >= 0; fahr = fahr - 20)
          printf("%3d %6.1f\n", fahr, (5.0/9.0)*(fahr-32));
}
```

惟一的修改之处是：

```
for (fahr = 300; fahr >= 0; fahr = fahr - 20)
```

这条for语句的第一部分：

```
fahr = 300
```

负责把华氏温度变量（fahr）初始化为它的上限；for语句的第二部分（即for循环的控制条件）：

```
fahr >= 0
```

负责检查变量fahr是否大于或等于它的下限——只要这个检查的结果为真，for语句就将继续循环执行；for语句的第三部分（即步长表达式）：

```
fahr = fahr - 20
```

负责对变量fahr按步长进行递减操作。

练习1-6 （教材第11页）

验证布尔表达式getchar() != EOF的取值是0还是1。

```
#include <stdio.h>

main()
{
    int c;

    while (c = getchar() != EOF)
        printf("%d\n", c);
    printf("%d - at EOF\n", c);
}
```

根据教材第11页的论述，表达式

```
c = getchar() != EOF
```

相当于

```
c = (getchar() != EOF)
```

本程序从系统的标准输入读取字符并使用了上面的表达式。当有字符可读时，getchar()不会返回文件结束符（即EOF），所以

```
getchar() != EOF
```

的取值为真，变量c将被赋值为1。当程序遇到文件结束符时，表达式取值为假，此时，变量c将被赋值为0，程序将结束运行。

练习1-7 （教材第11页）

请编写一个打印EOF值的程序。

```
#include <stdio.h>

main()
{
    printf("EOF is %d\n", EOF);
}
```

符号常量EOF是在头文件<stdio.h>中定义的。在上面这个程序中，printf()语句中双引号外的EOF将被替换为头文件<stdio.h>中紧跟在

```
#define EOF
```

之后的文本。在我们的系统中，EOF被定义为-1，但在其他系统中，EOF可能被定义为其他

的值。这正是使用EOF等标准符号常量能够增加程序可移植性的原因所在。

练习1-8 （教材第13页）

编写一个统计空格、制表符和换行符个数的程序。

```
#include <stdio.h>

/* count blanks, tabs, and newlines                                    */
main()
{
     int c, nb, nt, nl;

     nb = 0;                          /* number of blanks        */
     nt = 0;                          /* number of tabs          */
     nl = 0;                          /* number of newlines      */
     while ((c = getchar()) !=EOF) {
          if (c == ' ')
               ++nb;
          if (c == '\t')
               ++nt;
          if (c == '\n')
               ++nl;
     }
     printf("%d %d %d\n", nb, nt, nl);
}
```

整型变量nb nt和nl分别用来统计空格、制表符和换行符的个数。这3个变量的初值都是0。

在while循环的循环体内，出现在输入中的每一个空格、制表符和换行符都将被记录。while循环中的3条if语句在每次循环中都将被执行。如果程序读到的字符不是空格、制表符或换行符，就不执行任何操作。如果程序读到的字符是这三个符号之一，就对相应的计数器加1。当while循环终止（即getchar返回EOF）时，本程序将把空格、制表符和换行符的统计结果打印出来。

对if-else语句的介绍最早出现于教材第14页，下面是使用了这一语法结构的实现方法：

```
#include <stdio.h>

/* count blanks, tabs, and newlines                                    */
main()
{
     int c, nb, nt, nl;

     nb = 0;                          /* number of blanks        */
     nt = 0;                          /* number of tabs          */
     nl = 0;                          /* number of newlines      */
     while ((c = getchar()) !=EOF)
          if (c == ' ')
               ++nb;
          else if (c == '\t')
               ++nt;
          else if (c == '\n')
               ++nl;
```

```
        printf("%d %d %d\n", nb, nt, nl);
}
```

练习1-9 （教材第13页）

编写一个将输入复制到输出的程序，并将其中连续的多个空格用一个空格代替。

```
#include <stdio.h>

#define NONBLANK 'a'

/* replace string of blanks with a single blank          */
main()
{
        int c, lastc;

        lastc = NONBLANK;
        while ((c = getchar()) != EOF) {
                if (c != ' ')
                        putchar(c);
                if (c == ' ')
                        if (lastc != ' ')
                                putchar(c);
                lastc = c;
        }
}
```

整型变量c负责记录当前输入字符的ASCII值，而整型变量lastc则记录着前一个输入字符的ASCII值。符号常量NONBLANK负责把变量lastc初始化为一个任意的非空格字符。

while循环体中的第一条if语句输出非空格字符；第二条if语句处理空格字符，而第三条if语句用于检查当前的空格字符究竟是一个单个的空格符还是一串空格中的第一个空格。最后，对变量lastc进行刷新。以上操作将一直重复到while循环终止（即getchar返回EOF）为止。

对if-else语句的介绍最早出现于教材第14页，下面是使用了这一语法结构的实现方法：

```
#include <stdio.h>

#define  NONBLANK 'a'

/* replace string of blanks with a single blank          */
main()
{
        int c, lastc;

        lastc = NONBLANK;
        while ((c = getchar()) != EOF) {
                if (c != ' ')
                        putchar(c);
                else if (lastc != ' ')
                        putchar(c);
                lastc = c;
        }
}
```

对逻辑或（OR）操作符||的介绍也最早出现于教材第14页，下面是使用了这一知识的实现方法：

```
#include <stdio.h>

#define  NONBLANK 'a'

/* replace string of blanks with a single blank              */
main()
{
      int c, lastc;

      lastc = NONBLANK;
      while ((c = getchar()) != EOF) {
           if (c != ' ' || lastc != ' ')
                putchar(c);
           lastc = c;
      }
}
```

练习1-10 （教材第13页）

编写一个将输入复制到输出的程序，并将其中的制表符替换为\t，把回退符替换为\b，把反斜杠替换为\\，这样可以将制表符和回退符以可见的方式显示出来。

```
#include <stdio.h>

/* replace tabs and backspaces with visible characters       */
main()
{
      int c;

      while ((c = getchar()) != EOF) {
           if (c == '\t')
                printf("\\t");
           if (c == '\b')
                printf("\\b");
           if (c == '\\')
                printf("\\\\");
           if (c != '\b')
                if (c != '\t')
                     if (c != '\\')
                          putchar(c);
      }
}
```

输入的字符可以是一个制表符、一个回退符、一个反斜杠或者其他任何字符。如果输入是一个制表符，我们就把它替换为\t；如果输入是一个回退符，我们就把它替换为\b；如果输入是一个反斜杠，我们就把它替换为\\；其他字符则按原样输出。

在C语言中，反斜杠是用'\\'来表示的。因此，如果我们想输出两个反斜杠，就必须把字符串"\\\\"传递给printf函数。

对if-else语句的介绍最早出现于教材第14页，下面是使用了这一语法结构的实现方法：

```
#include <stdio.h>

/* replace tabs and backspaces with visible characters      */
main()
{
     int c;

     while ((c = getchar()) !=EOF)
          if (c == '\t')
               printf("\\t");
          else if (c == '\b')
               printf("\\b");
          else if (c == '\\')
               printf("\\\\");
          else
               putchar(c);
```

练习1-11 （教材第15页）

你准备如何测试单词计数程序？如果程序中存在某种错误，那么什么样的输入最可能发现这类错误呢？

单词计数程序的测试工作首先要从没有任何输入的情况开始。此时，该程序的输出结果应该是“0 0 0”，即零行、零单词、零字符。

接下来测试输入单字符单词的情况。此时，该程序的输出结果应该是“1 1 2”，即一行、一个单词、两个字符（一个字母加上一个换行符）。

再测试一个由两个字符组成的单词。此时，该程序的输出结果应该是“1 1 3”，即一行、一个单词、三个字符（二个字母加上一个换行符）。

然后，再测试两个单字符单词的情况。首先，两个单词出现在同一行，此时的输出结果应该是“1 2 4”；然后，两个单词各占一行，此时的输出结果应该是“2 2 4”。

那些满足边界条件的输入情况最有助于发现单词计数程序中的错误。这些边界条件包括：

——没有输入

——没有单词（只有换行符）

——没有单词（只有空格、制表符和换行符）

——每个单词各占一行的情况（没有空格和制表符）

——单词出现于文本行行首的情况

——单词出现于一串空格之后的情况

练习1-12 （教材第15页）

编写一个程序，以每行一个单词的形式打印其输入。

```
#include <stdio.h>

#define IN  1         /* inside a word                          */
```

```
#define OUT 0          /* outside a word                          */

/* print input one word per line                                  */
main()
{
     int c, state;

     state = OUT;
     while ((c = getchar()) != EOF) {
          if (c == ' ' || c == '\n' || c == '\t') {
               if (state == IN) {
                    putchar('\n');      /* finish the word    */
                    state = OUT;
               }
          } else if (state == OUT) {
               state = IN;              /* beginning of word */
               putchar(c);
          } else                        /* inside a word      */
               putchar(c);
     }
}
```

整型变量state是一个布尔量，用于记录程序的处理过程是否正处于某个单词的内部。在程序刚开始运行的时候，变量state将被初始化为OUT，表明尚未处理任何数据。

第一条if语句

```
if (c == ' ' || c == '\n' || c == '\t')
```

判断变量c是否是一个单词分隔符。如果是，则第二条if语句

```
if (state == IN)
```

将判断这个单词分隔符是否表示某个单词结束。如果是，就输出一个换行符并修改变量state的值；如果不是，则不进行任何操作。

如果c不是一个单词分隔符，那么，它将或者是某单词的第一个字符、或者是一个单词中除第一个字符之外的其他字符。对于第一种情况（c是某单词的第一个字符），程序将修改变量state的值并输出这个字符；对于第二种情况（c是某个单词中的其他字符），程序直接输出这个字符。

练习1-13 （教材第17页）

编写一个程序，打印输入中单词长度的直方图。水平方向的直方图比较容易绘制，垂直方向的直方图则要困难些。

```
#include <stdio.h>

#define  MAXHIST 15          /* max length of histogram   */
#define  MAXWORD 11          /* max length of a word      */
#define  IN      1           /* inside a word             */
#define  OUT     0           /* outside a word            */

/* print horizontal histogram                                     */
main()
{
```