Turbo C 2.0

应用程序精选

马日杰　沈继军　编著

电子工业出版社

科海培训中心技术丛书

Turbo C 2.0 应用程序精选

马日杰　沈继军　编

电子工业出版社

（京）新登字 055 号

内 容 提 要

本书针对 C 语言初学者和 C 程序员经常遇到的各种各样需要处理的问题，如：文件的处理、目录的搜索和转换、改变 DOS 的不足、驻留内存、加密处理、打印处理、汉字放大、数据库处理以及绘制统计图表等等，本书就这些问题提供了较完整的 Turbo C 源程序，供大家参考和撷取。另外，对 CLIB24 点阵汉字库的结构和 Borland International 的 BGI 矢量字库的结构也一一作了说明。

本书适合于 Turbo C 程序员，亦是初学者进一步了解 Turbo C、提高编程能力的一本参考书。

Turbo C 2.0 应用程序精选

马日杰　沈继军　编

特约编辑　夏非彼

责任编辑　魏　冬

*

电子工业出版社出版(北京市万寿路)

电子工业出版社发行　各地新华书店经销

门头沟胶印厂印刷

*

开本:787×1092 毫米　1／16　印张:17.5　字数:426 千字

1993 年 10 月第 1 版　1993 年 10 月第 1 次印刷

印数：10000 册　定价：32.00 元

ISBN 7-5053-2301-6／TP・645

前　言

C 语言是美国贝尔实验室于 70 年代初期在 BCPL 及 B 语言的基础上发展起来的，但直到 70 年代后期，随着 UNIX 操作系统的普及，C 语言作为一种程序设计语言，才开始被广泛使用。近年来，很多重要软件产品都是由 C 语言来开发的。

C 语言是介于汇编语言和高级语言之间的一种语言，常常被称为中级语言。它向程序员提供了接近机器硬件和可在较低水平上使用计算机的能力。同时，它还具有高级语言书写简便、易于使用的特点。汇编语言虽然和机器码存在着一一对应关系，可以很好地控制计算机，有很高的运行效率，但是，编写汇编语言程序是一件相当繁琐的工作，而且还很容易出错，所以，现在人们一般都不用汇编语言编程了，除非不得已。而高级语言虽然易于编写和使用，编程效率高，但是高级语言与机器码不存在一一对应关系，使程序员不易于控制机器硬件，这种对程序员隐蔽了硬件的特点，使很多程序员感到不便。C 语言这种既具有高级语言的优点，又可使程序员使用控制机器硬件的种种操作的特点，使得程序员可以更快地开发出高质量的程序，所以倍受人们的欢迎。

C 语言虽然广泛流行，而且市面上有关 C 语言的书籍比比皆是，但真正谈论编程经验、实例方面的参考书却为数不多。本书特地收集、整理了数十个较为实用的 C 语言程序，如：一次删除子目录所有文件及该子目录下的子目录的所有文件、点阵汉字放大、程序加密、键盘加速、统计图等。本书除了提供源程序外，还给出较详细的程序说明及应用举例，无论对 C 语言的初学者，还是对富有经验的程序员来说，都会有所帮助的。由于本书的程序有较高的实用价值，所以既可单独使用，也可略加修改用在自己的应用程序中。

Borland International 公司的 Turbo C 2.0 由于提供了大量的库函数和许多很有特色的功能，使程序员能很好地实现自己的意图，因此它的应用较为普遍，故本书的程序均用 Turbo C 2.0 编写。但是，其中大部分的程序，读者只要略加修改，同样也可在别的 C 语言环境下获得编译通过。

本书有些程序所实现的功能，无论从 C 编程的角度上来看，还是从利用其它手段去实现的角度上来看，并不一定是最简单的，如，FINDFILE.C，我们只须利用 DOS 的“CHKDSK”和“FIND”命令的组合“CHKDSK / V | FIND”便可非常简单地实现它的功能了。另外，我们还要提醒读者，即使是世界上最大的软件公司，如 Borland International 公司，Microsoft 公司等，对它们自己的产品也不是百分之百的满意，关于这点，从它们不断地更新版本就可以得到证明。所以，我们希望读者不要把心思费在研究本书的这些程序如何如何上去，而是从中吸取对自己有用的养分，尽快地迈进 C 语言这个神秘的门槛。本书的目的在于帮助读者掌握 C 语言，或者作为读者开发程序的参考书。

编者

1993 年 9 月

目　录

第一章　文件

DOS 的文件，是 DOS 操作系统中的一种信息集合，与之相联系的有：文件名，文件扩展名，长度，最后一次更新日期和时间，属性以及内容。文件名是每个信息集合的名称，它与录音带上的乐曲名称没什么两样。唯一不同的地方是前者被写在磁盘上，后者被印在纸上。文件扩展名是为了让操作系统和用户区分文件的类型而设的，通常，DOS 操作系统认为扩展名为″.EXE″的文件是可执行文件，它含有可执行机器代码。扩展名为″.COM″的文件是命令文件，它也含有可执行机器代码。这种文件与″.EXE″文件的区别是格式不同，一般说来，″.EXE″文件是较大型的程序文件，而″.COM″文件是较小型的程序文件。扩展名为″.BAT″的文件是批处理文件，也是可以执行的。但是，它是由一系列的命令行组成的，这些命令行与我们在操作系统提示符(例如：″C:＼＞″)下　键入的命令行没有多大区别，批处理文件中的命令行，由 DOS 操作系统命令解释器逐行解释执行，这个解释执行的过程当中无须用户干预，其效果与用户一行行键入并执行相似，这就是这种文件之所以被叫做批处理文件的原因。除了这几种扩展名是 DOS 操作系统特别识别以外，其余均是由用户自己或其它程序系统指定和识别的，例如，可以给书信文件预扩展名″.TXT″。文件长度是信息集合的大小，在此，需要指出的是，文件的大小，并不等于它所占的磁盘空间的大小，操作系统给文件分配空间的原则是，以磁盘分配单元为单位，对于不够一个整磁盘分配单元的文件内容，也要分配一个单元的空间，一般的硬盘磁盘分配单元的大小是 2K。由此可知，文件所占的磁盘空间往往要比它的实际大小大一些。最后更新的日期和时间是文件最后一次被写入的日期和时间。文件的属性有若干种，系统属性(操作系统专用属性)，档案属性(即一般的•文件)，目录属性(该文件名是子目录名)，只读属性(该文件只能读，不能写)，隐含属性(该文件在列表时，是看不到的)和卷标属性(只存在于根目录，是磁盘的卷标名)。一个文件可以同时具有若干种属性，如可同时具有隐含、只读和档案属性。但有些属性是不相容的，如卷标属性和目录属性。文件内容就是信息集合的内容，通常可以根据文件的内容把文件分成两种，一种称为文本文件，另一种称为二进制文件。文本文件是由字母，数字，符号，汉字和回车换行符组成的。人们一般能看得懂，不是文本文件的文件一般被称为二进制文件。

文件处理，是每个程序员都会遇到的问题。在这里给出一些如何利用 C 语言处理文件的例子，以帮助读者掌握处理文件的方法。

Ⅰ 显示文件长度 FILESIZE.C

功　能

显示指定文件的长度，并显示文件的总数，指出磁盘的总容量及可用(即未用)容量。filesize 程序能让你知道所指定的文件一共占用的磁盘空间，据此可计算出是否可拷贝到一片软磁盘中。

格　式

filesize [filename.ext]

其中，filename.ext 为指定的文件名，可使用"*"、"?"字符，还可含有驱动器符和路径名。方括号括起来的项目是可选择的。下同。

说　明

filesize 使用了 Turbo C 的库函数 findfirst、findnext 和 getdfree。在命令行中指定的文件名参数，被传给了 findfirst 函数及 findnext 函数，此二函数被重复调用以找出所有符合条件的文件的文件名及其长度，并计算出所有文件的总长度。然后通过调用 getdfree 函数，以得到指定磁盘的容量和磁盘中可用空间大小。　本程序可找出隐含文件，若想忽略隐藏文件和系统文件，可将传给 findfirst 最后的参数值 FA_RDONLY、FA_HIDDEN、FA_SYSTEM、FA_ARCH 改成 FA_RDONLY | FA_ARCH。

getdfree 函数所要求传给它的参数 driveno，0 代表当前盘，1 代表 A 盘，2 代表 B 盘等。fnsplit 函数仅用于取得磁盘代号，如果该磁盘的代号存在，则将其值减去 64，这是因为字母 A 的 ASCII 值是 65，减去 64 为 1，所以，任何正确的磁盘代号减去 64 都可得到 getdfree 函数所需要的磁盘代号值。如果不指定磁盘代号，则变量 driveno 不会改变其初始值 0。即当前盘。

findfirst 和 findnext 函数，是用来查找匹配文件信息的。第一次必须调用 findfirst，以后每次必须调用 findnext。这是因为 DOS 操作系统就是这样规定的。有一点需要注意的是，若在调用了若干次 findnext 后，又重新调用 findfirst，则所得到的匹配文件将又是上次调用 findfirst 所得到的那个文件。

举　例

1. C>FILESIZE *.*
   显示C盘当前目录中所有文件的长度、文件总数及所占用的磁盘空间。由于没有指定磁盘和路径，所以查找文件的磁盘为当前盘 C，路径为 C 盘中的当前路径。
2. C>FILESIZE D:*.TXT
   找出D盘根目录下所有后缀为".TXT"的文件，并给出全部匹配文件所占用的磁盘空间。
3. C>FILESIZE
   显示帮助信息。

程序清单 FILESIZE.C

```
#include <stdio.h>
#include <stdlib.h>
#include <dir.h>
#include <string.h>
#include <dos.h>
void filesize(void);
char searchstr[MAXPATH],fsize[11],ttotal[11],dtotal[11],
```

```
        davail[11],drive[MAXDRIVE],subdir[MAXDIR],
        file[MAXFILE],ext[MAXEXT];
int   flag, done, count, driveno;
unsigned long percl, avail, btotal, total;
struct ffblk dta;
struct dfree frespace;

main(int argc, char * argv[])
{
  if(argc > 1)
    {
     strcpy(searchstr, strupr(argv[1]));
     flag = fnsplit(searchstr, drive, subdir, file, ext);
     if (flag & DRIVE)
        driveno = drive[0] - 64 ;
    } else {
     printf("Usage: filesize filename.ext\n");
     exit(1);
    }
  filesize( );
  return 0;
}

void filesize(void)
{
  done = findfirst(searchstr, &dta, FA_RDONLY | FA_HIDDEN | \
         FA_SYSTEM | FA_ARCH);
  if (done)
    printf("\nNo files match the parameter.\n");
  while (!done)
   {
    count++;
    ultoa(dta.ff_fsize,fsize,10);
    printf("\n %-12s              %10s",dta.ff_name,fsize);
    total += dta.ff_fsize;
    done = findnext(&dta);
   }
  if (count != 0)
   {
    ultoa(total,ttotal,10);
    printf("\n\n Total of %3d files is : %10s bytes\n",\
      count,ttotal);
   }
  getdfree(driveno,&_frespace);
  percl = frespace.df_sclus * frespace.df_bsec;
  avail = frespace.df_avail * percl;
  btotal = frespace.df_total * percl;

  ultoa(avail,davail,10);
  ultoa(btotal, dtotal,10);
```

在win16模式下：

```
struct ffblk {
    char ff_reserved[21];
    char ff_attrib ;
    int ff_ftime ;
    int ff_fdate ;
    long ff_fsize;
    char ff_name[13];
    };

struct dfree {
    unsigned df_avail ;
    unsigned df_total ;
    unsigned df_bsec ;
    unsigned df_sclus;
    };
```

```
    printf("\nDrive usage:\n");
    printf("      Total space on drive: %10s bytes\n",\
      dtotal);
    printf("           Space not yet used: %10s bytes\n\n",\
      davail);
}
```

Ⅱ 查看文件尾内容 TAIL.C

功 能

查看文件尾的内容。有些文件占用磁盘的空间是很大的，如，汉字字模定义文件，汉语词组输入对照表，某些计算程序产生的结果数据文件。它们的大小一般都有几百 K 到数兆，如果用 DOS 内部命令 TYPE 去浏览其中的一部分，也许等上几个小时 也等不到所要查看的内容出现，利用 tail 去解决这些问题是非常有效的，它可以迅速定位到文件某行，然后输出文件该行开始以后的内容，unix 操作系统中有一个功能类似的命令，tail 的功能基本上和它差不多。注意 tail 只适用于浏览文本文件，利用它浏览二进制文件，结果是不可知的。

格 式

tail [-n] [filename.ext]

其中， n 是从文件尾开始往回数的行数。n 之前的"-"为选择项起始符，如果为了适应 DOS 操作系统的习惯，可以在程序中把"-"改为"/"，在这里用"-"而不用"/"的原因是为了适应 C 语言或 unix 操作系统的习惯。filename.ext 是文件名，文件名中，可以含有盘号和路径，但不可以含有"*"和"?"。n 的缺省值为 20。

说 明

程序 tail 是利用 Turbo C 的库函数 fseek 来定位文件的读操作位置的。fseek 的最后一个参数有三种选择，选择 SEEK_SET 则从文件头开始移动读操作位置指针，选择 SEEK_CUR 则从当前读操作位置开始移动指针，选择 SEEK_END 则从文件尾开始移动指针，在函数 tail 开始，选择 SEEK_END 来把文件读指针定位于文件尾，并用 ftell 来获得文件当前读操作位置。该位置事实上就是文件的大小，然后 tail 根据文件大小及未用内存(即 coreleft 的返回值)，用 malloc 申请一块大小(即 bufsz)适当的内存，作为读缓冲区(即 buf)，接着用 fread 函数从文件尾开始往前读，每次读 bufsz 字节到缓冲区 buf 中，从 buf 的尾部往前计算换行符(即'\n')的个数，若达到规定的行数(即换行符的个数)，则输出位置便确定了，否则再读文件。每次读文件，都要检查剩余未读部分是否小于 bufsz，是则对读入的字符数作适当的调整。最后把文件指针定位到满足要求的位置，并输出该位置及其以后的内容。tail 在输出时，调用了函数 putchar，该函数在输出换行符时，把换行符转换为回车符和换行符，所以文件中原有的回车符就不再输出了。

举 例

1. C>TAIL TAIL.C

显示文件TAIL.C最后20行的内容。

2. C>TAIL −10 TAIL.C

显示文件TAIL.C最后10行的内容。

3. C>TAIL −1050 TAIL.C

显示文件TAIL.C最后1050行的内容。

4. C>TAIL −20

显示帮助消息。

5. C>TAIL

显示帮助信息。

程序清单 TAIL.C

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

int tail(char * ,unsigned long);

int main(int argc, char * * argv)
{
unsigned long lines = 20L;
int i,tmp = 0;

  if(argc = = 1)
    tmp = 1;
    else if(argv[1][0] = = '−') {
    lines = atol(argv[1] + 1);
    if(lines = = 0L)
      lines = 20L;
    if(argc < 3)
      tmp = 1;
      else
      i = 2;
    } else
    i = 1;

  if(tmp = = 1)
    {
    printf("Usage: tail [−n] filename.ext\n");
    printf("\t−n: n is the number of lines which is expected \n");
    printf("\t      to output from the tail of the file.\n");
    exit(1);
    }

  switch(tmp=tail(argv[i],lines))
    {
    case 1:
```

```
            fprintf(stderr,"Error---can't open %s.\n",argv[i]);
            break;
          case 2:
            fprintf(stderr,"Error---no sufficient memory available.\n");
            break;
          }
        if(tmp)
          exit(1);
        return 0;
}

int tail(char * filename,unsigned long lines)
{

unsigned long lc = 0L,curpos;
char ch, * buf;
FILE * fp;
unsigned int bufsz,i = 0;
        if((fp = fopen(filename,"rb")) == NULL)
          return 1;
        fseek(fp,0L,SEEK_END);
        if((curpos = ftell(fp)) == 0L)
          {
          fclose(fp);
          return 0;
          }
        bufsz = coreleft();
        if(curpos < (long)bufsz)
          bufsz = (unsigned int)curpos;
        for(;bufsz > 0; bufsz--)
          if((buf = (char * )malloc(bufsz)) != NULL)
            break;
        if(buf == NULL)
          {
          fclose(fp);
          return 2;
          }
        while((curpos > 0L) && (lc <= lines))
          {
          if(curpos > (unsigned long)bufsz)
            curpos = curpos - (unsigned long)bufsz;
            else {
            bufsz = (unsigned int)curpos;
            curpos = 0L;
            }
          fseek(fp,curpos,SEEK_SET);
          fread(buf,bufsz,1,fp);
          for(i = bufsz;
i > 0;--i)
            {
            if( * (buf + i - 1) == '\n')
```

```
                lc++;
            if(lc > lines)
              break;
            }
        }
      if(lc < = lines)
        fseek(fp,0L,SEEK_SET);
        else
        fseek(fp,curpos + (unsigned long)i,SEEK_SET);
      while(!feof(fp))
        {
        if((ch = fgetc(fp)) = = EOF)
          break;
        if(ch != '\r')
          putchar(ch);
        }
      fclose(fp);
      free(buf);
      return 0;
}
```

Ⅲ 统计文件行、字和字符数 WC.C

功 能

统计文件行、字(包括汉字)和字符数，对于搞编辑排版的用户，可利用 wc 程序来统计行、字和字符数，对于以一行为一条记录的数据文件，可利用它来统计记录数。读者也许还可以想到更多的用途。unix 操作系统也有一个功能类似的命令，wc 程序的功能也基本上和它相同。注意 wc 也只适用于文本文件。

格 式

wc [options] [filename.ext]

其中 filename.ext 是文件名，可含有盘号和路径，可使用通配符"*"和"?"。options 是选择项，一共有三个选择项。"-l"为统计行数，"-w"为统计字数(英文字和汉字)，"-c"为统计字符数。选择项被省略时，将统计行、字和字符数。

说 明

程序首先检查命令行参数中有无选择项，即看每个参数的第一个字符是否为"-"，是则看第二个字符是否为"L"、"W"或"C"，是则记录下来，其中，把第二个字符和"0x5f"相与的目的为将小写字母转换为大写字母，因为大写字母 A~Z 的 ASCII 码值为 0x41~0x5b，小写字母 a~z 的 ASCII 码为 0X61~0X7b，把 0x61~0x7b 分别和 0x5f 相与的结果正好分别是 0x41~0x5b。这是把小写字母转换为大写字母的一种简便方法。接着利用 Turbo C 的库函数 findfirst 和 findnext 去查找匹配文件，有匹配的则打开之，并统计该文件的行、字和字符数。然后根据要求输出统计结果。在统计字的函数 wordcount 中，对于英文字则以一首之前和尾之后不是字母的串为一个字。根据中华人民共和国国家

标准 GB2312-80，可知连续两个编码在 161~254 之间的字符组为一个汉字。所以，一个汉字为连续两个编码在 161~254 之间的字符组。字符类型函数 chartype 的返回值有三个，ALPHA 表示传给该函数的字符是字母，HANZI 表示传给该函数的字符在编码区 161~254 之间，OTHER 则表示其它。在统计字符的函数 charcount 中，不统计小于 0x20 的字符，这是因为小于 0x20(空格的 ASCII 码值)的字符是有特殊用途的字符，例如，回车符、换行符、文件结束符以及通讯呼叫符等。

举 例

1. C>WC *.C *.TXT
   统计每个以".C"和".TXT"为扩展名的文件的行、字和字符数。
2. C>WC -L RESULT.DAT
   统计RESULT.DAT的行数。
3. C>WC -L -W DATABASE.DAT
   统计DATABASE.DAT的行数和字数。
4. C>WC
   显示帮助信息。
5. C>WC -W -C
   显示帮助信息。

程序清单 WC.C

```
#include <stdlib.h>
#include <stdio.h>
#include <dir.h>
#include <dos.h>
#include <string.h>
#include <ctype.h>

#define MAXLEN 2048
#define CHARS 1
#define WORDS 2
#define LINES 4

int chartype(unsigned char);
unsigned int charcount(unsigned char *);
unsigned int wordcount(unsigned char *);
unsigned int linecount(void);

int main(int argc, char * * argv)
{
unsigned long cs,ws,ls;
int cmd = 0;
int i,j,done,flag;
unsigned char buffer[MAXLEN];
struct ffblk ffblk;
char driver[MAXDRIVE],dir[MAXDIR],file[MAXFILE],ext[MAXEXT];
```

```
char path[MAXPATH],filename[MAXPATH];
FILE * fp;

  for(i = 1; i < argc && argv[i][0] = = '-'; ++i)
    {
    if((argv[i][1] & 0x5f) = = 'L') cmd = cmd | LINES;
    if((argv[i][1] & 0x5f) = = 'W') cmd = cmd | WORDS;
    if((argv[i][1] & 0x5f) = = 'C') cmd = cmd | CHARS;
    }
  if(cmd = = 0)
    cmd = LINES | WORDS | CHARS;

  if(argc < = i)
    {
    printf("Usage: wc [options] [filename.ext]\n");
    printf("Options:\n");
    printf("\t-l: count lines.\n");
    printf("\t-w: count words.\n");
    printf("\t-c: count characters.\n");
    exit(1);
    }

  for(; i<argc; ++i)
    {
    fnsplit(argv[i],driver,dir,file,ext);
    fnmerge(path,driver,dir,"","");
    done = findfirst(argv[i], & ffblk,0);
    if(done)
      printf("Error—no file matching %s.\n",argv[i]);
    while (!done)
      {
      strcpy(filename,strupr(path));
      strcat(filename,ffblk.ff_name);
      if((fp=fopen(filename,"r"))==NULL)
            printf("Error—can't open %s.\n",filename);
        else {
            cs = ws = ls =0L;
            while(!feof(fp))
              {
              if(fgets(buffer,sizeof(buffer),fp) = = NULL) break;
              if(cmd & LINES) ls = ls + (unsigned long)linecount();
              if(cmd & WORDS) ws = ws + (unsigned long)wordcount(buffer);
              if(cmd & CHARS) cs = cs + (unsigned long)charcount(buffer);
              }
            fclose(fp);
            printf("\n%s:\n",filename);
            if(cmd & CHARS) printf("\tCharacters: %ld\n",cs);
            if(cmd & WORDS) printf("\tWords: %ld\n",ws);
            if(cmd & LINES) printf("\tLines: %ld\n",ls);
            }
      done = findnext(& ffblk);
```

```
        }
      }
    return 0;
}

unsigned int charcount(unsigned char * string)
{
int chs = 0;
unsigned char * p;

    for(p = string; * p != 0; ++p)
      if( * p > 0x20)
        chs = chs + 1;
    return chs;
}

unsigned int linecount( void )
{
    return 1;
}

#define ALPHA 0
#define HANZI 1
#define OTHER 2

int chartype(unsigned char ch)
{
    if(ch > 160 && ch < 255)
      return HANZI;
    return isalpha(ch)? ALPHA: OTHER;
}

unsigned int wordcount(unsigned char * string)
{
int type,prevtype = OTHER;
unsigned int wds = 0;
unsigned char * p;
    for(p = string; * p != 0; ++p)
      {
      type = chartype( * p);
      if(prevtype = = ALPHA && type != ALPHA)
        wds++;
      if(prevtype = = HANZI && type = = HANZI)
        {
        wds++;
        prevtype = OTHER;
        } else
        prevtype = type;
      }
    return wds;
}
```

Ⅳ 文件分解 SPLIT.C

功 能

把文件分解成若干个较小的文件。分解可以按行，也可以按大小进行。一个文件如果大小有几百 KB 或者几兆，一般的文件编辑器是编辑不了的,这时可以首先利用 split 把文件分解成若干个较小的文件，然后编辑这些小文件，编辑好了以后，再用 DOS 的 COPY 命令把它们合并成一个。另外，在用 DOS 的 RESTORE 命令恢复以前用 BACKUP 命令备份大文件时，经常会遇到 DOS 的版本问题。如果用 split 去备份大文件是不会有这种问题发生的，方法很简单，首先根据软盘的容量(360KB 盘为 362496Bytes，1.2MB 盘为 1213952Bytes。)，把大文件分解成若干个较小的文件，然后直接用 COPY 命令备份即可，在恢复时也是用 COPY 命令，由于 DOS 的 COPY 命令不存在版本问题，所以，使用这种方法，是不会有版本问题的。

格 式

split [options] [source [targetplate]]

其中，source 为被分解文件的文件名，可含有盘号和路径，不可以使用通配符″*″和″?″。 targetplate 为分解后，生成的较小文件的文件名样板，文件名样板最多可给六个合法的 DOS 文件名字符，假设样板为″SMALL″，生成的小文件文件名一般为″SMALLAA.AAA″，″SMALLAA.AAB″，..... 等等，若盘中已有一个名为″SMALLAA.AAA″的文件，则小文件文件名为″SMALLAA.AAB″，″SMALLAA.AAC″，..... 等等。options 是选择项，一共有两个选择项。″-s″为要求按大小分解，生成文件的最大容量应跟在″-s″后给出，例如，要求生成文件的最大容量为 362496，那么该选项应为″-s362496″。″-l″为要求按行分解，生成文件最多行数应跟在″-l″后给出，例如，要求生成文件的最多行数为 1000，那么该选择项应该为″-l1000″。

说 明

程序首先检查命令行参数中有无选择项，即看第一个参数的第一个字符是否为″-″，是则看第二个字符是否为″S″或″s″、″L″或″l″，是则记录下来，并根据该参数调整生成文件的大小或行数的缺省值。如果没有给出被分解文件名和生成文件的文件名样板，则提示用法。否则把它们按要求，传给按行数或按大小分解文件的函数。在分解函数 szsplit 和 lnsplit 中，首先是用 Turbo C 的库函数 mktemp 和传来的文件名样板，构造一个与盘上已有文件不同名的文件名作为生成的目标文件名，并打开它。然后根据要求(按大小或按行数)读入作为源的文件，将读入的内容写到刚才打开的目标文件中。如果达到要求(大小或行数达到了指定值)，则关闭目标文件，并重新构造新的唯一文件名，再重复刚才的过程，直到源文件被读完。

举 例

1. C>SPLIT -S362496 C:\HANZIXT\CCLIB SCLIB