增强 DOS 的 DIR 命令功能

宁波市机械工业局计算机室　朱孟海

众所周知,DIR 命令向用户提供了列磁盘中目录、查看盘片中储存的文件和按某种条件找出满足给定要求的一类文件的功能,是一个十分有用的 DOS 内部命令。

DIR 命令只能查看磁盘中当前目录或指定目录下与给定条件相匹配的一类文件,而没有整个磁盘的搜寻能力,并且 DIR 命令不能列出属性为隐式或系统的一类文件,这些问题的存在给用户在使用 DIR 命令时造成种种不便。

比如用户在只知道当前磁盘中存放着某一文件,而不知道其具体存在哪个目录,如果使用 DIR 命令来查找,在最坏的情况下可能得查遍盘中所有的子目录,浪费了大量的时间,并增加了工作量;为了提高磁盘空间的利用率,更好地管理磁盘,我们得常常做磁盘的维护工作,在维护中可能会发现同版程序在盘中的不同目录下有好几个备份,到底有多少备份和各自存放在那些子目录中不清楚,如果使用 DIR 命令所存在来查看,会发生上述同样的情况。

为解决 DIR 命令所存在的上述问题,笔用者采 C 语言设计了一个与 DIR 功能相似的程序 XDIR.c,通过调试、执行后证实该程序达到原先设想的要求,其覆盖了 DIR 命令功能,能够列出磁盘中的所有文件和子目录、能按指定的条件进行整盘搜寻,其执行过程中所显示的目录形式也与 DIR 相似。如果把该程序稍作修改,即可成为删除当前盘或指定盘中满足给定条件的一类文件的实用程序。

本程序在 TURBO C 2.0 版下编译,在 IBM PC／XT 及其兼容机上调试通过。

说明:程序中的引用宏 MAX　LEVEL 表示磁盘中目录的层级数,如果实际目录级数的值比该宏的值还要大,请修改该宏的值,再重新编译即可。

附录:XDIR.c 源程序。

```
define MAX  LEVEL 8
#include "ctype.h"
#include "dir.h"
#include "dos.h"
#include "stdio.h"
struct FDATE
  {
   unsigned day: 5;
   unsigned month: 4;
   unsigned year: 7;
  };
struct FTIME
  {
   unsigned second: 5;
   unsigned minute: 6;
   unsigned hour: 5;
  };
union DATE
  {
   unsigned int temp;
   struct FDATE fdate;
   struct FTIME ftime;
  } date  time;
char root[80];
int numbers  of  file,dir  tree;
long bytes  per  cluster,file  size,whole  size;
struct ffblk fcb[MAX  LEVEL];
void search  directory(char  * );
main(int arge,char  * argv[])
{
 int cur  driver,save  driver,whole  driver,flag,index;
 char save  dir[80],cur  dir[80],path[80];
 struct dfree disk  space;
```

```
    e,whole space;
    flag=0
    numbers of file=0:
    dir tree=0;
    file size=01;
    whole size=01;
    printf("查找磁盘中所有与给定条件相匹配的一类文件
        的实用程序 XDIR.exe \n");
  if(argc==1)
   {
    printf("\n\n 格式:XDIR [d:]filename[.ext]\n");
    printf("示例:XDIR *.bak (列当前盘中以.bak 为后缀
的所有文件)\n");
    exit(1);
   }
  save driver=getdisk();
  getcwd(save dir,80);
  whole)driver=setdisk(save driver):
  if(argv[1][1]==':')
   {
    cur driver=toupper(argv[1][0])-'A';
    if(cur driver>whole driver)
     {
      printf("\n\n 命令行中有无效驱动器符
          :%c\07",cur driver+'A');
      exit(1);
     }
    flag=1;
    setdisk(cur driver);
    getcwd(cur dir,80);
   }
  /*进入当前驱动器的根目录*/
  strcpy(root,"\\");
  chdir(root);
  for (index = strlen (argv[1])-1; index>=0 && argv[1]
    [index]!='.'; --index);
  strcpy(path,argv[1]);
  if(index<0) strcat(path,'*');
getdfree(0,disk space);
  free space = (long) disk space . df avail * (long) disk
      space. df bsec * (long) disk space. df sclus;
  whole space = (long) disk space.df total * (long) disk
      space.df bsec * (long) disk space.df sclus;
  bytes per cluster = (long) disk space.df bsec *
      (long)disk space.df sclus;
  search directory(path);
  if(numbers of file==0) printf("\n\n 文件未发现
!\07");
  else
   {
    printf("\n\n%d 个文件共%8ld个字节,占盘空间
      %8ld 个字节
    ",numbers of file size,whole size);
    printf("  ,可压缩率%2d%, 100 - (int) (100.0 *
      (double) file size / (double) whole size));
    printf("\n 盘片中总共有%8ld个字节, 还剩%8ld个
      字节可用", whole space , free space);
    printf("  ,可使用率%2d%", (int) (100.0 * (double)
      free space / (double) whole space));
   }
  if(flag==1)
   {
    chdir(cur dir);
    setdisk(save driver);
   }
  chdir(save dir);
  exit(0);
  ]
  void Search directory(char *filename)
  {
  char sub directory[80],fname[9],fext[5];
  unsigned year,month,day,hour,minute,second;
  int done,length,index,loop;
  int display=-1;
  lont space-size;
  /*列出当前目录下与指定条件匹配的文件*/
  done=findfirst(filename,&fcb[dir tree],0xff);
  while(!done)
```

```
{
 if (( fcb [dir tree]. ff attrib & FA DIREC)! = FA
DIREC)
  {
   if(dir tree!=display)
    {
     getcwd(sub directory,80);
     printf("\n\n%s 目录",sub directory);
     display=dir tree;
    }
   for(index=0; index<8 & & fcb [dir tree].ff
     name [index]! ='.'& & fcb [dir tree].ff name
     [index]! = 0:++ index)
    fname[index]=fcb[dir tree].ff name[index];
   fname[index]=0;
   for(loop = index; loop-index<4 & &
     fcb[dir-tree].ff name [loop] ! = 0;++loop)
    fext[loop-index]=0;
   date time.temp = fcb[dir tree].ff fdate; / * 取文
     件最后修改时间 * /
   year=date time.fdate.year+80;
   month=date time.fdate.month;
   day=date time.fdate.day;
   date time.temp = fcb[dir-tree], ff ftime; / * 取文
     件最后修改时间 * /
   hour=date time.ftime,minute;
   second=date time.ftime.second * 2;
   printf ("\n %-8s%-42 %8ld %2d- %02d-
     %02d %2d: %02d: %02d", fname, fext,
     fcb[dir tree].ff fsize,
   year,month,day,hour,minute,second);
   file size=fcb[dir tree0.ff fsize / bytes per cluster

   if ( space size * bytes per cluster = = fcb [dir
    tree], ff fsize) whole size = whole xize + fcb [dir
    tree]. ff fsize;
   else whole size = whole size + (space size+11) *
    bytes per cluster;
   ++numbers of file;
  }
 done=findnext(&fcb[dir tree]);
}
/ * 搜索当前目录以下的子目录 * /
done=findirst("*.*",fcb[dir tree],0xff);
while(!done)
 {
  if( (fcb [dir tree]. ff attrib & FA DIREC ) = = FA
   DIREC & & fcb [dir tree]. ff name [0]! = '.')
   {
    getcwd(root,80);
    if(root[strlen(root)-1]!='\\')strcat(root,"\\");
    strcat)root,fcb[dir rtee].ff name);
    if(root [strlen (root)-1) = = '\\' & & strlen
     (root)>3) root [strlen (root(-1]=0;
    chdir(root);
    ++dir tree;
    search directory(filename);
    getcwd(root,80);
    length=strlen(root)-1;
    if(root[length][= ='\\')-length;
    while(length>0 &&root[length]!='\\')-length;
    root[length+1]=0
    if (root [strlen (root)-1 = = '\\' & & strlen
     (root)>3 root [strlen (root)-1]=0
    chdir(root);
    --dir tree;
   }
  done=findnext(&fcb[dir tree];
 }
return;
}
```