为 DOS 增加目录树删除功能

朱孟海　（宁波机械局计算机室）

DOS 是运行在 IBM PC／XT、AT 及其兼容机上的一种单用户单任务操作系统，它对磁盘上的文件目录管理采用树状结构，它允许在每个目录中存放有限个文件或下级子目录，我们把某个目录下由文件及下级子目录组成的这种形式称为目录树。

使用过 WINDOWS 操作系统的人都知道，在 WINDOWS 下要删除一个目录树是非常方便的，只要操作者选择该功能，给出要删除的目录名系统就把该目录连同其下的文件及子目录统统删除掉。而在 DOS(从 1.0 至 6.2 版)操作系统下，系统即没有向我们提供删除一个目录树的功能，以往我们要删除一个目录一般要经过以下几步：先删除该目录的所有下级子目录下的文件；再删这些下级子目录；删除指定目录下的文件；把工作目录移到本目录的上级目录；最后删除指定目录。在这些步骤中要用到 DOS 的 DEL、CD、RD 等命令，可见要删除一个 DOS 的目录树是十分费时的，也存在误操作的可能。

为了能方便、快速地删除 DOS 的目录树，减少操作中的失误，笔者采用 C 语言编写了一个 DOS 目录树删除功能的程序 DDT.C，经编译、链接后得到一个可执行实用软件 DDT.EXE，使用它可以十分方便、快速地删除一个指定的目录树。

例：假定在硬盘 C:中有如下的目录结构。

```
C:\———DOS——— EXE
              — COM
   —BIN——— EXE
              — COM
   —USER—PAS—PRG—BAS
```

那么命令：DDT C:\DOS

表示删除 C:\DOS 指定的目录树，执行后盘中目录形式如下：

```
C:\———BIN——— EXE
              —COM
   —USER—PAS—PRG—BAS
```

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。注意：程序中的引用宏 MAXLEVEL 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，再重新编译链接即可。

源程序 DDT.C

```
# define MAXLEVEL 6
# include "ctype.h"
# include "dir.h"
# include "dos.h"
# include "stdio.h"
# include "string.h"
char boot [80]
char root [80]
int nrmbers;
int tree;
struct ffblk fcb[MAXLEVEL];
void serach  directory(char * );
main(int argc,char  * argv[])
{
 int save  driver;
 char save  dir[80];
 int current  driver;
 char current  dir[80];
 char path[80];
 int counter;
 int index;
 numbers = 0;
 tree = 0;
 printf("DDT—Delete Directory Tree,Version 1.00
 Copyright 1994.8 by Zhumenghai\n");
 if(argc = = 1)
  {
  printf("\n\nDelete the given directory tree,
  delete the files and subdirectory in it\n");
  printf("nUsage:[<dirve>:][<path  name>]DDT
  [<dirve>:][<user  directory  name>]\n");
  printf("\nExamples"DDT c:\\temp,Delete
  the c:\\temp directory tree in dirve C:\n");
  exit(1);
  }
strcpy(boot,argv[1]);
strupr(boot);
save  dirver = getdisk();
getcwd(save  dir,80);
counter = setdisk(save  driver);
current  driver = save  driver;
strcpy(current  dir,save  dir);
strcpy(root,boot);
if( * (boot+1) = = ':')
  {
   current  driver = toupper( * (boot+0)) — 'A';
   for(index = 0;index < 80;++index)
    * (root+index) = * (boot+index+2);
   if(current  driver > counter)
    {
     printf("\n\nError:invalid the driver:
     [%c]\07\n",current  driver+'A");
     exit(1);
    }
   setdisk(current  driver);
   getcwd(current  dir,80);
  }
if( * (root+0)! = '\\')
  {
       strcpy(path,root);
       strcpy(root,current  dir);
       if(root[strlen(root)—1]! = '\\')
       strcat(root,"\\");
```

```
        rtrcat(root,path);
    }
/ * 进入当前驱动器中在命令行中的指定目录 * /
if(chdir(root)= =-1)
    {
     pritnf("Errot:the given path name
     <%s> is not exist!\07\07\n",root);
     exit(1);
    }
getcwd(boot,80);
strcpy(path,root);
search directory(path);
strcpy(root,boot);
if(strlen(root)>3)
    {
     counter=strlen(root)-1;
     if(root[counter]= ='\\'
      {
       root[counter]=0;
       --counter;
      }
     while(counter>0 && root[counter]!
     ='\\') --counter;
     for(index=counter+1;index<80;
     ++index) path[index-countre-1]=root[index];
      * (root+counter+1)=0;
     if(strlen(root)>3 \\ root[strlen(root)-1]
     = ='\\') root[strlen(root)=1]=0;
     if(chdir(root)= =-1)
       {
       pritnf("Errot:the given path name
       <%s> is not exist!\07\07\n",root);
       exit(1);
        }
     pritnf("\n= =%s",path);
     if(rmdir(path)= =-1)printf
     ("...undelete!\07");
     else++numbers;
 }
 if(bumbers= =0) pritnf("\n\n
 the files and sub directory does not found.\n");
 else printf("\n\n delete %d
 files and sub directorys\n",numbers);
 chdir(current dir);
 setdisk(save driver);
 chdir(save dir);
 exit(0);
}
void search directory(char * filespec)
{
 char sub d[80];
 int done;
 int length;
 int display=-1;
 int index;
 / * 删除当前目录下与指定条件匹配的文件 * /
 if( * (filespec+strlen(filespec)-1)!='\\')
 strcat(filespec,"\\");
 strcat(filespec," * . * ");
 done = findfirst(filespec,&fcb[tree],
0xff);
 while(!done)
   {
    if((fcb[tree].ff attrib & FA DIREC)!
    =FA direc)
      {
       if(tree != display)
         {
          getcwd(sub d,80);
          printf("\n%s",sub d);
          display = tree;
         }
        if((fcb[tree].ff attrib & FA LABEL)
        !=FA LABEL)
       {
       printf("\n %s",fcb[tree].ff name);
        if(remove(fcb[tree].ff name) = = -1)
        printf("...undelete!\07");
        else ++numbers;
        }
      }
    done=findnext(&fcb[tree]);
  }
/ * 搜索当前目录以下的子目录 * /
done=findfirst( * . * ),&fcb[tree],0xff);
while(!done)
    {
    if((fcb[tree].ff attrib & FA DIREC)
     = =FA DIREC && fcb[tree].ff name[0]!='.')
      {
       getcwd(root,80);
       if(root[strlen(root)-1] !='\\')
       strcat(root,"\\");
       strcat(root,fcb[tree].ff name);
       if(strlen(root)>3 && root[strlen(root)-1]
       = ='\\')root[strlen(root)-1]=0;
       chdir(rood);
       ++tree;
       search directory(root);
       getcwd(root,80);
       length = strlen(root)-1;
       if(root[length]= ='\\')
         {
         root[length]=0;
         if(strlen(root)>3 && root[strlen(root)-1]
         = ='\\')root[strlen(root)-1]=0;
       chdir(root);
       --tree;
       if(strlen(root)<(strlen(boot))) break;
       printf("\n= = =%s",sub d);
       if(rmdir(sub d)= = -1)printf
       ("...undelete!\07");
       else ++numbers;
       }
     done=findnext(&fcb[tree]);
  }
 return;
}
```