

C 语言课程设计

(C language course programming)

指 导 书

广东省交通职业技术学院北院

计算机系 余明艳编

C 语言课程设计

(C language course programming)

指 导 书

广东省交通职业技术学院北院

计算机系 余明艳编

前言

C 语言是近年来国内外得到迅速推广使用的一种现代语言。C 语言功能丰富，表达能力强，使用灵活方便、应用面广，目标程序效率高、可移植性好，既具有高级语言的优点，又具有低级语言的特点。近年来 C 语言已成为计算机专业的必修课程之一，为使同学们能将 C 语言融会贯通，能使用 C 语言去编写设计实用程序，C 语言课程设计就成为实现该目的的一个主要途径。

C 语言课程设计的主要特点是利用几周的集中时间，采用模块化、从上到下、逐步求精的设计思想，通过一个设计小组成员的相互配合完成设计任务。该指导书共有四章，第一章介绍了软件开发的基本步骤，第二章主要介绍了程序设计的基本方法，第三章对同学们提出了设计任务，第四章介绍了利用 C 语言进行编程的一些实例。

目录

第一章 软件开发的基本步骤

1.1 可行性研究

1.2 需求分析

1.3 总体设计

1.4 详细设计

1.5 编码

1.6 测试

1.7 维护

第二章 程序设计的基本方法

2.1 逐步求精的方法

2.2 模块化程序设计的方法

第三章 C语言课程设计的任务

第四章 C程序设计的实例

4.1 用C语言设计中文菜单实例

4.2 文件管理实例

第一章 软件开发的基本步骤

在计算机系统发展的早期(60年代中期以前),通用硬件相当普遍,软件却是为每个具体应用而专门编写的。这时的软件通常是规模较小的程序,编写者和使用者往往中是同一个人(或同一组人)。这种个体化的软件环境,使得软件设计通常是在人们头脑是进行的一个隐含的过程,除了程序清单之外,没有其它文档资料保存下来。

从60年代中期到70年代中期是计算机系统发展的第二代时期,这个时期的一个重要特征是出现了“软件作坊”,广泛使用产品软件。但是“软件作坊”基本是仍然沿用早期形成的个体化软件开发的方法。随着计算机应用的日益普及,软件数量急剧膨胀。在程序运行时发现的错误必须设法改正;用户有了新的需求时必须相应地修改程序;硬件或操作系统更新时,通常需要修改程序以适应新的环境。上述种种软件维护工作,以令人吃惊的比例耗费资源。更为严重的是,许多程序的个体化特性使得它们最终成为不可维护的。“软件危机”开始出现了!软件产业悄悄兴起,越来越多的软件工作者开始从事软件开发的工作。

软件开发的具体步骤通常包括:可行性研究,需求分析,总体设计,详细设计,编码,测试,维护等。

1.1 可行性研究

这个阶段的任务不是具体解决问题,而是研究问题的范围,探索这个问题是否值得去解,是否有可行的解决办法。

可行性研究阶段应该导出系统的高层逻辑模型(通常用数据流图表示),并且在此基础上更准确、更具体地确定出工程规模和目标,然后分析员更准确地估计系统的成本和效益,对建议的系统进行仔细的成本\效益分析是这个阶段的主要任务之一。

可行性研究的结果是使用部门负责人做出是否继续进行这项工程的决定的重要依据,一般说来,只有投资要取得较大效益的那些工程项目才值得继续进行下去。

1.2 需求分析

这个阶段的任务仍然不是具体地解决问题,而是准确地确定“为了解决这个问题,目标系统必须做什么”,主要确定目标系统必须具备哪些功能。

用户了解他们所面对的问题,知道必须解决什么,但是通常

不能完整地表达出他们的要求，更不知道怎样利用计算机解决他们的问题，因此在需求分析阶段，系统分析员必须和用户密切配合，充分交流信息，以得出用户确认的逻辑模型。

1.3 总体设计

这个阶段必须回答的问题是“概括地说，应该如何解决这个问题？”

首先，应该考虑几种可能的解决方案。例如目标系统的一些主要功能是用计算机自动完成还是人工完成；如果使用计算机，那么是使用批处理方式还是人机交互方式；信息存储使用传统的文件系统还是数据库……。通常至少应该考虑下述几种可能的方案：

1. 低成本的解决方案。系统只能完成最必要的工作，不能多做一点额外的工作。
2. 中等成本的解决方案。这样的系统不仅能够很好地完成预定的任务，使用起来很方便，而且可能还具有用户没有指定的某些功能和特点。
3. 高成本的“十全十美”的系统。这样的系统具有用户可希望有的所有的功能和特点。

上面的工作确定了解决问题的策略以及目标系统需要哪些程序，但是，怎样设计这些程序呢？结构设计的一条基本原理就是程序应该模块化，也就是一个大程序应该由许多规模适中的模块按合理的层次结构组织而成。总体设计的第二项主要任务就是设计软件的结构，也就是确定程序由哪些模块组成以及模块间的关系。通常用层次图或结构图描绘软件的结构。

1.4 详细设计

总体设计阶段以比较抽象概括的方式提出了解决问题的办法。详细设计阶段的任务就是把解法具体化，也就是回答下面这个关键的问题：“应该怎样具体地实现这个系统呢？”

这个阶段的任务还不是编写程序，而是设计出程序的详细规格说明。这种规格说明的作用很类似于其他工程领域中工程师经常使用的工程蓝图，它们应该包含必要的细节，程序员可以根据它们写出实际的程序代码。

1.5 编码

这个阶段的关键任务中写出正确的容易理解、容易维护的程序模块。

程序员应根据目标系统的性质和实际环境，选取一种适当的高级程序设计语言，把详细设计的结果翻译成用选定的语言书写的程序，在程序设计的过程中可使用多种灵活多样的程序设计方

法和技术。如：“自顶向下，逐步求精”，“模块化”，“求逆法”等多种基本程序设计的技术。

1. 6 测试

测试可分为单元测试和综合测试。所谓单元测试是指仔细地测试程序的每一个模块，以验证其代码的正确性。综合测试的关键任务是通过各种类型的测试（及相应的调试）使软件达到预定的要求。

最基本的测试是集成测试和验收测试。所谓集成测试是根据设计的软件结构，把经过单元测试的模块按某种选定的策略装配起来，在装配的过程中对程序进行必要的测试。所谓集成测试是根据设计的软件结构，把经过单元测试的模块按某种选定的策略装配起来，在装配的过程中对程序进行必要的测试。所谓验收测试是按照规格说明书的规定（通常在需求分析阶段确定），由用户（或在用户积极参加下）对目标系统进行验收。

应该用正式的文档资料把测试计划、详细测试方案以信实际测试的结果保存下来，做为软件配置的一个组成成分。

1. 7 软件维护

维护阶段的关键任务是，通过各种必要的维护活动使系统持久地满足用户的需要。

通常有四类维护活动：改正性维护，也就是诊断和改正使用过程中发现的软件错误；适应性维护，即修改软件以适应环境的变化；完善性维护，即根据用户的要求改进或扩充软件使它更完善。预防性维护，即修改软件为将来的维护活动预先做准备。

虽然没有把维护阶段进一步划分为更小的阶段，而事实上每一项维护活动都应该经过提出维护要求（或报告问题），分析维护要求，提出维护方案，确定维护计划，修改软件设计，修改程序，测试程序，复查验收等一系列步骤，因此实质上是经历了一次压缩和简化了的软件定义和开发的全过程。

每一项维护活动都应该准确地记录下来，做为正式的文档资料加以保存。

第二章 程序设计的基本方法

2. 1 逐步求精法

将一个完整的问题分解为若干相对独立的问题，只要将这些问题分别得到正确的解决，整个问题也就解决了。子问题又可以分解成为若干子问题重复进行下去，直到每个问题都已简单到我们满意的使我们满意的程度。对每一步分解，都要作出分解方法

的决策。不同的决策会导致不同的解法。这种程序设计的方法我们称之为逐步求精，也就是编程过程中一步步地、不断精细化的过程。

精细化过程可以从顶上始向下进行或者从底端开始向上进行。根据经验，程序自顶向下设计，再不断精化较好。

2.2 模块化程序设计的方法

为降低大型软件系统结构的复杂性，通常采用的二种有效手段是：分解和抽象。分解即将软件系统划分成若干子系统分而治之。抽象即抽取系统的次要细节。模块化方法正是基于这二者而发展起来的。

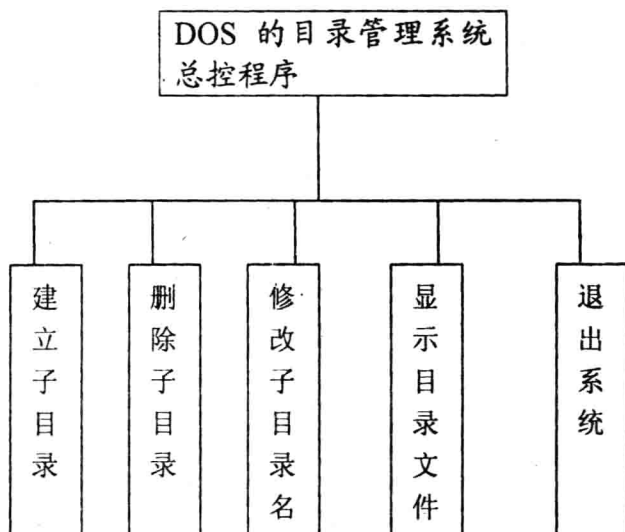
模块的概念对于大型程序设计系统及程序设计小组来说是最基本的。用模块方法编程时，程序通常都可划分为若干个模块，只要各模块之间的接口保持不变，则各模块就可以由程序设计的不同成员来编程。

模块化的设计方法是将问题分割成若干问题，并对子问题再作进一步的分割，这样形成了问题的一个层次结构。这里含有自顶向下的设计思想，再对该层次结构自下而上用模块进行逐级抽象，而得到该问题的程序，对于每个模块的具体设计，仍可采用模块化方法，即形成模块的嵌套或自顶向下逐步求精方法。这里含有自底向上的设计思想。因而模块化思想可以看作是逐步求精的一种发展。

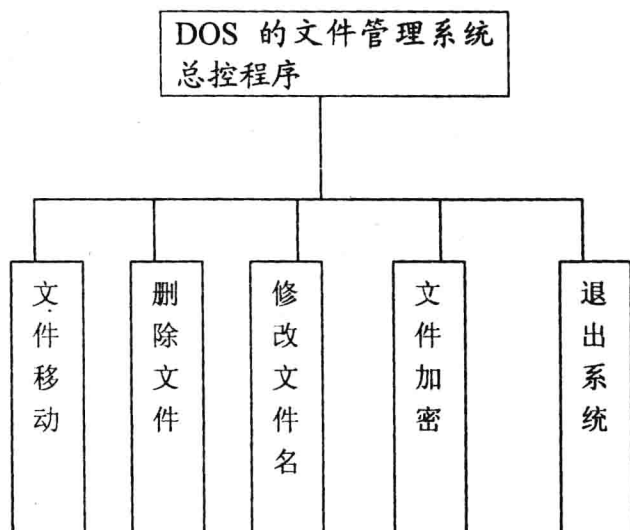
第三章 C 课程设计的任务

一、结构框图

结构框图 1



结构框图 2



二、要求:

(一) 请根据上述结构框图, 任选一个框图采用 C 语言用模块化的程序设计方法编制程序。建立一个良好的人—机对话界面。

主菜单程序设计可采用以下两种方法:

- (1) 用简易的方式显示出菜单提示部分, 给出各项功能供用户选择, 然后进行各项调用。
- (2) 根据第四章的第一节中文菜单实例仿做一个中文菜单, 并进行各模块的连接与调用。

要求人机对话介面清晰, 美观实用, 能按选择完成各项功能函数的调用。

(二) 编制各功能程序, 使它们能分别完成以下功能:

- (1) 建立子目录
- (2) 删除子目录
- (3) 修改子目录名
- (4) 显示目录文件
- (5) 文件移动
- (6) 删除文件
- (7) 修改文件名
- (8) 文件加密
- (9) 退出系统

根据以上内容, 先编制功能模块程序, 并能通过带参数的 `main()` 在 dos 状态下运行, 使其能完成相应的功能, 用总控程序实现各模块的调用。为了避免造成对 dos 系统的破坏,

各功能模块的文件名不能与 dos 中的命名相同。特作如下规定:

建立子目录——— cdir.c

删除子目录——— ddel.c

修改子目录名——— dmod.c

显示目录文件——— dddf.c

文件移动——— fremove

删除文件——— fdel.c

修改文件名——— fmod.c

文件加密——— fsecret.c

(三) 完成设计后, 需要上交以下的资料:

(1) 程序设计的文档资料, 包括程序总调用框图和各模块实现的程序名及功能使用说明。

(2) 源程序清单。

(四) 在开发该系统的过程中可借鉴以下参考资料:

1. 《C 程序设计》 谭浩强 编著 清华大学出版
2. 《C 语言编程基础》 陈庆祥 周勤学 编著 中山大学出版
3. 《微型机操作与应用基础教程》 柳青主编 高等教育出版

在实际编程的过程中可查阅 turboc2.0 的安装目录下 include 子目录中的头文件的使用。

附录部分:

一、带参数的 main() 函数

C 语言中, main 函数可定义成以下的形式:

```
main(argc,argv)
    int argc ;
    char  *argv[];
{.....
    .....
}
```

编译处理时, 与处理其它函数数据的编译方式基本相同, 但在运行时, 宿主环境 (如 dos 状态下) 可为 argc, argv 两个参数提供支持, 即用户在输入可执行文件名时 (此处可视作命令), 带上相应的参数, 执行程序时, 将程序名 (命令名) 赋值给 argv[0], 后面参数依次赋值给 argv[1], argv[2]....., 并由程序对这些参数作相应的处理。

使用参数时应注意:

1. 命令行的参数均为字符串, 如果有数值型数据, 则应用 `atoi`, `atol`, `atof` 等库函数将字符型数据转换成所需要的数据类型。

2. `main()` 函数的终止执行。

`main()` 函数的终止, 可以用 `exit` 函数来终止执行。

`exit (int status)` 函数可以使程序返回到宿主状态, 执行带的 `main()` 函数时, 其宿主状态为 dos 系统, 即程序返回到 dos 操作系统。

`int status` 的取值为 0 或 1, 其含义如下:

(1) `exit(0)` 等同于 `exit(EXIT-SUCCESS)`, 如果返回成功即在返回系统后提示 “`exit sucess(成功终止)!`” 的状态信息。

(2) `exit(1)` 等同于 `exit(EXIT-FAILURE)`

如果返回不成功, 则在返回系统后提示 “`exit failure(不成功终止)!`” 的状态信息。

二、C 语言中头文件的使用

文件的包含功能是 C 语言中预处理功能之一, 利用文件包含手段, 可减少程序编制中的重复工作, 有利于程序的维护与修改。在调用标准函数库函数时, 通过头文件 (`.h`), 可以在程序编译时, 得到一些需要经常使用到的信息, 既方便编程, 又可提高程序的通用性, 下面就课程设计中需要使用的标准头文件予以说明。

(一) `stdio.h`

`stdio.h` 是一个由若干类型, 宏与函数组成的用于输入输出的标准头文件。在该文件中, 预定义了三个类型, 其中有我们比较熟悉的 `FILE` 类型, 另外定义了部分常用的宏, 如文件结束符 `EOF`, 文件处理时常用的 `NULL` 等。还定义了一些常用的输入输出函数, 是使用得较多的标准头文件之一。本次课程设计除使用常用的输入输出函数外, 还将用到其中的两个与文件处理有关的函数:

1. `remove(路径)`, 删除文件; 该命令将使文件与文件所在的路径名脱离连接关系, 将该文件从此路径下清除 (可用于文件删除操作)
2. `rename(源文件名, 新文件名)`; 该命令将用新的文件名代替原有的文件名, 可用于文件名的修理工改。

(二)、`\sys\stat.h`

该文件是一个处理文件与目录之间关系的头文件，文件中，定义了一组与文件有关的宏，如：

```
#define S_IFDIR 0x4000 /*可用 S_IFDIRR 的值来判断当前的字符串是文件名还是目录。*/
```

```
#define S_IREAD 0x0100 /*说明文件属性为只读文件。*/
```

```
#define S_IWRITE 0x0080 /*说明该文件为可写文件。*/
```

在该文件中，还定义了一个结构体 stat，该结构体中给出了与文件有关的一组参数，如其中的 st_mode，可以用来描述文件是否有目录，如果不是目录时，可用 st_size 来计算文件的大小，stat 结构体的定义如下：

```
Struct stat
{ short st_dev; /* 结点的设备号*/
  short st_ino /*结点号*/
  short st_mode /*方式位——用来描述文件的属性，可检查是文件还是目录*/
  short st_nlink /*与文件连接号*/
  int st_uid /*自身的用户号*/
  int st_gid /*自身的组号*/
  short st_rdev /*用于特殊文件*/
  long st_size /*文件的大小（以字符计）*/
  long st_atime /*最后的存取时间*/
  long st_mtime /*最后的修改时间*/
  long st_ctime /*最初创建文件的时间*/
```

(三)、dir.h

该文件定义了与目录和路径有关的结构、宏和函数。如与文件与目录和路径有关的函数、宏及函数。如文件目录和路径有关的函数 chdir(切换目录)，mkdir(建立目录)，rmdir(删除目录)等，可以对系统的目录名进行显示、创建和删除等操作。其中的 findfirst、findnext 函数，可以用来查找所需要的路径名，在该文件中，还定义了一个和目录信路径有关的结构类型 fblk。

```
struct fblk
{ char ff_reserved[21]; /*文件的版本*/
  char ff_attrib; /*文件的属性*/
  unsigned ff_fime; /*建立文件的时间*/
  unsigned ff_fdate; /*建立文件的日期*/
  long ff_size; /*文件的大小*/
  char ff_name[31]; /*文件的名称*/
```

(四) 其它一些有关的头文件

在本次课程设计中, 根据需要及编程处理方式的不同, 还需要用到一些其它头文件, 如用于定义通用类型、变量及函数的 `stdio.h`, 用于数学处理的 `math.h`, 以及与 dos 和处理器有关的 `DOS.h`, 与屏幕操作有关的 `conio.h` 等等。

通过以上头文件的使用, 我们可以将所编源程序的结构进行优化, 完善系统的性能, 提高程序的实用性。

第四章 C 程序设计的实例

4. 1 用 C 语言设计中文菜单的实例

```
#include "bios.h"
#include "dos.h"
#define ENTER 13
#define ESC 27
int mmenu(), xlcd(), boldwin(), cahz();
static int j[4];
char *f[]={"11", "22", "33"};
char *fi[]={"111", "112"},
char *f2[]={"221", "222", "223"};
char *f3[]={"331", "332", "333", "334"};
main()
{ int mmenu(), boldwin();
void noscr(), goto_xy(), set_clr(), disp_hz();
int x=0, v=0;
system("cls");
noscr();
boldwin(1, 1, 3, 13);
disp_hz(2, 3, f[0], 12);
for(x=1; x<4; x++)
{boldwin(1, x*18+1, 3, x*18+17, 9);
disp_hz(2, x*18+3, f[x], 14);}
x=0; mmenu(x, v, 0);}
/*****/
int mmenu(int x, int v, int o)
{do {
do { if (v==75||v==77)
boldwin(1, x*18+1, 3, x*18+17, 9);
disp_hz(2, x*18+13, f[x], 14);
```

```

if (v==75) x=(x==0)?3:(x+1);
if (v==77) x=(x==3)?0:(x+1);
boldwin(1, x*18+1, 3, x*18+17, 13);
disp_hz(2, x*18+3, f[x], 12);
if(o==1) xlcd(x);
if (v==xlcd(x)) o=1;}
while(o==1);}
while ((v=keyx())!=45);}
int xlcd(int vl)
{int x;
int i, k, l; char *fx[5];
printf("x range is 0, 1, 2");
scanf("%d", &x);
switch(x)
{ case 0: l=2; for(i=0; i<l; i++) fx[i]=f1[i]; break;
  case 1: l=3; for (i=0; i<l; i++) fx[i]=f2[i]; break;
  case 2: l=4; for (i=0; i<l; i++) fx[i]=f3[i]; break; }
for (i=0; i<l; i++)
disp_hz(i*2+5, x*18+3, fx[i], 14);
i=j[x];
disp_hz(i*2+5, x*18+3, fx[i], 12);
boldwin(4, x*18+1, 2*1+5, x*18+17, 7);
do { if ((vl==72) || (vl==77))
{ cahz(1, x); mmenu(x, vl, 1);
if (vl==ENTER)
{if (i==3 && x==0)
{ system("cls"); exit(0); }
/*calcu(x, i); */ } } }
while ((vl==keyx())!=ESC);
cahz(1, x);
mmenu(x, vl, 0);
for(i=0; i<(11*2+1); i++)
for(k=0; k<=18; k++)
disp_hz(i+4, x*18+j, "", 0);}
int cahz(int ll, int xx)
{ int i, k;
for(i=0; i<=(11*2+1); i++)
for(k=0; k<=18; k++)

```

```

disp_hz(i+4, xx*18+j, "", 0);}
int keyx(void)
{ int key;
while (bioskey(1)==0);
key=bioskey(0);
key=key&0xff?(key&0xff):key>>8;
return(key);}
int boldwin(int x1, int y1, int x2, int y2, int att)
{int i;
char *bc[6]={" ", " ", " ", " ", " ", " ", " ", " ", " "};
disp_hz(x1, y1, bc[0], att);
for (i=y1+2; i<y2; i+=2)
disp_hz(x1, i, bc[1], att);
disp_hz(x1, i, bc[2], att);
for (i=x1+1; i<x2; i++)
{ disp_hz(i, y1, bc[3], att);
disp_hz(i, y2, bc[3], att);}
disp_hz(x2, y1, bc[4], att);
for (i=y1+2; i<y2; i+=2)
disp_hz(x2, i, bc[1], att);
disp_hz(x2, i, bc[5], att);}
void disp_hz(x, y, p, attrib)
int x, y;
char *p;
int attrib;
{union REGS r;
register int i, j;
for (i=y; *p; i++)
{ goto_xy(x, i);
r.h.ah=9;
r.h.bh=0;
r.x.cx=1;
r.h.al=*p++;
r.h.bl=attrib;
int86(0x10, &r, &r);}}
void goto_xy(x, y)
int x, y;
{union REGS r;

```

```
r.h.ah=2;r.h.dl=y;r.h.dh=x;r.h.bh=0;
int86(0x10,&r,&r);}
```

```
void noscr()
{ union REGS reg;
reg.h.ah=19;reg.h.al=0;int86(0x10,&reg,&reg);
}
```

程序说明:

1> 本程序用 turboc2.0 编译,生成 5 K 左右的 com 命令文件,在一般的 ccdos,super ccdos、ucdos 等汉字系统下均能正确运行,美观程序略次于图形编程。

2> 在本菜单程序中当移到某选项通过回车后,连到 calcu() 函数,这是确认项转去执行的模块,有菜单号 X 和子菜单 i 两个参数传递,据此就可去计算所需选项的内容。此处也可改为开关语句,按 x 和 i 分别转入相应的处理模块或调用模块,当然用户在改编自己的实用菜单时,还需要将前面的指针数组初值改为适当的汉字。

4.2 文件管理实例

下面的实例要用于修改文件名或目录名

```
/* modify the filename or the name of subdirectory*/
#include <stdio.h>
#include <dos.h>
#include <dir.h>
main(argc,argv)
int argc;
char *argv[];
{ char source [15],object[15];
int i;
printf("Modify the filename or the name of subdirectory\n");
switch(argc)
{case 1:/*直接在 turbo c 下运行*/
printf("please input the source filename->");
scanf("%s",source);
printf("please input the object->");
scanf("%s",object);
break;}
case 2:/*由命令行给原文件名,按提示输入新文件*/
printf("the source file ->%s",argv[1]);
```

```

strcpy(source,argv[1]);
printf("please input the object filename—>");
scanf("%s",object);
break;
}
case 3: { /*由命令行给出原文件名和新文件名*/
{printf("the source filename—>%s",argv[1]);
printf("the object filename —>%s",argv[2]);
strcpy(source,argv[1]);
strcpy(object,argv[2]);
break;
}
default: { printf("Too many prarmeters:\n");
exit(0);}
}
if rename(source,object)!=0) printf("rename error!\n");
else printf("rename successful! \n");
}

```

程序中说明如下：该程序名为 chna.c,编译后行成名为 chan.exe 的可执行文件。

1> 该程序使用带参的 main()函数，在 dos 状态下运行时，命令行中的第一个参数为可执行文件 chan.exe 的文件名 chan;

2> 该文件执行时可有以下方式：

A、 在 C 语言环境下执行该程序，运行时出现提示，根据提示输入原文件名和新文件名，程序运行后用新文件名替代原文件名。

B、 在 DOS 状态下，在命令行中输入：命令名 参数 1，系统将参数 1 作为原文件名，并提示：“please input the object filename—>”，根据提示输入新的文件名，程序运行后用输入的新文件名代替命令行中参数 1 所给出的原文件名。

C、 在 D O S 状态下输入：命令名 参数 1 参数 2，则程序运行时将参数 1 所表示的原文件名用参数 2 所表示的新文件名代替。

3> 该程序中预包含的头文件名为 stdio.h、dos.h、dir.h。程序中使用了与文件操作有关的函数 rename。

4> 在处理子目录及带路径名的文件时，应注意判断用“\”分隔的路径名，并区分是路径名还是文件名。