

微机应用基本技能丛书



C语言实用程序汇编

孙尚淳 编



化学工业出版社

TP312
SSC/1

微机应用基本技能丛书

C 语言实用程序汇编

孙尚淳 编

化学工业出版社

北京

(京)新登字 039 号

图书在版编目(CIP)数据

C 语言实用程序汇编/孙尚淳编. —北京:化学工业出版社, 1997. 6

(微机应用基本技能丛书)

ISBN 7-5025-1611-5

I. C... II. 孙... III. C 语言-程序设计 IV. TP312

中国版本图书馆 CIP 数据核字(96)第 0000*号

微机应用基本技能丛书

C 语言实用程序汇编

孙尚淳 编

责任编辑:高 钰

封面设计:于 兵

*

化学工业出版社出版发行

(北京市朝阳区惠新里 3 号 邮政编码 100029)

新华书店北京发行所经销

北京管庄永胜印刷厂印刷

三河市延风装订厂装订

*

开本 850 × 1168 毫米 1/32 印张 7 1/2 字数 194 千字

1997 年 10 月第 1 版 1997 年 10 月北京第 1 次印刷

印数: 1 — 5500 册

ISBN 7-5025-1611-5 / TP · 62

定价: 15.00 元

版权所有 违者必究

该书如有缺页、倒页、脱页者,本社发行部负责调换

目 录

绪论 如何编写一个程序.....	1
第一节 编程前的准备.....	1
1. 编写程序的过程.....	1
2. 熟悉一种编译器.....	2
第二节 TURBO C 简介.....	2
第三节 编写第一个程序.....	3
 第一章 基本应用技巧.....	5
第一节 使用命令行参数.....	5
1. 什么是命令行参数.....	5
2. 如何在程序中使用命令行参数.....	5
3. 加密文本文件(应用程序举例).....	6
第二节 使用随机数.....	11
第三节 函数的递归调用.....	14
第四节 使用链表.....	16
 第二章 使用系统功能.....	21
第一节 利用 BIOS、DOS 中断例行程序.....	21
1. 利用 BIOS 中断例行程序.....	21
2. 利用 DOS 例行程序.....	24
第二节 键盘控制.....	26
第三节 使用声音.....	29
第四节 读取系统的日期和时间.....	30
1. 读取系统日期.....	31
2. 读取系统时间.....	31
 第三章 文本模式下的屏幕操作.....	33
第一节 计算机的显示系统.....	33

第二节 文本模式下的显示缓冲区	34
第三节 屏幕操作	35
1. 文本窗口的定义	35
2. 显示属性的设置	36
3. 清除窗口	36
4. 窗口内的文本输入输出	36
5. 控制光标	38
6. 保存屏幕内容	40
第四章 图形功能	45
第一节 如何使用 TURBO C 的图形功能	45
第二节 图形操作中所用的颜色	48
第三节 曲线图	49
第四节 直方图	55
第五节 圆饼图	59
第六节 其他图形函数	63
第七节 建立独立图形运行程序	63
第五章 动画功能	65
第一节 异或方式	65
第二节 换页方式	68
第三节 动画应用实例（自动报时钟）	71
第六章 使用鼠标	79
第一节 如何进行鼠标操作	79
第二节 实屏与虚屏	79
第三节 用于鼠标操作的基本函数	80
第四节 建立鼠标程序库	87
第五节 文本模式下使用鼠标	88
第六节 图形模式下使用光标	91
第七节 鼠标应用实例（模拟扫雷游戏）	93
第七章 建立文本模式下的窗口及菜单系统	105
第一节 建立窗口库	105

1. VIEW 结构	105
2. 初始化窗口 View_Init()	106
3. 打开窗口 View_Open()	108
4. 关闭窗口 View_Close()	110
5. 清除窗口内容 View_Clear()	111
6. 输出字符串 View_PutString()	111
7. 建立窗口库 VIEW.LIB	112
8. 窗口程序举例	114
第二节 建立菜单库	116
1. MENU 结构	116
2. 初始化菜单 Menu_Init()	116
3. 显示当前选项 Menu_ShowCurItem()	118
4. 打开菜单 Menu_Open()	118
5. 关闭菜单 Menu_Close()	119
6. 驱动菜单 Menu_Drive()	119
7. 建立菜单库 MENU.LIB	121
8. 菜单应用举例	122
第三节 建立下拉式菜单	124
第八章 简单的数据库管理	131
第一节 建立描述人员信息的数据结构	131
1. PERSON 结构	131
2. CARD 结构	132
3. BOOK 结构	133
4. 初始化 BOOK Book_Init()	133
5. 添加成员 Book_AppendCard()	133
6. 插入成员 Book_InsertCard()	135
7. 删除成员 Book_DeleteCard()	135
8. 更换当前项	137
9. 建立头文件 BOOK.H	137
第二节 建立界面功能函数	139
1. 隐藏和恢复光标 HideCursor()和 ShowCursor()	139

2. 显示人员信息 ShowCard()	140
3. 取得用户输入 GetMessage()和 InputString()	141
4. 输入人员信息 InputCard().....	143
5. 添加人员信息项目 AppendCard().....	144
6. 插入人员信息项 InsertCard().....	145
7. DeleteCard().....	145
8. 保存和装入信息 SaveBook()和 LoadBook()..	146
第三节 建立主程序	148
第九章 西文状态下显示汉字	153
第一节 显示 16 × 16 点阵汉字.....	153
第二节 显示 24 × 24 点阵汉字.....	158
第三节 更复杂的输出方式	162
第十章 西文状态下显示矢量汉字	169
第一节 读取矢量字库.....	169
1. 字形的矢量描述	169
2. 矢量字库的结构	170
第二节 填充字形.....	186
第三节 显示带阴影的汉字	202
第十一章 使用 VGA256 色模式显示 BMP 图像.....	207
第一节 如何实现 VGA256 色的图形操作.....	207
1. 功能 0 设置显示模式.....	207
2. 功能 0x0f 获取当前显示模式.....	207
3. 功能 0x0c 写像素点	208
4. 功能 0x10 设置调色板寄存器.....	208
5. 建立图形操作函数	208
第二节 如何读取 BMP 图形文件.....	210
1. BITMAPFILEHEADER 结构.....	210
2. BITMAPINFO 结构.....	211
3. 图形阵列	213
4. 读取图像	213

第三节 建立读取 BMP 图形的程序 VIEWBMP213

附录 TURBO C 2.0 简介	221
1. TURBO C 2.0 的安装	221
2. 菜单功能	222
3. 使用项目(Project)文件	225
4. 建立库文件	225

绪论 如何编写一个程序

第一节 编程前的准备

1. 编写程序的过程

编写一个程序，需要经过以下几个过程：

① 编辑(Edit)。

在一个文本编辑器中，按照C语言的语法输入程序代码，并对源代码进行编辑修改，直至无误后将其保存在磁盘文件中，文件扩展名一般为*.C，如HELLO.C。这种文件是以ASCII码形式保存的，不能被计算机执行，因此称为源程序。

② 编译(Compile)。

用专用的编译程序对已编辑好的源程序进行编译，若编译成功将自动生成相应的*.OBJ文件，如HELLO.C编译后生成HELLO.OBJ，若不成功往往是因为源程序语法有错误，编译程序“看不懂”源程序，无法进行编译，但一般会给出是哪种错误的提示信息，这时就要回到编辑器中，根据提示信息找出源程序的错误所在，并对其进行修改，然后重新编译，直至通过为止。值得指出的是，即使是有经验的程序员也不能保证一次就编译成功，这一过程往往要反复多次，提高效率有赖于实践经验和良好的编程习惯。

编译后生成的*.OBJ文件称为目标文件，是二进制形式的，但还不能被计算机执行，因为一个程序可能由多个模块构成，各个模块分别进行编译，生成各自的*.OBJ文件，它们还没有“组装”好。

③ 连接(Link)。

用专用的连接程序将已编译好的各目标模块连接起来，若连接成功将自动生成*.EXE文件，若不成功会给出提示信息，

这时还要回到编辑器中修改源文件，再重复编译、连接过程。可以看出*.EXE文件就是DOS下的可执行文件，这时就可以在DOS提示符下执行该程序了。

④ 调试(Debug)。

连接后生成的*.EXE文件虽然可以执行，但有可能没有实现我们预想的功能，原因是我们在设计程序时有些没考虑周全的地方，这时就要根据错误的现象，分析出错误的原因，修改源文件，重新编译、连接，必要时可以借助专用的调试工具，使用跟踪等调试手段。对于复杂的程序，这一过程往往是漫长而艰苦的。

2. 熟悉一种编译器

通过以上的叙述，读者可能会感到编程是一项非常繁琐的工作，的确，需要有足够的耐心。所幸的是，现在出现了许多商品化的编译器，它们能提供集成化的编程环境。所谓“集成化”，是指编辑源文件、编译、连接、运行调试都在同一个软件环境下进行，为使用者提供了极大的方便。

选择编译器时应注意一个问题：由于生产编译器的厂商往往按各自的标准对C语言进行了扩充，这就带来了不同版本C语言的差异，这些差异大多体现在较底层的功能上，如图形功能等，但这不会成为我们学习C语言的真正障碍，正如学好一门外语不仅仅在于多记几个单词，要学到C语言的精髓，决不仅仅意味着掌握几条语句，几个函数，而是要领略到其中的编程思想，这才是最重要的。

目前流行的编译器很多，有些功能十分强大，但使用起来也较复杂，况且有许多功能是初学者很难用到的，因此本书以Borland公司的TURBO C 2.0为基础加以介绍，书中涉及到版本问题时将会指明，读者可根据情况选择自己喜欢的编译器。

第二节 TURBO C 简介

TURBO C是美国Borland公司生产的C语言程序开发环

境，它集程序编辑、编译、连接、调试于一体，具有快速高效的特点，使用起来灵活方便，深受广大软件开发人员的青睐。TURBO C 2.0是较常用的一种版本，其功能介绍及安装方法可参见附录一。

如果您已安装了TURBO C 2.0，只要在TC子目录下输入TC即可进入TURBO C开发环境，此时屏幕将如图0-1所示。

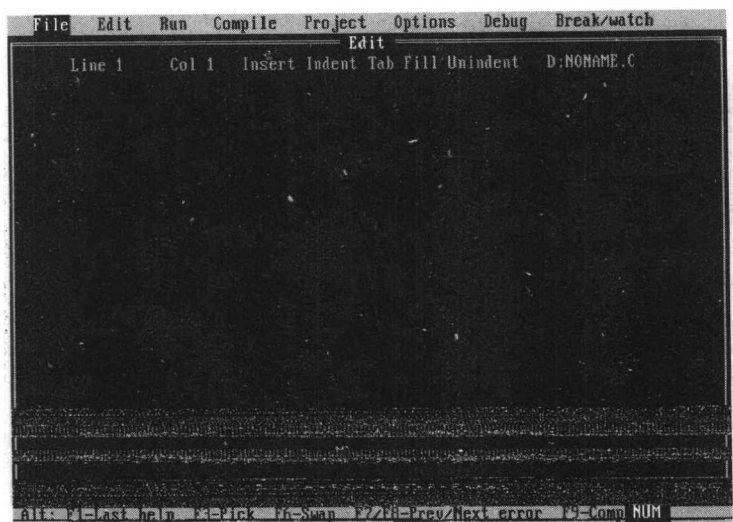


图 0-1 TURBO C 开发环境

屏幕最上面一行为功能菜单，通过它可以完成文件管理、编译连接、跟踪调试等功能，其各选项功能可参见本书附录一；屏幕中间的主体部分为编辑区，它实际上是一个编辑器，可以用它来编辑源文件；屏幕下方为信息窗口，在对程序进行编译连接时用来显示错误或警告信息。

第三节 编写第一个程序

运行TC，进入TURBO C环境，选择File菜单的New项，此时编译器将打开一个空的编辑窗口，文件名为NONAME。

绪论 如何编写一个程序

在编辑窗口中输入以下代码，输入时注意区分大小写，并确保输入正确。

```
main()
{
    printf("hello !");
}
```

输入后的屏幕如图0-2所示。

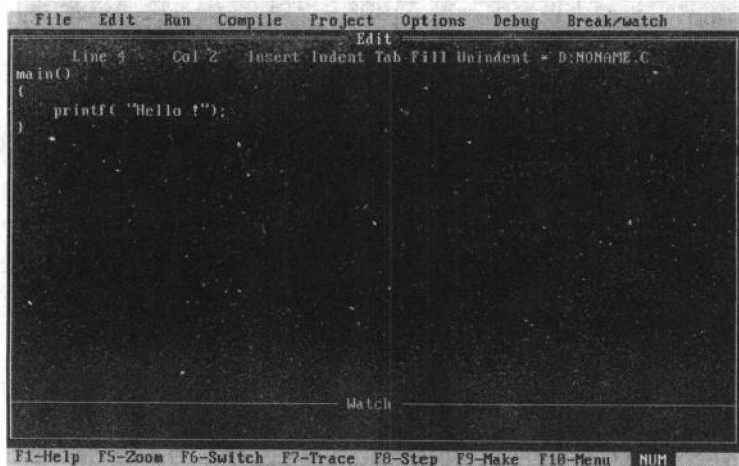


图 0-2 输入代码后的屏幕

选择File菜单Save项，此时将要求用户输入文件名，例如输入HELLO.C，则以该文件名存盘。

选择菜单Compile的Compile项，编译。若未通过，将在Message区显示出错信息，这时应回到编辑区修改源代码，重新编译。

编译成功后，选择菜单Compile的Link项，连接。若未成功，将显示出错信息。

连接成功后，退出TC，在DOS下用DIR命令看是否生成了HELLO.EXE文件，键入HELLO，屏幕上将显示：

```
C:\>hello!
```

第一章 基本应用技巧

第一节 使用命令行参数

1. 什么是命令行参数

通常情况下，假设我们编写了一个名为 `FILE.C` 源程序文件，编译后生成 `FILE.EXE` 文件，如果想在 `DOS` 下执行该程序，只要在 `DOS` 提示符下输入文件名即可，如：

```
C:\>FILE [enter]
```

而当我们使用某些 `DOS` 外部命令时，除了输入文件名之外，还可以（有时是必须）输入一些参数，如：

```
C:\>FORMAT A:
```

这些参数被称为命令行参数，它将控制着程序如何运行。

2. 如何在程序中使用命令行参数

我们能不能使自己的程序在运行时也能接受命令行参数，并根据这些参数确定程序如何运行呢？

通常我们定义的 `main()` 函数是不带参数的，但实际上它可以有两个参数，即 `main()` 函数可以写成下面的形式：

```
main(int argc, char* argv[ ])
```

显然 `argc` 是一个整型变量，`argv` 是一个指针数组。我们都知道，`main()` 函数是由系统调用的，那么，这两个参数分别表示什么？又是从哪里来的呢？

事实上，当我们在 `DOS` 提示符下输入如下字符串来运行 `FILE` 程序时：

```
C:\>FILE PARAMETER1 PARAMETER2[enter]
```

程序运行时变量 `argc` 就等于 3, 表示 3 个字符串, 而 `argv[]` 中就保存着这些字符串的指针:

```
argv[0]="FILE"  
argv[1]="PARAMATER1"  
argv[2]="PARAMATER2"
```

这样, 我们就可以在程序中检查这些参数, 使我们的程序更加灵活。

注意

`argv[0]` 保存的是文件名 (如果文件名前有路径, 路径信息也在 `argv[0]` 中), `argv[1]` 保存的才是第一个参数。

3. 加密文本文件 (应用程序举例)

下面的两个程序 `CIPHER.C` 用于加密和解密一个文本文件, 其中就用到了命令行参数。它们的用法是这样的:

```
CIPHER FILE.TXT
```

将文本文件 `FILE.TXT` 加密, 生成新的文本文件 `FILE.CPH`, 而

```
CIPHER FILE.CPH /X
```

则是将加密后的文本文件解密, 得到原始的文本文件 `FILE.ORG`, 即加密或解密哪一个文件, 依赖于运行程序时所输入的参数。

例 1-1

```
/****** Cipher.C *****/  
#include <stdio.h>  
  
typedef struct  
{  
    char OrgChar;
```

```

    char CphChar;
} ITEM;

TEM Table[] =
{
    {'a','m'}, {'b','d'}, {'c','l'},
    {'d','y'}, {'e','u'}, {'f','o'},
    {'g','h'}, {'h','c'}, {'i','r'},
    {'j','a'}, {'k','p'}, {'l','w'},
    {'m','t'}, {'n','j'}, {'o','g'},
    {'p','l'}, {'q','e'}, {'r','v'},
    {'s','b'}, {'t','x'}, {'u','q'},
    {'v','f'}, {'w','k'}, {'x','z'},
    {'y','s'}, {'z','n'},
};

/* 取得加密字符 */
char GetCphChar(char orgChar)
{
    int i, num;
    num = sizeof(Table)/sizeof(ITEM);
    for(i=0; i<num; i++)
        if(Table[i].OrgChar==orgChar)
            return Table[i].CphChar;
    return orgChar;
}

/* 取得原始字符 */
char GetOrgChar(char cphChar)
{
    int i, num;
    num = sizeof(Table)/sizeof(ITEM);
    for(i=0; i<num; i++)
        if(Table[i].CphChar==cphChar)
            return Table[i].OrgChar;
    return cphChar;
}

```

```
}  
  
main(int argc, char *argv[])  
{  
    FILE *srcFile, *rstFile;  
    char fileName[80];  
    char ch, *p;  
    int cipher;  
  
    /* 检查参数 */  
    if(argc==2)  
        cipher = 1;  
    else if(argc == 3 &&  
        ((!strcmp("/X", argv[2]))||(!strcmp("/x",  
argv[2]))) )  
        cipher = 0;  
    else  
    {    puts("parameter error !");  
        exit(0);  
    }  
  
    /* 打开原文件 */  
    strcpy(fileName, argv[1]);  
    srcFile = fopen(fileName, "r");  
    if(srcFile == NULL)  
    {    printf("Can't open file %s !", fileName);  
        exit(0);  
    }  
  
    /* 建立生成文件 */  
    p = fileName + strlen(fileName);  
    while(p>fileName && *p! = "\\")  
    {    if(*p == '.')  
        {    *p = 0;  
            break;  
        }  
    }
```

```

    p--;
}
if(cipher)
    strcat(fileName, ".CPH");
else
    strcat(fileName, ".ORG");
rstFile=fopen(fileName,"w");
if(rstFile == NULL)
{
    printf("Can't create%s!",fileName);
    fclose(srcFile);
    exit(0);
}

while(1)
{
    ch = fgetc(srcFile);
    if(ch == EOF)
        break;
    if(cipher)
        ch = GetCphChar(ch);    /* 取得加密字符 */
    else
        ch = GetOrgChar(ch);    /* 取得原始字符 */
    fputc(ch,rstFile);
}

fclose(srcFile);
fclose(rstFile);
}

```

程序先定义了一个 ITEM 类型，该类型是一个结构，包含着两个字符变量，OrgChar 表示原始字符，CphChar 表示加密后的字符。接着定义了一个 ITEM 数组 Table，保存着我们想要加密的字符的对照关系，如 a 加密后为 m，因此 Table 实际上是一个密码表。



注意

加密字符和原始字符必须一一对应，不能有重复，否则加