

目 录

第一部分 运行 Borland C++

第一章 IDE 入门

1.1 安装 Borland C++	2
1.1.1 对硬件和软件的要求	2
1.1.2 安装步骤	2
1.1.3 启动 Borland C++	4
1.1.4 获取帮助信息	5
1.2 配置 IDE	5
1.2.1 改变 SpeedBars	6
1.2.2 设置 IDE preference	6
1.2.3 保存 IDE 的设置	7
1.3 使用编辑器	7
1.3.1 配置 IDE 的编辑器	7
1.3.2 语法制导彩色显示(Syntax Highlighting)	8
1.4 使用消息窗口(Message)工作	8
1.5 浏览代码	9
1.5.1 浏览对象的类	10
1.5.2 过滤器	10
1.5.3 查看所列符号的声明	10
1.5.4 浏览全局符号	11
1.5.5 在浏览器中使用一般通配符	12
1.5.6 浏览源代码中的符号	12
1.6 使用命令行工具	12
1.6.1 DPMI 和命令行工具	13
1.6.2 内存和 MAKESWAP.EXE	13
1.6.3 运行时间管理器和工具	13
1.6.4 控制 RTM 占用的内存	14
1.7 在 IDE 中运行其他程序	14

第二章 运行工程管理器

2.1 什么是工程管理器	16
2.2 建立一个工程	17
2.2.1 建立一个多任务工程	18
2.2.2 转换旧的工程	19
2.2.3 将工程转化为制作文件(makefiles)	19
2.2.4 改变工程视图	19
2.3 建立一个工程	20
2.3.1 建立工程的一部分	20
2.4 编辑工程树	21
2.4.1 使用 TargetExpert 编辑任务属性	21
2.4.2 编辑节点属性	21
2.4.3 增加和删除一个节点	22
2.4.4 增加和删除任务	22
2.4.5 移动节点和任务	23
2.4.6 拷贝节点	23
2.5 使用 Source Pool	23
2.6 设置工程选项	24
2.6.1 Local Override	24
2.6.2 使用 Style Sheet	25
2.6.3 将一个 Style Sheet 与一个节点相连	25
2.6.4 生成一个 Style Sheet	26
2.6.5 编辑 Style Sheet	26
2.6.6 共享 Style Sheet	26
2.6.7 查看工程中的选项	27
2.7 转换器	27
2.7.1 安装一个转换器	28
2.7.2 使用 SpeedMenu 中的 Special 命令	29
2.7.3 安装观察器和工具	29

第三章 编译

3.1 在 IDE 中编译	30
3.1.1 使用 IDE 编译器选项	30
3.2 使用命令行编译器	31
3.2.1 配置文件	31
3.2.2 应答文件	32
3.2.3 选项的优先级规则	32
3.3 编译选项参考	32

3.4	目录选项	44
3.4.1	查找文件算法	45
3.5	编译器宏定义选项(Compiler Defines)	45
3.6	编译器代码生成选项(Compiler Code-generation)	46
3.7	编译器浮点处理选项(Compiler Floating Point)	47
3.8	编译器输出选项(Compiler Compiler Output)	47
3.8.1	编译源文件代码选项(Compiler Source)	48
3.9	编译调试选项(Compiler Debugging)	48
3.10	预编译头文件选项(Compiler Precompiled headers)	50
3.11	16位编译处理器选项(16-bit Compiler Processor)	50
3.11.1	16位编译器调用约定选项(16-bit Compiler Calling Convention)	51
3.12	16位编译存储模式选项(16-bit Compiler Memory Model)	51
3.13	16位编译数据段命名选项(16-bit Compiler Segment Names Data)	53
3.14	16位远程数据段命名选项(Compiler Segment Names Far Data)	54
3.15	16位编译代码段命名选项(16-bit Compiler Segment Names Code)	54
3.15.1	16位编译入口 出口代码选项 (16-bit Compiler Entry/Exit Code)	55
3.16	32位编译处理器选项(32-bit Compiler Processor)	56
3.17	32位编译调用约定选项(32-bit Compiler Calling Convention)	57
3.18	C++选项 成员指针(C++ Options Member Pointer)	57
3.19	C++兼容性选项(C++ Options Compatibility)	58
3.20	C++虚表选项(C++ Options Virtual Tables)	59
3.21	C++模板生成选项(C++ Options Templates)	59
3.22	C++例外控制选项(C++ Options Exception handling/RTTI)	60
3.22.1	优化选项	60
3.23	优化声明选项(Optimizations Specific)	60
3.24	优化代码长度选项(Optimization Size)	62
3.25	优化速度选项(Optimization Speed)	63
3.26	消息选项(Messages)	65
3.27	移植性警告选项(Messages Portability)	65
3.28	ANSI 违例警告(Message ANSI Violations)	65
3.29	C++错误(Messages Obsolete C++)	66
3.30	潜在的C++错误(Messages Potential C++ Errors)	66
3.31	无效C++代码(Messages Inefficient C++ Coding)	67
3.32	潜在错误(Messages Potential errors)	67
3.32.1	无效代码(Messages Inefficient coding)	68
3.33	一般错误(Message General)	68
3.34	Make选项	68
3.35	命令行选项	68

第四章 运行 AppExpert 构造应用程序

4.1 AppExpert 基础	72
4.2 使用 AppExpert 生成一个应用程序	72
4.2.1 缺省的 AppExpert 应用程序	73
4.3 应用程序选项	73
4.3.1 应用程序的基本选项 (Application Basic Options)	74
4.3.2 应用程序的高级选项 (Application Advanced Options)	74
4.3.3 应用程序的代码生成控制选项 (Application Code Gen Control)	75
4.3.4 应用程序管理选项 (Application Admin Options)	75
4.4 主窗口选项 (Main Window Options)	75
4.4.1 主窗口基本选项 (Main window Basic Options)	76
4.4.2 主窗口的 SDI 用户选项 (Main Windows SDI Client)	76
4.4.3 主窗口 MDI 用户选项 (Main Window MDI Client)	77
4.5 MDI 子窗口显示选项 (MDI Child/View Options)	77
4.5.1 MDI 子窗口 显示的基础选项 (MDI Child/View Basic Options)	77

第五章 运行 ClassExpert

5.1 启动 ClassExpert	79
5.1.1 Class Expert 基础	79
5.1.2 添加一个类	80
5.1.3 创建文档类型	81
5.1.4 添加和删除事件处理程序	82
5.1.5 添加和删除实例变量	82
5.1.6 跳转到类的源代码	83
5.2 使用资源管理程序与 ClassExpert	83
5.2.1 IDE 中运行 Resource Workshop	84
5.3 使用 Rescan	84
5.3.1 删除一个类	84
5.3.2 移动一个类	85
5.3.3 更名一个 AppEXpert 元素	85
5.3.4 引入一个类	85
5.3.5 重建 .APX 数据库文件	85

第六章 运行集成调试器

6.1 错误的类型	86
6.1.1 编译时间错误	86
6.1.2 运行时间错误	87
6.1.3 逻辑错误	87

6.2	生成调试信息	87
6.3	指明程序参数	87
6.4	控制程序执行	87
6.4.1	监视程序的输出(Watching program output)	88
6.4.2	单步执行程序(Step over code)	88
6.4.3	跟踪执行代码(Tracing into code)	89
6.4.4	单步跳过一段程序代码	89
6.4.5	停止程序运行	90
6.4.6	重新开始	90
6.5	检查变量的值	91
6.5.1	什么是表达式	91
6.5.2	监视表达式	91
6.5.3	计算和修改表达式	93
6.5.4	检查数据元素	94
6.5.5	检查寄存器的值	95
6.6	使用断点	96
6.6.1	设置断点	96
6.6.2	使用断点工作	96
6.6.3	用户定做断点和执行点	98
6.7	解决一般保护错误	99
6.8	使用 Event Log 窗口	99
6.9	调试动态连接库	99

第七章 运行 WinSight

7.1	开始启动	101
7.1.1	启动和结束屏幕更新	101
7.1.2	关闭消息跟踪	101
7.2	选择视窗	102
7.3	Class List	102
7.3.1	使用 Class List 视窗	102
7.3.2	监测类	102
7.4	Window Tree	103
7.4.1	寻找窗口	103
7.4.2	监测窗口	104
7.5	选择跟踪消息	104
7.5.1	使用 Message Trace 视窗	104
7.5.2	其它的跟踪选项	104

第八章 运行 WinSpector

8.1 使用 WinSpector	109
8.1.1 配置 WINSPECTR.LOG	109
8.1.2 WINSPECTR.LOG 参考	110
8.2 处理 WinSpector 数据	114
8.2.1 DFA 输出	114
8.2.2 使用带有 WINSPECTR.LOG 的 DFA	115
8.2.3 使用带 WINSPECTR.BIN 的 DFA	115
8.3 其它 WinSpector 工具	115
8.3.1 使用 EXEMAP.EXE	116
8.3.2 使用 TMAPSYM.EXE	116
8.3.3 使用 BUILDSYM.EXE	116

第九章 运行连接程序:TLINK

9.1 TLINK 基础	118
9.1.1 TLINK.CFG	119
9.1.2 响应文件	119
9.1.3 用 BCC.EXE 使用 TLINK	120
9.1.4 连接库	120
9.2 TLINK 选项	121
9.3 模块定义文件参考	127
9.3.1 CODE 语句	127
9.3.2 DATA 语句	128
9.3.3 DESCRIPTION 语句	128
9.3.4 EXETYPE 语句	128
9.3.5 EXPORT 语句	129
9.3.6 IMPORTS 语句	129
9.3.7 LIBRARY 语句	130
9.3.8 NAME 语句	130
9.3.9 SEGMENTS 语句	130
9.3.10 STACKSIZE 语句	131
9.3.11 STUB 语句	131
9.3.12 缺省的模式定义文件	131

第十章 运行资源工具

10.1 BRCC.EXE:资源编译器	133
10.2 RLINK:资源连接器	134
10.3 BRC.EXE 资源外壳	135

第十一章 运行库管理程序

11.1 使用 IMPLIB; 导入库	137
11.2 使用 IMPDEF; 模块定义文件管理器	138
11.2.1 在 DLL 中分类	138
11.2.2 在 DLL 中的函数	139
11.3 使用 TLIB; Turbo 库	139
11.3.1 为什么要使用目标模块库	140
11.3.2 TLIB 命令行	140
11.4 例子	142

第十二章 运行 MAKE

12.1 MAKE 基础	144
12.1.1 BUILTHINS.MAK	145
12.1.2 使用 TOUCH.EXE	145
12.1.3 MAKE 选项	145
12.2 使用 makefile	147
12.2.1 符号目标	148
12.3 显式和隐式规则	148
12.3.1 显式规则语法	148
12.3.2 隐式规则语法	149
12.3.3 命令语法	150
12.4 使用 MAKE 宏	152
12.4.1 定义宏	152
12.4.2 使用宏	152
12.4.3 在宏中替换字符串	153
12.4.4 缺省的 MAKE 宏	153
12.4.5 修改缺省宏	154
12.5 使用 MAKE 指令	154
12.5.1 自动依赖	155
12.5.2 ! error	155
12.5.3 ! if 和其它条件指示字	156
12.5.4 ! include	157
12.5.5 ! message	157
12.5.6 .path.ext	157
12.5.7 .precious	158
12.5.8 .suffixes	158
12.5.9 ! undef	158
12.5.10 在指示字中使用宏	158

第二部分 Borland C++ 语言基础

第十三章 C 语言概述

13.1 预备知识.....	162
13.1.1 源文件、目标文件和装载模块	162
13.1.2 程序的逻辑和执行流程.....	163
13.2 基本数据类型.....	168
13.2.1 C 语言的基本数据类型.....	169
13.2.2 何处定义数据对象.....	171
13.3 编写 C 语言表达式和语句	173
13.3.1 表达式和语句.....	173
13.4 控制类型转换.....	177
13.4.1 理解隐式类型转换.....	177
13.5 显式类型转换的使用.....	179
13.6 使用 C 的宏	180
13.6.1 定义类似对象的宏.....	180
13.6.2 定义类似函数的宏.....	183

第十四章 操作符和表达式

14.1 什么是操作符.....	187
14.2 单目操作符.....	187
14.3 双目操作符.....	188
14.4 三目操作符.....	189
14.5 标点符号.....	189
14.6 操作符语义.....	191
14.6.1 后缀和前缀操作符.....	191
14.6.2 单目操作符.....	193
14.6.3 sizeof 操作符	194
14.6.4 乘法类操作符.....	194
14.6.5 加法类操作符.....	195
14.6.6 按位移位操作符.....	195
14.6.7 关系操作符.....	196
14.6.8 相等类操作符.....	196
14.6.9 位运算操作符.....	197
14.6.10 逻辑运算符	198
14.6.11 条件操作符?:	199

14.6.12	赋值操作符	199
14.6.13	逗号操作符	200
14.7	高级运算符的使用实例	200
14.7.1	位运算符	200
14.7.2	移位运算符	202
14.7.3	?:运算符	206
14.7.4	C语言的简写	207
14.7.5	逗号运算符	208
14.7.6	运算符优先级表	208
14.8	表达式	209

第十五章 说 明

15.1	有关概念	212
15.1.1	对 象	212
15.1.2	左 值	212
15.1.3	右 值	213
15.1.4	类型与存储类	213
15.1.5	作用域	213
15.1.6	可见性	214
15.1.7	生存期	214
15.1.8	编译单元	215
15.1.9	连 接	215
15.2	说明的语法	216
15.2.1	暂时定义	216
15.2.2	可能的说明	216
15.3	类型说明	219
15.3.1	外部说明与定义	219
15.3.2	类型指明符	220
15.3.3	类型分类	220
15.3.4	基本类型	220
15.3.5	标准转换	224
15.3.6	特殊的 char、int 与 enum 间的转换	224
15.3.7	初始化	224
15.4	简单说明	227
15.5	存储类指明符	227
15.5.1	存储类指明符 auto 的使用	227
15.5.2	存储类指明符 extern 的使用	227
15.5.3	存储类指明符 register 的使用	227
15.5.4	存储类指明符 static 的使用	228

15.5.5	存储类指明符 typedef 的使用	228
15.6	修饰符	228
15.6.1	const 修饰符	229
15.6.2	中断函数修饰符	229
15.6.3	volatile 修饰符	230
15.6.4	cdecl 与 pascal 修饰符	230
15.6.5	指针修饰符	231
15.6.6	函数类型修饰符	231
15.7	复杂说明与说明符	232

第十六章 程序控制语句

16.1	程序控制语句的语法	234
16.1.1	带标号语句	235
16.1.2	复合语句	235
16.1.3	表达式语句	235
16.1.4	选择语句	236
16.1.5	循环语句	237
16.1.6	跳转语句	238
16.2	if 语句	239
16.2.1	else 语句的用法	240
16.2.2	if-else-if 阶梯的用法	241
16.2.3	条件表达式	242
16.2.4	if 语句的嵌套结构	243
16.3	switch 语句	243
16.3.1	default 语句的用法	245
16.3.2	break 语句的用法	246
16.3.3	switch 语句的嵌套结构	246
16.4	循 环	249
16.5	for 循环	249
16.5.1	for 循环的灵活用法	250
16.5.2	无穷的 for 循环	252
16.5.3	无穷 for 循环的中断	252
16.5.4	空循环的用法	252
16.6	while 循环语句	252
16.7	do while 循环	254
16.8	循环嵌套	255
16.9	循环中断	257
16.10	continue 语句	259
16.11	goto 语句	260

第十七章 函 数

17.1 函数的初步概念	262
17.1.1 说明与定义	262
17.1.2 说明与原型	262
17.1.3 定 义	263
17.1.4 形参说明	264
17.1.5 函数调用与参数转换	264
17.2 return 语句	265
17.2.1 从一个函数中返回	265
17.2.2 返回值	266
17.2.3 函数返回非整型值	268
17.3 有关函数原型的进一步说明	271
17.3.1 参数不匹配	271
17.3.2 使用头文件	272
17.3.3 无任何参数的函数原型	272
17.3.4 有关旧式 C 程序	272
17.4 作用域规则	273
17.4.1 局部变量	273
17.4.2 形式参数	275
17.4.3 全局变量	275
17.4.4 有关作用域的最后一个例子	276
17.5 有关函数的参数和自变量的更详尽说明	277
17.5.1 赋值调用和赋地址调用	277
17.5.2 一个赋地址调用的建立	278
17.5.3 数组与函数调用	279
17.6 argc, argv 和 env —— main 中的参数	282
17.7 从 main() 中返回值	284
17.8 递 归	285
17.9 参数说明的传统形式和现代形式的比较	287
17.10 对一些影响函数的效率和实用性问题的讨论	287
17.10.1 参数和通用函数	288
17.10.2 效 率	288
17.11 库函数	288
17.12 改变程序的执行流程	291
17.12.1 使用 exit() 和 abort() 函数	291
17.12.2 使用 system(), exec...(), 和 spawn() 函数	292
17.13 使用可变参数表	293
17.13.1 设计可变参数表	294

17.13.2 使用 va...()函数	294
----------------------------	-----

第十八章 指 针

18.1 指针的语法规则	299
18.1.1 什么是指针	299
18.1.2 指针说明	300
18.1.3 指针与常量	300
18.1.4 指针算术运算	301
18.1.5 指针转换	302
18.2 指针是地址	302
18.3 指针变量	302
18.4 指针操作符	302
18.5 指针表达式	304
18.5.1 指针赋值	304
18.5.2 指针运算	305
18.5.3 指针比较	306
18.6 指针和数组	306
18.6.1 索引指针	307
18.6.2 指针和字符串	307
18.6.3 如何得到一个数组元素的地址	308
18.6.4 指针数组	309
18.6.5 一个使用数组和指针的实例	311
18.7 指针的指针	314
18.8 指针的初始化	315
18.9 指针的一些问题	316
18.9.1 使用 C 语言的间接操作符和取地址操作符	317
18.9.2 使用数组和串	319
18.10 使用指向函数的指针	325
18.10.1 指向函数的指针说明和初始化	325
18.10.2 利用指针引用某调用函数	326
18.11 在动态内存中使用指针	329
18.11.1 C 语言程序和动态内存	330
18.11.2 使用动态存储	331

第十九章 数组、结构、位域、联合和枚举

19.1 高级数据类型的语法规则	338
19.1.1 数 组	338
19.1.2 结 构	338
19.1.3 位 域	342

19.1.4	联合	342
19.1.5	枚举	343
19.2	数组	344
19.2.1	一维数组	344
19.2.2	字符串	346
19.2.3	二维数组	351
19.2.4	多维数组	353
19.2.5	数组初始化	353
19.2.6	一个水下搜索游戏	355
19.3	结构	357
19.3.1	访问结构元素	359
19.3.2	结构数组	359
19.3.3	结构赋值	366
19.3.4	将结构传递给函数	367
19.3.5	结构指针	368
19.3.6	结构内部的数组和结构	372
19.4	位域	372
19.5	联合(union)	375
19.6	枚举	379
19.7	使用 sizeof 来确保可移植性	381
19.8	typedef	382

第二十章 Borland C++ 预处理程序指令

20.1	空指令 #	386
20.2	#define 与 #undef 指令	386
20.2.1	简单的 #define 宏	386
20.2.2	#undef 指令	387
20.2.3	-D 与 -U 选择项	388
20.2.4	关键字与保护字	388
20.2.5	带参宏	388
20.3	文件包含指令 #include	390
20.3.1	<头名>形式的头文件搜索	391
20.3.2	"头名"形式的头文件搜索	391
20.4	条件编译	391
20.4.1	#if、#elif、#else 和 #endif 条件指令	391
20.4.2	defined 运算符	392
20.4.3	#ifdef 和 #ifndef 条件指令	392
20.5	#line 行控制指令	393
20.6	#error 指令	394

20.7	#pragma 指令	394
20.7.1	#pragma argsused	395
20.7.2	#pragma exit 与 #pragma startup	395
20.7.3	#pragma inline	395
20.7.4	#pragma option	396
20.7.5	#pragma saveregs	397
20.7.6	#pragma warn	397
20.8	预定义的宏	398

第三部分 中级 Borland C++ 编程技术

第二十一章 文件输入输出

21.1	两个预处理指令	402
21.1.1	#define 指令	402
21.1.2	#include 指令	404
21.2	文件与流	404
21.3	流(streams)	404
21.3.1	文本流	404
21.3.2	二进制流	405
21.3.3	文件	405
21.4	概念和实际	405
21.5	控制台 I/O	405
21.5.1	字符读写	406
21.5.2	字符串读写	407
21.6	控制台格式化 I/O	407
21.6.1	printf() 函数	408
21.6.2	scanf() 函数	409
21.7	缓冲型 I/O 系统(ANSI 型 I/O 系统)	412
21.7.1	文件指针	412
21.7.2	打开文件	412
21.7.3	写字符	414
21.7.4	读字符	414
21.7.5	feof() 的使用	415
21.7.6	关闭文件	415
21.7.7	ferror() 和 rewind() 函数	415
21.7.8	fopen(), getc(), putc() 和 fclose() 函数的用法	416
21.7.9	getw() 和 putw() 函数的使用	418
21.7.10	fgets() 和 fputs() 函数	418

21.7.11	fread()和fwrite()函数	418
21.7.12	fseek()函数和随机访问I/O	420
21.7.13	标准流	422
21.7.14	fprintf()和fscan()函数	422
21.7.15	删除文件	425
21.8	非缓冲型I/O——UNIX型文件系统	425
21.8.1	open(),creat()和close()函数	426
21.8.2	read()和write()函数	427
21.8.3	unlink()函数	428
21.8.4	随机访问文件和lseek()函数	429
21.9	理解I/O概念	430
21.9.1	文件与设备	430
21.9.2	文件与流	432
21.9.3	文本流和二进制流	433
21.10	利用标准流进行I/O	434
21.10.1	使用格式化I/O函数	434
21.10.2	scanf()函数	438
21.10.3	printf()函数	441
21.10.4	使用字符I/O函数	443
21.11	使用文件控制函数	447
21.11.1	开文件、关文件和控制文件	447
21.11.2	控制文件缓冲区	450
21.12	使用直接文件I/O函数	451
21.12.1	理解直接I/O概念	452
21.12.2	读写直接文件	453
21.13	使用文件定位函数	458
21.13.1	得到当前文件位置	458
21.13.2	建立一个新文件位置	459
21.14	处理文件I/O错误	461
21.14.1	查出文件I/O错误	461
21.14.2	显示和清除文件I/O错误	461

第二十二章 屏幕文本和图形程序设计

22.1	图形系统和要素	463
22.1.1	视频模式	463
22.1.2	窗口和视区	463
22.1.3	在文本模式下编程	464
22.1.4	在图形模式下编程	468
22.2	Borland C++图形程序设计	478

22.2.1	基本正文模式函数	178
22.2.2	Borland C++的图形子系统简介	487
22.3	IBM/PC的文本方式	495
22.3.1	PC显示器适配器和屏幕	495
22.3.2	视频缓冲区I/O	496
22.4	控制文本屏幕	497
22.4.1	使用文本方式控制函数	497
22.4.2	使用直接控制台I/O以获得高性能	499
22.5	使用窗口函数	500
22.6	了解IBM-PC的图形方式	503
22.6.1	象素点与调色板	504
22.6.2	控制图形屏幕	505
22.7	介绍BGI图形库	506
22.7.1	使用画图 and 填充函数	506
22.7.2	控制屏幕和视口	511
22.8	在图形方式下显示文本	511
22.8.1	BGI字库	512
22.8.2	使用图形方式下的文本函数	512

第二十三章 存储模式

23.1	80x86的体系结构	515
23.1.1	段、节以及偏移地址(Offset)	515
23.1.2	CPU的地址寄存器	517
23.2	near指针、far指针和huge指针	518
23.2.1	选择想要的指针大小	518
23.2.2	near、far和huge说明符	520
23.3	六个Borland C++存储模式	522
23.3.1	决定使用哪种存储模式	522
23.3.2	以混合模式编程	523
23.4	创建COM型的可执行程序文件	525
23.4.1	使用COM文件	525

第二十四章 和汇编语言的接口

24.1	混合语言程序设计	528
24.1.1	参数传递顺序	528
24.2	建立从Borland C++对ASM的调用	530
24.2.1	简化的段指令	530
24.2.2	标准段指令	531
24.2.3	定义数据常量和变量	531

24.2.4	定义全局和外部标识符	531
24.3	建立从 ASM 中对 Borland C++ 的调用	533
24.3.1	引用函数	533
24.3.2	引用数据	533
24.4	定义汇编语言过程	534
24.4.1	传递参数	534
24.4.2	处理返回值	534
24.5	寄存器约定	538
24.6	从 ASM 过程中调用 C 函数	538
24.7	伪变量、嵌入汇编和中断函数	540
24.7.1	伪变量	540
24.7.2	嵌入汇编语言	542
24.7.3	中断函数	547
24.8	使用直接插入 (inline) 汇编语言	549
24.8.1	直接插入式汇编环境	549
24.8.2	使用 asm 关键字	549
24.9	与汇编语言例程的接口	553
24.9.1	在 C 程序里调用汇编例程	553
24.9.2	在汇编例程中调用 C 函数	561
24.10	使用中断功能	568
24.10.1	80x86 的中断结构	569
24.10.2	使用 Borland 的中断接口	569
24.11	使用中断处理程序	572
24.11.1	声明中断处理程序函数	573
24.11.2	实现一个时钟中断处理程序	575

第四部分 高级 Borland C++ 程序设计技术

第二十五章 通用 C 语言函数的设计

25.1	函数功能描述	580
25.2	通用函数源程序	582

第二十六章 窗口:丰富的用户界面

26.1	窗 口	588
26.2	视频内存结构	590
26.3	雪花现象与视频回扫	592