西北工业大学
NORTHWESTERN POLYTECHNICAL UNIVERSITY

C程序设计

Programming in C

1011014

主讲：姜学锋，计算机学院

编程任务的接口与版本控制

3、条件编译

4、实用预处理命令

5.3 条件编译

- 通常，源程序中的所有代码行都参与编译，如果希望部分代码只在一定条件时才参与编译，可以使用条件编译命令。

5.3 条件编译

- 使用条件编译，可以针对不同硬件平台和软件开发环境来控制不同的代码段被编译，从而方便了程序的可维护性和可移植性，同时提高了程序的通用性。
- 典型的条件编译是将程序编译分成调试版本“Debug”和发行版本“Release”，一些供程序员调试的代码在Release中没有参与编译，也即在最终的程序可执行文件中不包含这些调试代码。

5.3.1 #define定义条件

- 条件编译使用宏定义条件，其命令形式为：

```
#define 条件字段
```

- 或

```
#define 条件字段 常量表达式
```

5.3.1 #define定义条件

- 例如：

```
1 #define DEBUG
2 #define WINVER 0x0501
```

- 第1行表示DEBUG已经定义，第2行表示WINVER已经定义且值为0x0501。

5.3.1 #define定义条件

- 主流的编译器系统也支持通过编译参数设置条件
- GCC命令行使用参数为

```
>gcc -D条件字段 //等价于 #define 条件字段
>gcc -D条件字段=常量表达式 //等价于 #define 条件字段 常量表达式
```

- VC命令行使用参数为

```
>CL /D条件字段 //等价于 #define 条件字段
>CL /D条件字段=常量表达式 //等价于 #define 条件字段 常量表达式
```

5.3.1 #define定义条件

- 示例

```
>gcc -DDEBUG //等价于 #define DEBUG
>gcc -DWINVER=0x0501 //等价于 #define WINVER 0x0501
```

5.3.2 #ifdef、#ifndef

- #ifdef条件编译命令测试条件字段是否定义，以此选择参与编译的程序代码段，它有两种命令形式。
- ①第一种形式：

```
#ifdef 条件字段
        ..... //程序代码段1
#endif
```

5.3.2 #ifdef、#ifndef

- ②第二种形式：

```
#ifdef 条件字段
        ..... //程序代码段1
#else
        ..... //程序代码段2
#endif
```

- 表示如果条件字段已经被#define定义过，无论是否有值，编译器只编译程序代码段1，否则只编译程序代码段2，程序代码段可以是任意行数的程序或预处理命令。

5.3.2 #ifdef、#ifndef

- 示例

```
#ifdef DEBUG
  printf("x=%d,y=%d,z=%d\n",x,y,z);
#endif
```

- 表示如果DEBUG已经定义则编译printf语句，否则不编译；当printf语句未参与编译时，程序可执行代码中不会有这句。

5.3.2 #ifdef、#ifndef

- 比较与此相似的if语句的含义，例如：

```
if (DEBUG)
  printf("x=%d,y=%d,z=%d\n",x,y,z);
```

- 无论if语句条件满足与否，程序可执行代码中是肯定有printf语句指令的，if语句条件用来决定是否执行它。

5.3.2 #ifdef、#ifndef

- #ifndef条件编译命令测试条件字段没有被定义过，以此选择参与编译的程序代码；它也有两种命令形式，形式如同#ifdef，但作用与#ifdef相反。

5.3.2 #ifdef、#ifndef

- 下面代码测试是否使用VC编译器且为控制台程序，如果是则编译程序代码段：

```
#ifdef _MSC_VER //如果是Visual C++编译器，其内部已定义
#ifndef _CONSOLE //Visual C++编译器根据控制台编译参数内部已定义
   ..... 程序代码段
#endif
#endif
```

5.3.3 #if-#elif

- #if条件编译命令根据表达式的值选择参与编译的程序代码，其命令形式为：

```
#if 常量表达式
        ..... //程序代码段1
#else
        ..... //程序代码段2
#endif
```

5.3.3 #if-#elif

- 条件编译命令中
- “#ifdef 条件字段”与“#if defined 条件字段”是等价的，“#ifndef 条件字段”与“#if !defined 条件字段”是等价的，“#ifdef 条件字段”与“#if define(条件字段)”是等价的。

5.3.3 #if-#elif

- 可以使用嵌套的#if条件编译命令#if-#elif，命令形式为：

```
#if 常量表达式1
        ..... //程序代码段1
#elif 常量表达式2
        ..... //程序代码段2
#else
        ..... //程序代码段3
#endif
```

- 其中#elif分支可以有多项。

5.3.3 #if-#elif

- 下面代码测试是否使用GCC编译器且版本大于3.0，如果是则编译程序代码段：

```
#ifdef __GNUC__ //如果是GCC编译器，其内部已定义
#if (__GNUC__ >= 3) //编译器是GCC3.0以上
   ......//程序代码段
#endif
#endif
```

5.3.3 #if-#elif

- 下面代码根据Windows操作系统的版本选择相应的程序代码段进行编译：

```
#if (WINVER >= 0x0501) //在Windows XP及以上系统
.......//程序代码段1
#elif (WINVER == 0x0500) //在Windows 2000系统
.......//程序代码段2
#else //在Windows 98系统
.......//程序代码段3
#endif
```

- WINVER已经在编译器内部事先定义过。

5.3.3 #if-#elif

【条件编译举例】

一个球从100米高度自由落下，每次落地后反弹回原高度的一半，再落下。求它在第10次落地时，共经过多少米？第10次反弹多高？用“DEBUG”和“RELEASE”分别表示调试、正式版本，编写条件编译。正式版本直接计算结果，调试版本则还要输出球每次落地反弹的数据便于调试中间过程。

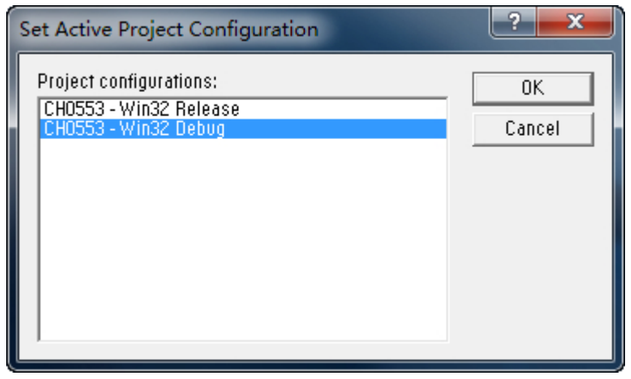
5.3.3 #if-#elif

例5.53

```
1 #include <stdio.h>
2 int main()
3 {
4   double sn=100.0, hn=sn/2;
5   int n;
6   for(n=2;n<=10;n++) {
7     sn=sn+2*hn; //第n次落地时共经过的米数
8     hn=hn/2; //第n次反跳高度
9 #ifdef _DEBUG
10    printf("sn=%lf,hn=%lf\n",sn,hn);
11 #endif
12  }
13  printf("the total of road is %lf\n",sn);
14  printf("the tenth is %lf meter\n",hn);
15  return 0;
```

5.3.3 #if-#elif

- 1. Visual C++ 6.0添加调试宏的方法
- 在Visual C++ 6.0中，_DEBUG即是工程项目“Project Configuration”的Debug版本由系统设定的调试宏。



5.3.3 #if-#elif

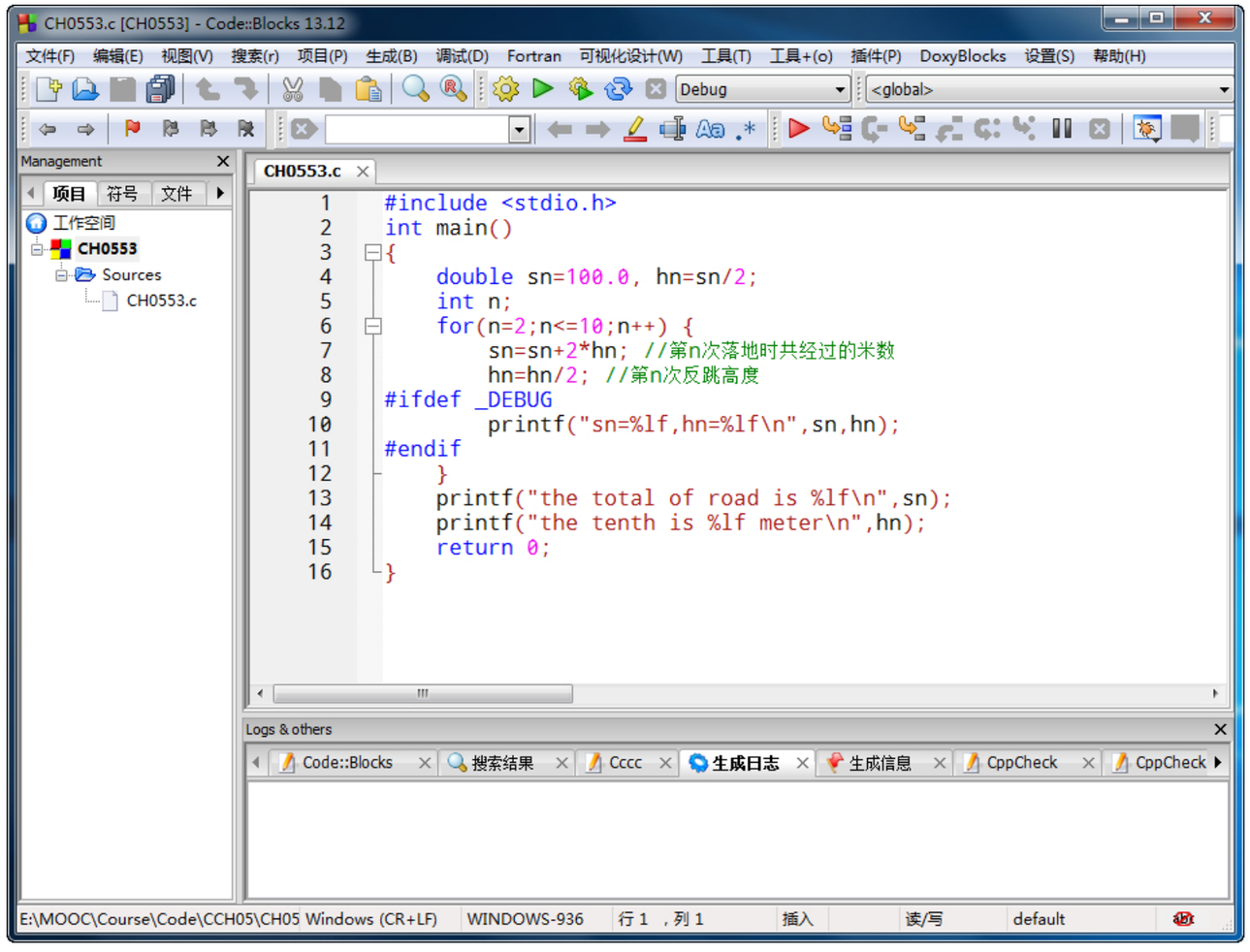
- 2. Code::Blocks添加调试宏的方法
- （1）在GCC中，可以从GCC参数定义宏，使用-D选项即可。例如定义宏_DEBUG。

```
>gcc -D _DEBUG CH0553.c -o CH0553.exe
```

- （2）但是在Code::Blocks中，也可以指定-D选项的宏值

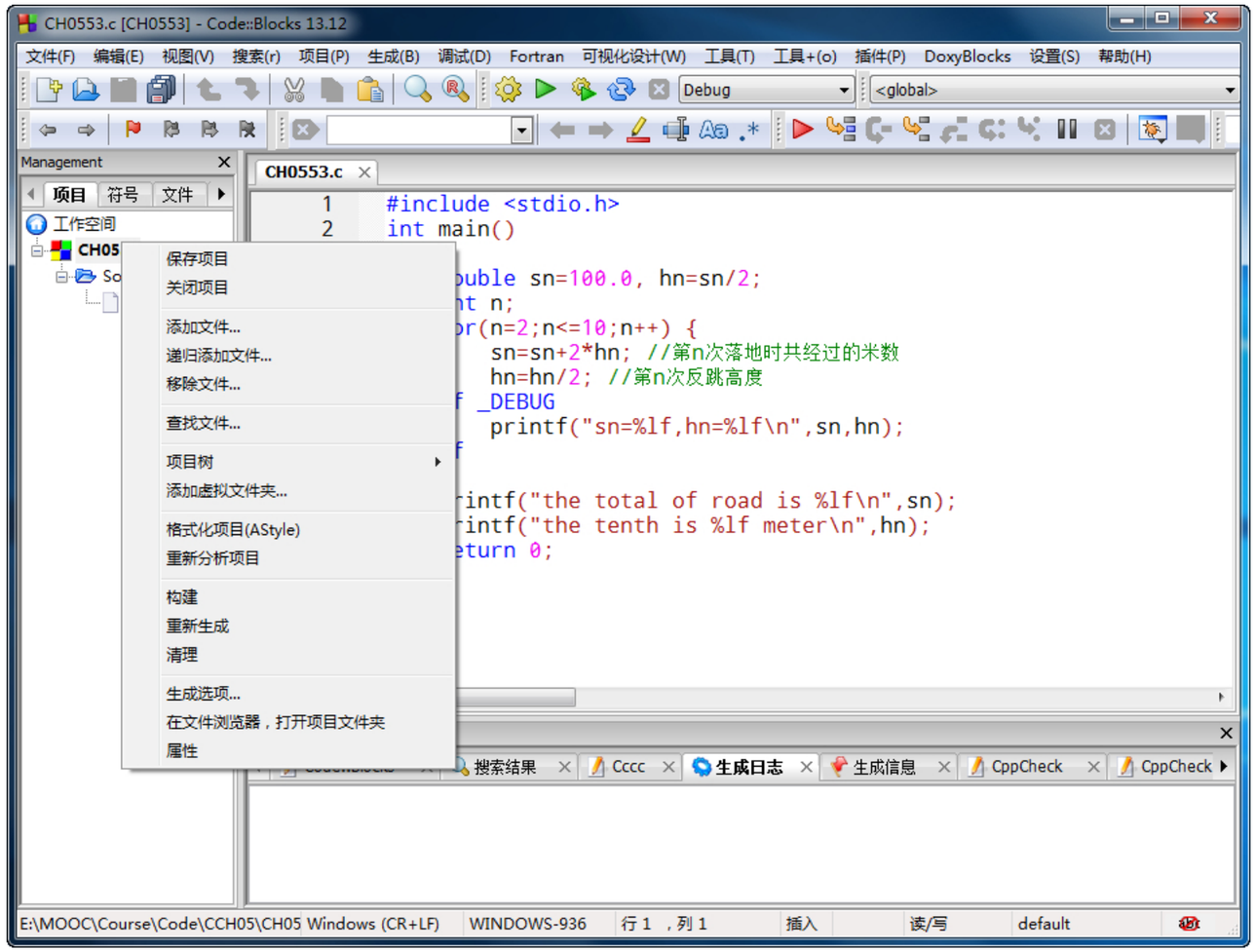
5.3.3 #if-#elif

- 选择项目名（如CH0553）
- 一>右键



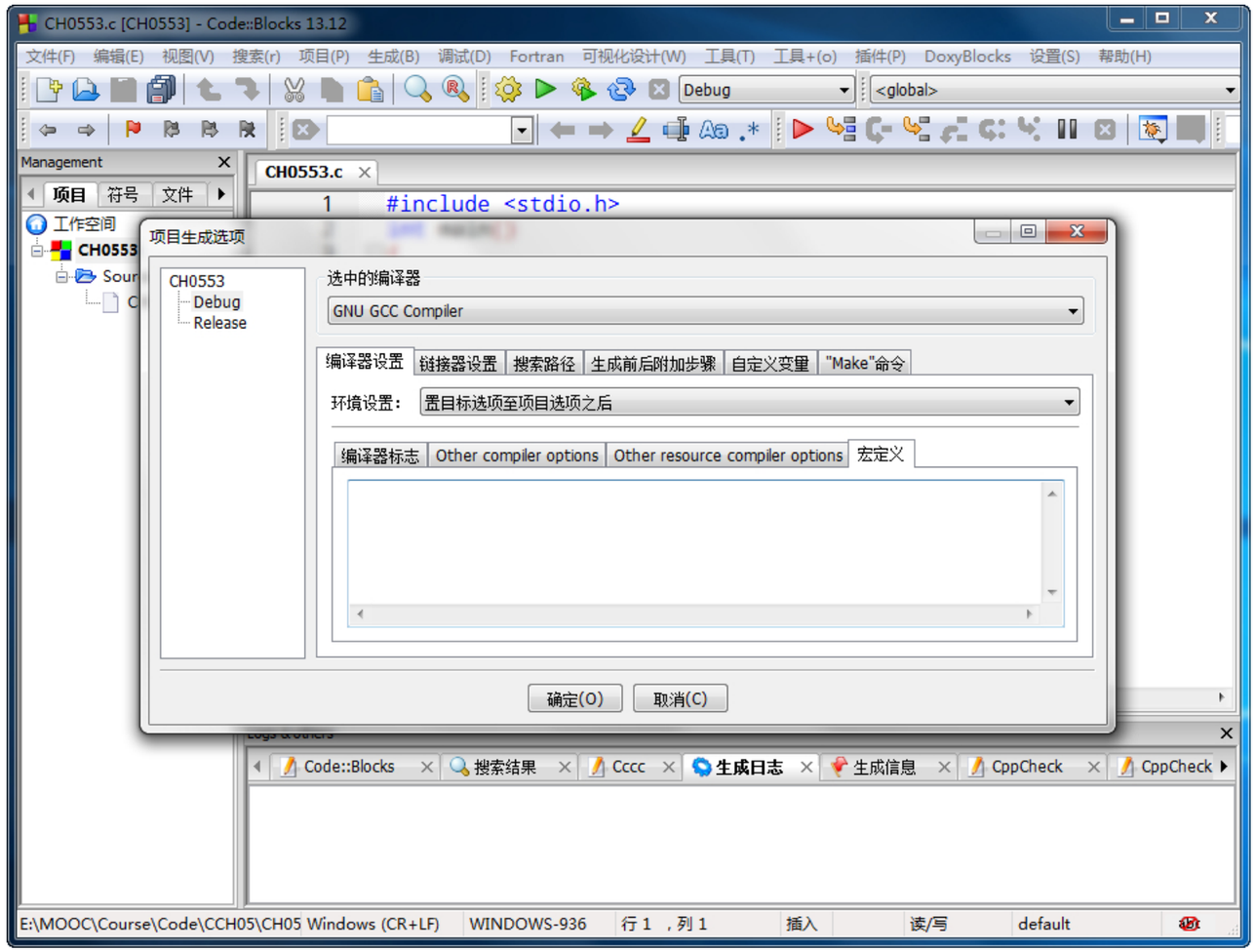
5.3.3 #if-#elif

- 单击“构建选项”



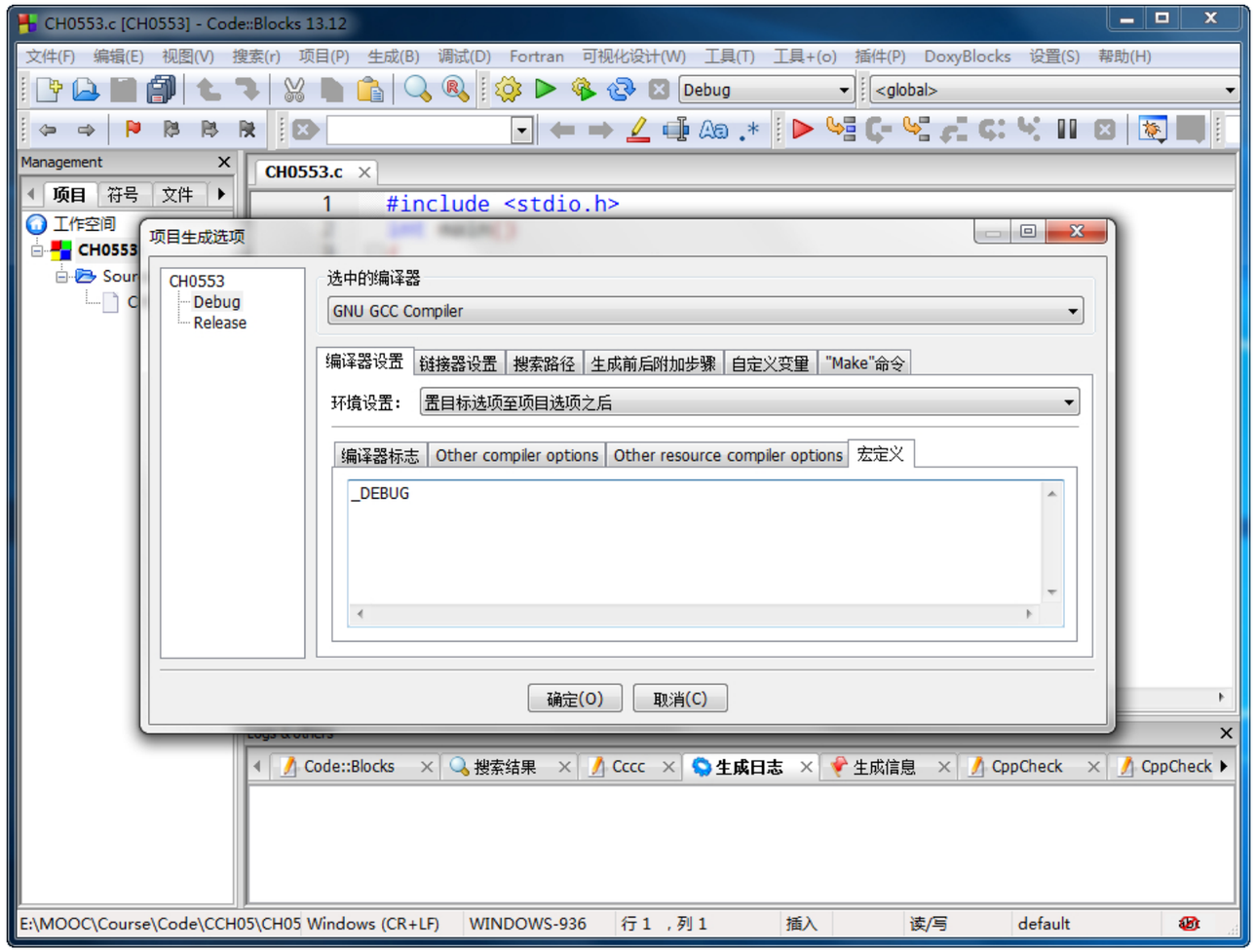
5.3.3 #if-#elif

- 如图
- 选择右边的“#defines”选项



5.3.3 #if-#elif

- 如图
- 然后加入自己的宏值（如_DEBUG）即可



5.3.3 #if-#elif

例5.53

```
1 #include <stdio.h>
2 int main()
,sn,hn);


is %lf\n",sn);
eter\n",hn);
```

程序运行屏幕

```
the total of road is 299.609375
the tenth is 0.097656 meter
```

5.3.3 #if-#elif

例5.53

程序运行屏幕

```
sn=200.000000,hn=25.000000
sn=250.000000,hn=12.500000
sn=275.000000,hn=6.250000
sn=287.500000,hn=3.125000
sn=293.750000,hn=1.562500
sn=296.875000,hn=0.781250
sn=298.437500,hn=0.390625
sn=299.218750,hn=0.195313
sn=299.609375,hn=0.097656
the total of road is 299.609375
the tenth is 0.097656 meter
```

5.4 其他命令

- 1. #error
- #error命令的作用是编译时输出错误信息，命令形式为：

```
#error 错误信息
```

- 如果预处理器遇到一个#error命令，它将显示错误信息并且停止编译过程。
- 实际编程中，#error命令一般与条件编译一起使用，用于测试正常编译中出现的异常现象。

5.4 其他命令

- 例如当不是32位Windows 操作系统时，给出编译错误的预处理命令为：

```
#ifndef _WIN32 //未定义32位Windows
  #error _WIN32 must be defined
#endif
```

5.4 其他命令

- 2．#pragma
- #pragma命令的作用是设定编译器的状态或者是指示编译器完成一些特定的动作，命令形式为：

```
#pragma token-string
```

- 其中token-string表示一条编译器要处理的命令，如果该命令存在则触发执行，否则忽略。

5.4 其他命令

- #pragma命令使编译器能够在保持与C语言完全兼容的情况下，针对计算机或操作系统专有的特征，作出不同的编译处理。例如#pragma once表示文件被包含一次。
- 下面代码仅用于VC编译器，指示连接库文件winmm.lib。

```
#pragma comment(lib, "winmm.lib") //VC连接winmm.lib库文件
```

5.4 其他命令

- 3．#line
- #line命令的作用是改变程序行编号的方式，命令形式为：

```
#line 常量值 "filename"
```

- “filename”可以省略。

- #line命令强制编译器按指定的行号对源程序代码重新编号，说明行号来自文件filename，由常量值开始。行号必须在1～32767之间。

5.4 其他命令

- #line命令会修改预定义宏__FILE__和__LINE__为新值。例如：

```
#line 151 "copy.c"
printf( "line %d, file(%s)\n",__LINE__, __FILE__);
```

- 输出

```
line 151, file(copy.c)
```


程序设计