31.传媒专业群教师（交互设计与研发方向）岗位试讲内容

注意事项：

1.每位考生试讲时间为8分钟；

2.试讲统一采用PPT讲授方式（自备U盘，如因U盘打不开课件，责任自负，U盘不能用考生姓名命名）；

3.试讲的考生在候考室抽签结束后在教案封面填写抽签号提交教案打印件（一式7份）给工作人员。教案不能透露任何个人信息，考生不得穿制服、单位工作服或有明显文字或图案标识的服装参加面试，凡透露个人信息的考生，扣减面试成绩的5%—20%，情节严重的，取消面试成绩。

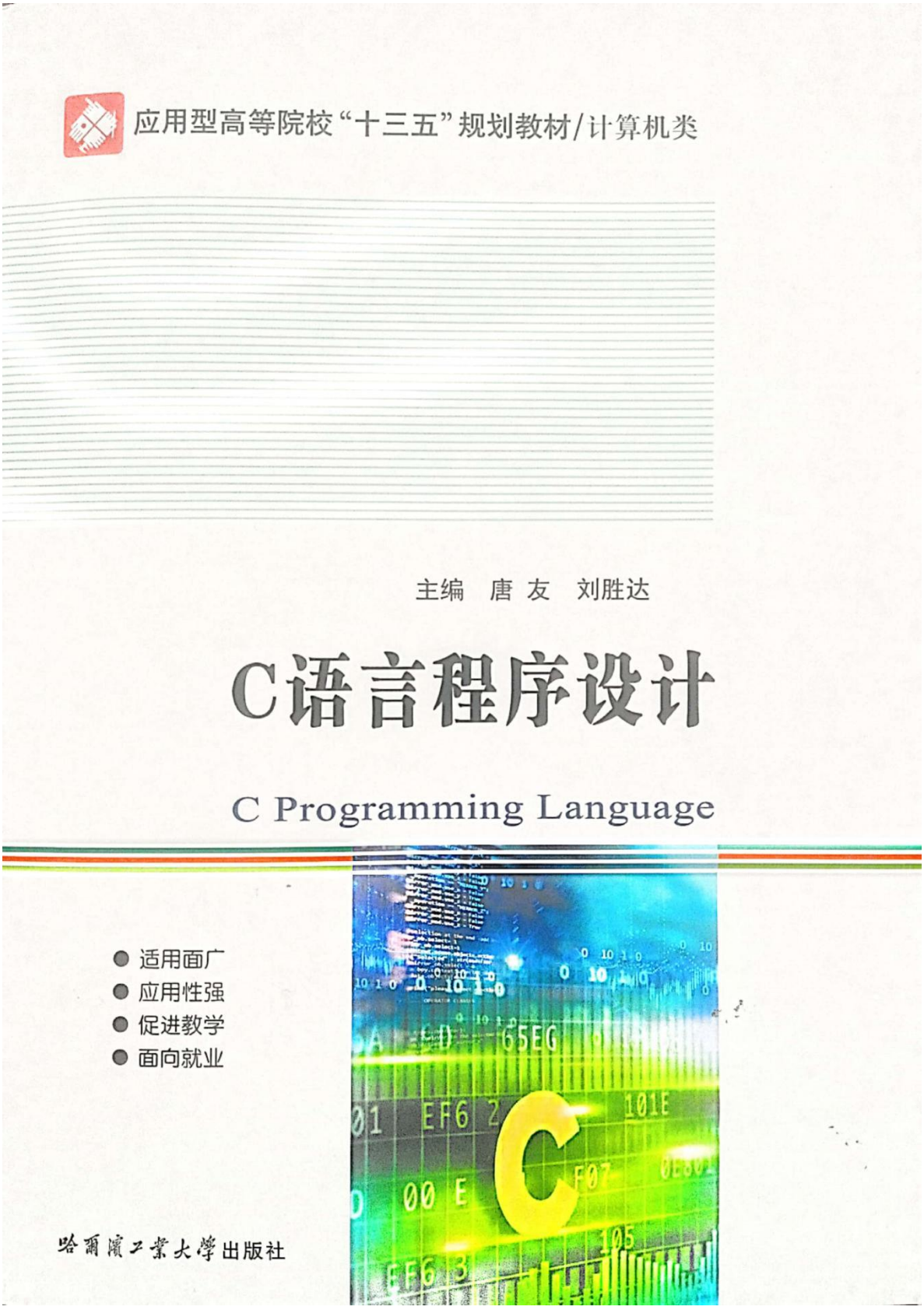
教学内容：第9章 编译预处理和动态存储分配

9.2 宏定义命令

教学重点：说明宏定义命令的使用方法,可自备教具及自备案例。

教材信息：教材名称《C语言程序设计》，哈尔滨工业大学出版社，2020.06出版，唐友、刘胜达主编。

教材封面


应用型高等院校“十三五”规划教材/计算机类
主编 唐 友 刘胜达
C语言程序设计
C Programming Language
适用面广
应用性强
促进教学
面向就业
哈爾濱工業大學出版社

教学内容：第 9 章 编译预处理和动态存储分配

9.2 宏定义命令

图 9-2 例 9-1 运行结果

9.2 宏定义命令

宏定义命令是最为常用的预处理命令之一。在 C 语言中有两种宏定义命令，一种为不带参数的宏定义，另一种为带参数的宏定义。

1. 不带参数的宏定义

在 C 语言程序中，经常使用到一些常量，如 3.14 等。这些常量可能在程序中经常出现，为了避免出现输入错误，可以使用不带参数的宏定义来定义这些常量，其语法格式如下：

```
#define 标识符 字符串
```

例如：

```
#define PI 3.14
```

上述代码中的“#”表示该语句为预处理命令，“define”为宏定义命令，“PI”为宏名，“3.14”为宏体。值得注意的是，由于宏名是标识符的一种，因此其命名规则与标识符一样，只不过宏名习惯上一般用大写字母表示，且与字符串之间用空格分离。宏名的有效范围是从宏定义开始至源程序结束，可以使用宏命令#undef 提前结束宏名的作用域。字符序列可以是常数、表达式及各种格式串等。没有特殊需要，不要在宏定义行末尾加分号，否则将会把分号理解为字符串中的字符。另外，在宏定义时可以引用之前已经定义的宏名，宏替换是层层替换的。

例如：

```
#define PI 3.14
#define DIAM 10
#define C PI * DIAM
int main(int argc, char *argv[])
{
    printf("C=%.1f\n",C);
    return 0;
}
```

上述代码在预处理时，“C”被宏替换为“3.14 * 10”，结果输出为“31.4”。值得注意的是，若宏体为表达式，则根据情况可能需要加括号，否则可能会出现意想不到的错误。

例如：

```
#define PI 3.14
#define R 5
```

```
#define DIAM R + R
#define C PI * DIAM
int main(int argc, char *argv[])
{
    printf("C = % .1f\n",C);
    return 0;
}
```

上述代码在预处理时，“C”被宏替换为“3.14 * 5 + 5”，输出结果为“20.7”。因此，上述代码可以改为：

```
#define PI 3.14
#define R 5
#define DIAM (R + R)
#define C PI * DIAM
int main(int argc, char *argv[])
{
    printf("C = % .1f\n",C);
    return 0;
}
```

例9-2　编写程序，利用宏定义命令定义字符串并输出。

```
#include "stdafx.h"
#define S1 "Hello"
#define S2 "C"
#define S3 "Free!"
int main(int argc, char *argv[])
{
    printf("% s % s % s\n",S1,S2,S3);
    return 0;
}
```

上述代码运行结果如图9-3所示。

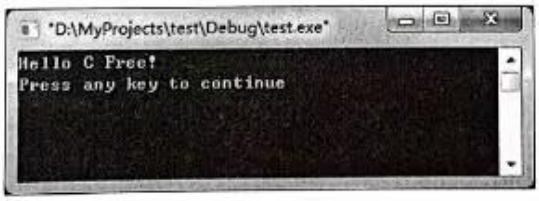


图9-3　例9-2运行结果

2. 带参数的宏定义

不带参数的宏定义只能完成较为简单的宏替换工作，若希望完成更为复杂的宏替换工作，可以使用带参数的宏定义。带参数的宏定义语法格式如下：

```
#define 标识符(形参列表) 字符串
```

调用带参数宏的语法格式如下：

```
标识符(实参列表)
```

例如：

```
#define S(r) 3.14 * r * r
printf("% .2f\n",S(10));
```

上述代码在预处理时，用实参“10”替换形参“r”，“S(10)”被宏替换为“3.14 * 10 * 10”，输出结果为“314.00”。值得注意的是，在带参数的宏定义中，由于形参不分配内存单元，因此不需要定义其类型。而在带参数的宏调用中，实参是具体的值，因此必须定义其类型。另外，在带参数的宏定义中，根据情况同样需要增加括号，否则可能出现意想不到的错误。

例如：

```
#define S(r) 3.14 * r * r
printf("% .2f\n",S(5 +5));
```

上述代码在预处理时，用实参“5 +5”替换形参“r”，“S(5 +5)”被宏替换为“3.14 * 5 +5 * 5 +5”，输出结果为“45.70”。因此，上述代码可以改为：

```
#define S(r) 3.14 * (r) * (r)
printf("% .2f\n",S(5 +5));
```

上述代码在预处理时，用实参“5 +5”替换形参“r”，“S(5 +5)”被宏替换为“3.14 * (5 +5) * (5 +5)”，输出结果为“314.00”。另外，带参数的宏和函数虽然有相似之处，但也是有区别的，具体主要包括以下几点：

(1)函数先计算实参表达式再传递给形参，而带参数的宏直接进行宏替换并不计算实参表达式。

(2)函数在运行时进行值处理且分配临时内存空间，而带参数的宏直接进行宏替换并不进行值处理且不分配内存空间。

(3)函数实参和形参均需要定义类型且必须一致，而带参数的宏只是用标识符表示，并不需要定义类型。

(4)函数被调用时存在一定的开销，而带参数的宏被调用时并没有额外的开销。

例 9 -3　编写程序，分别用带参数的宏定义和函数求绝对值。

```
#include "stdafx.h"
#define ABS(number) ((number) > =0? (number): -(number))
int abs(int);
int main(int argc, char * argv[])
{
    printf("宏定义方法计算 -5 的绝对值为:% d\n",ABS( -5));
    printf("函数的方法计算 -5 的绝对值为:% d\n",abs( -5));
    return 0;
}
int abs(int number)
{
    return number > =0 ? number : -number;
}
```

上述代码运行结果如图 9 -4 所示。

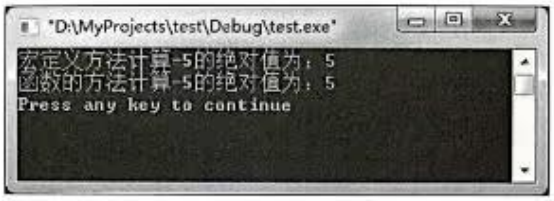


图 9-4　例 9-3 运行结果

9.3　文件包含

文件包含是 C 语言最为常用的预编译处理操作之一，其作用是在预编译处理阶段将另一个 C 语言源程序文件的内容包含到当前 C 语言源程序中，其语法格式如下：

```
#include <文件名>
```

或者：

```
#include "文件名"
```

例如：

```
#include "stdafx.h"
```

上述代码实现了对名为“stdio.h”的文件的文件包含，在包含该文件后，就可以在当前 C 语言源程序中使用输入输出类库函数，如 printf()函数等。值得注意的是，若用“<>”符号则预处理器会在系统指定路径下寻找该文件，若用双引号则预处理器会先在源程序当前路径下寻找该文件，如果找不到再到系统指定路径下查找。

在 C 语言中有很多以“.h”为扩展名的文件，一般称其为头文件，其内容主要是使用相应库函数所需要的函数定义和宏定义等。一个文件包含语句仅能指定一个头文件，若需要包含多个头文件则需要使用多条相应的文件包含语句。文件包含命令也可以嵌套使用，即在一个被包含的头文件中可以利用“#include”命令包含其他头文件，依此类推。另外，在 C 语言源程序中也可以包含自定义头文件。

例 9-4　编写程序，调用自定义头文件比较两个整数的大小，输出较大的整数。

(1)在“main.c”源程序当前路径下新建“max.h”文件，内容如下：

```
int max(int,int);
```

(2)在源程序当前路径下新建“max.c”文件(注意，若是 VC6.0 环境则文件名称为“max.cpp”)，内容如下：

```
#include < stdio.h >                    //若是 VC6.0 环境则该行代码改为#include
                                          "stdafx.h"
int max(int number1,int number2)|
    return number1 > number2 ? number1 : number2;
|
```

(3)在“main.c”源程序中录入以下代码。

```
#include "stdafx.h"
#include "max.h"
int main(int argc, char *argv[])
```