Ch-00
预处理命令

0.1 编译预处理

□ 基本概念

编译预处理：在编译之前，对源程序中以#开头的命令做一些处理， 生成扩展C++程序。

□ 种类：

- 宏定义　　#define
- 文件包含　#include
- 条件编译　#if #else #endif等

□ 格式：

- “#”开头；
- 占单独书写行；
- 语句尾不加分号。

0.1.1 宏定义

□ 不带参数的宏定义

- **一般形式：#define　宏名　宏体**
- **功能：用宏名代替宏体。**
- **说明；宏名中不允许有空格，而且必须遵循标识符命名规则。宏名一般用大写字母。**

```
例  #define    YES     1
    #define    NO      0
    #define    PI      3.1415926
    #define    OUT     cout << “Hello,World”;
```

- **定义位置：一般在程序开头。**
- **作用域：从命令定义到文件结束。**
  - **#undef 可终止宏名作用域，格式：#undef 宏名。**

```
例  #define  YES  1
    main()  {…}
    #undef  YES
    #define  YES  0
    max()  {…}
```

- **宏展开：预编译时，用宏体替换宏名——不作语法。**

```
例          if(x==YES)          printf("correct!\n");
            else if (x==NO)     printf("error!\n");

展开后：    if(x==1)            printf("correct!\n");
            else if (x==0)      printf("error!\n");
```

- 引号中的内容与宏名相同不需置换。

```
例    #define  PI  3.14159
      printf("2*PI=%f\n", PI*2);
```

宏展开：printf("2*PI=%f\n", 3.14159*2);

- 宏定义中使用必要的括号()。

```
例    #define  WIDTH  80
      #define  LENGTH  WIDTH+40
      var = LENGTH*2;
```

宏展开：var = 80+40*2;

```
例    #define  WIDTH  80
      #define  LENGTH  (WIDTH+40)
      var = LENGTH*2;
```

宏展开：var = (80+40)*2;

- 宏定义可嵌套，不能递归

```
例  #define  MAX   MAX+10  (×)
```

□ 带参数的宏定义

- **一般形式：#define 宏名(参数列表) 宏体**
  - **注意：“宏名”与“(参数列表)”中间不能有空格。**

```
例 #define S (r) PI*r*r
   相当于定义了不带参数的宏S，代表字符串“(r) PI*r*r”
```

- **宏展开：形参用实参置换，其它字符保留。**

```
例 #define S(a,b) a*b
   …
   area = S(3,2);
宏展开：area = 3*2;
```

- **宏体及各个形参一般应加括号()。**

```
例 #define POWER(x) x*x
   z=POWER(x+y);
宏展开：z=x+y*x+y;
```

```
一般写成： #define POWER(x) ((x)*(x))
           z=POWER(x+y);
宏展开： z=((x+y)*(x+y));
```

□ 思考：比较宏定义和函数

- **用宏定义和函数实现同样的功能**

```
#define  MAX(x,y)   (x)>(y)?(x):(y)
 …
main()
{   int  a,b,c,d,t;
    …
    t=MAX(a+b,c+d);
     …
}
```

宏展开：t=(a+b)>(c+d)?(a+b):(c+d);

```
int   max(int x,int y)
{  return(x>y?x:y);
}
main()
{    int a,b,c,d,t;
      …
      t=max(a+b,c+d);
       …
}
```

- **带参的宏与函数区别**

	带参数的宏	函数
处理时间	预编译时处理	程序运行时处理
参数类型	无类型问题	定义实参、形参类型
处理过程	不分配内存，不求值，无参数传递，无返回值，只做简单的字符替换	分配内存，先求实参值，再传递给形参，有返回值
程序长度	宏体替换宏名后会使程序变长	程序不会变长
运行速度	占用编译时间，不占运行时间	占用程序运行时间

0.1.2 文件包含

- 功能：一个源文件可将另一个源文件的内容全部包含进来。
- 一般形式：

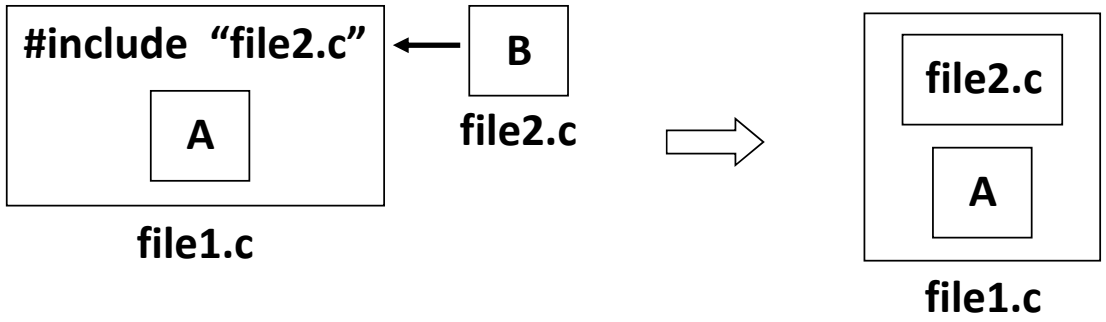
#include “文件名”

或 #include <文件名>

- 说明：<> 直接搜索标准目录；“” 先搜索当前目录，再搜索标准目录。

- 处理过程：

预编译时，用被包含文件的内容取代该预处理命令，再将“包含”后的文件作为一个源文件单位进行编译，得目标文件（.obj ）。



□ 被包含文件的类型

- 源文件(*.cpp)，如：#include “file2.cpp”。
- 头文件
  - ➢ 系统头文件，如：#include <cstdio>。
  - ➢ 用户自定义头文件，如： #include “myhead.h”。

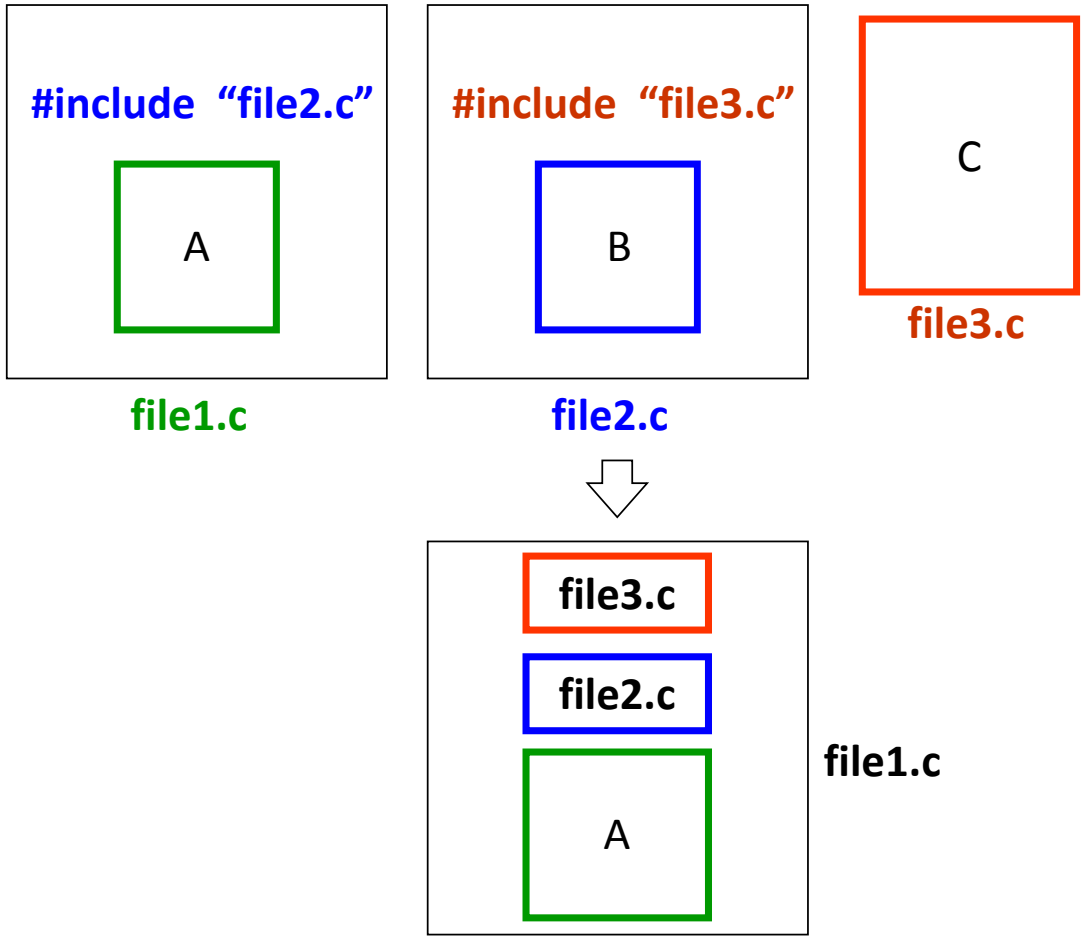
头文件应包含如下内容：

- 类型定义
- 函数原型
- 内联函数声明
- 全局数据声明
- 常量定义
- 包含指令
- 宏定义
- 注释

头文件不能包含以下内容：

- 一般函数定义
- 数据定义

□ 文件包含可嵌套


#include “file2.c”
A
file1.c
#include “file3.c”
B
file2.c
C
file3.c
file3.c
file2.c
A
file1.c

例0.1 文件包含的使用

```c
/* powers.h */
#define  sqr(x)     ((x)*(x))
#define  cube(x)   ((x)*(x)*(x))
#define  quad(x)   ((x)*(x)*(x)*(x))
```

```c
/*powers.c*/
#include <stdio.h>
#include "d:\head\powers.h"
#define  MAX_POWER 10

int
main()
{
    int n;
    printf("number\t exp2\t exp3\t exp4\n");
    printf("----\t----\t-----\t------\n");
    for(n=1; n<=MAX_POWER; n++)
        printf("%2d\t %3d\t %4d\t %5d\n", n, sqr(n), cube(n), quad(n));
    return 0;
}
```

0.1.3 条件编译

- **功能：**

  根据指定的标识符是否被定义过，确定在程序编译阶段编译哪一段程序段。

- **使用形式：**

形式1：

```
#if 表达式
    程序段1
#else
    程序段2
#endif
```

形式2：

```
#ifdef 标识符
    程序段1
#else
    程序段2
#endif
```

形式3：

```
#ifndef 标识符
    程序段1
#else
    程序段2
#endif
```

例0.2 输入字符串，根据需要设置条件编译，使字母改为大写或小写。

```c
#include <stdio.h>
#define LETTER 1    /* 1大写，0小写 */

int
main()
{
    char str[20]="C Language", c;
    int i = 0;
    while((c=str[i]) != '\0')
    {
        i++;
        #if LETTER
            if(c>='a' && c<='z')  c = c-32;
        #else
            if(c>='A' && c<='Z')  c = c+32;
        #endif
        printf("%c", c);
    }
   printf("\n");
   return 0;
}
```

```
C LANGUAGE
```

口 说明：

在调试程序时，常常希望输出一些所需的信息，而在调试完成后不再输出这些信息，可在源程序中插入以下的条件编译段：

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