第9章 预处理命令

- 宏定义
- “文件包含”处理
- 条件编译

★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

§9.1 宏定义

宏定义命令

宏体可缺省,表示宏名定义过或取消宏体

★不带参数的宏定义

- ❖一般形式： #define 标识符 字符串
- ❖功能：用指定标识符(宏名)代替字符串序列(宏体)

```
如  #define   YES   1
    #define   NO    0
    #define   PI    3.1415926
    #define   OUT   printf("Hello,World");
```

- ❖定义位置：任意(一般在函数外面)
- ❖作用域：从定义命令到文件结束
- ❖#undef可终止宏名作用域

格式： #undef 宏名

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

YES原作用域

YES新作用域

例1 使用不带参数的宏定义

```
#include <stdio.h>
#define PI 3.1415926
int main()
{ float l,s,r,v;
  printf("input radius : ");
  scanf("%f",&r);
  l=2.0*PI*r;
  s=PI*r*r;
  v=4.0/3.0*PI*r*r*r;
  printf("l=%10.4f\ns=%10.4f\nv=%10.4f\n",l,s,v);
}
```

运行：

```
input radius: 4↵
l=25.1328
s=50.2655
v=150.7966
```

- 宏名一般用大写字母，与变量区别。
- 使用宏便于修改变量值，提高程序通用性。

例 `#define ARRAY_SIZE 1000`
`int array[ARRAY_SIZE];`

❖宏展开：预编译时,用宏体替换宏名---不作语法检查

```
如          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也不置换

```
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;
```

❖宏定义可嵌套，不能递归

例 #define WIDTH 80
#define LENGTH WIDTH+40
var=LENGTH*2;
宏展开：var= 80+40 *2;

例 #define MAX MAX+10 (×)

例2 在宏定义中引用已定义的宏名

```
#include <stdio.h>
#define R  3.0
#define PI  3.1415926
#define L  2*PI*R
#define S  PI*R*R
int main()
{ printf("L=%f\nS=%f\n",L,S);}
```

运行：
L=18.849556
S=28.274333

★带参数的宏定义

- ❖一般形式： #define 宏名(参数表) 宏体
- ❖功能：进行字符串替换，并进行参数替换

例 #define S(a,b) a*b

.....

area=S(3,2);

宏展开： area=3*2;

不能加空格

- ❖宏展开：形参用实参换，其它字符保留
- ❖宏体及各形参外一般应加括号（）

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

★带参数的宏定义

- ❖一般形式： #define 宏名(参数表) 宏体
- ❖功能：进行字符串替换，并进行参数替换

例 #define S(a,b) a*b

.....

area=S(3,2);

宏展开： area=3*2;

不能加空格

- ❖宏展开：形参用实参换，其它字符保留
- ❖宏体及各形参外一般应加括号（）

例 #define POWER(x) x*x

x=4; y=6;

z=POWER(x+y);

宏展开：z=x+y*x+y;

一般写成： #define POWER(x) ((x)*(x))

宏展开： z=((x+y)*(x+y));

例3 使用带参数的宏

```
#include <stdio.h>
#define PI  3.1415926
#define S(r)  PI*r*r
int main()
{ float a,area;
  a=3.6;   area=S(a);
  printf("r=%f\narea=%f\n",a,area);
}
```

运行：

```
r=3.600000
area=40.715038
```

- 带参宏定义与函数的区别
  - 函数调用时，先求实参表达式的值，再带入形参。
    宏只进行简单字符替换，不求值。
  - 函数调用在程序运行时处理和分配临时内存单元。
    宏展开在编译时进行，不分配内存单元，无值传递和返值。
  - 函数要定义形实参且类型一致，宏无类型，其参数无类型。
  - 函数只有一个返回值，宏可以设法得到几个结果。

例4 使用宏带回几个结果

```
#include <stdio.h>
#define PI  3.1415926
#define CIRCLE(R,L,S,V)  L=2*PI*R;S=PI*R*R;V=4.0/3.0*PI*R*R*R
int main()
{ float r,l,s,v;
  scanf("%f",&r);
  CIRCLE(r,l,s,v);
  printf("r=%6.2f,l=%6.2f,s=%6.2f,v=%6.2f\n",r,l,s,v);
}
```

宏展开后：

```
int main()
{ float r,l,s,v;
  scanf("%f",&r);
  l=2*3.1415926*r;s=3.1415926*r*r;v=4.0/3.0*3.1415926*r*r*r;
  printf("r=%6.2f,l=%6.2f,s=%6.2f,v=%6.2f\n",r,l,s,v);}
```

运行：

3.5 ↵

r=3.50,l=21.99,s=38.48,v=179.59

- 宏展开使源程序变长，函数调用源程序不变长。
- 宏替换不占运行时间，只占编译时间。
  函数调用占运行时间。

例 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);
       .........
}
```

例5 用宏代表输出格式

```
#include <stdio.h>
#define  PR  printf
#define  NL  "\n"
#define  D  "%d"
#define  D1  D  NL
#define  D2  D  D  NL
#define  D3  D  D  D  NL
#define  D4  D  D  D  D  NL
#define  S  "%s"
```

```
int main()
{ int a,b,c,d;
  char string[]="CHINA";
  a=1;b=2;c=3;d=4;
  PR(D1,a);
  PR(D2,a,b);
  PR(D3,a,b,c);
  PR(D4,a,b,c,d);
  PR(S,string);
}
```

运行结果：

```
1
12
123
1234
CHINA
```

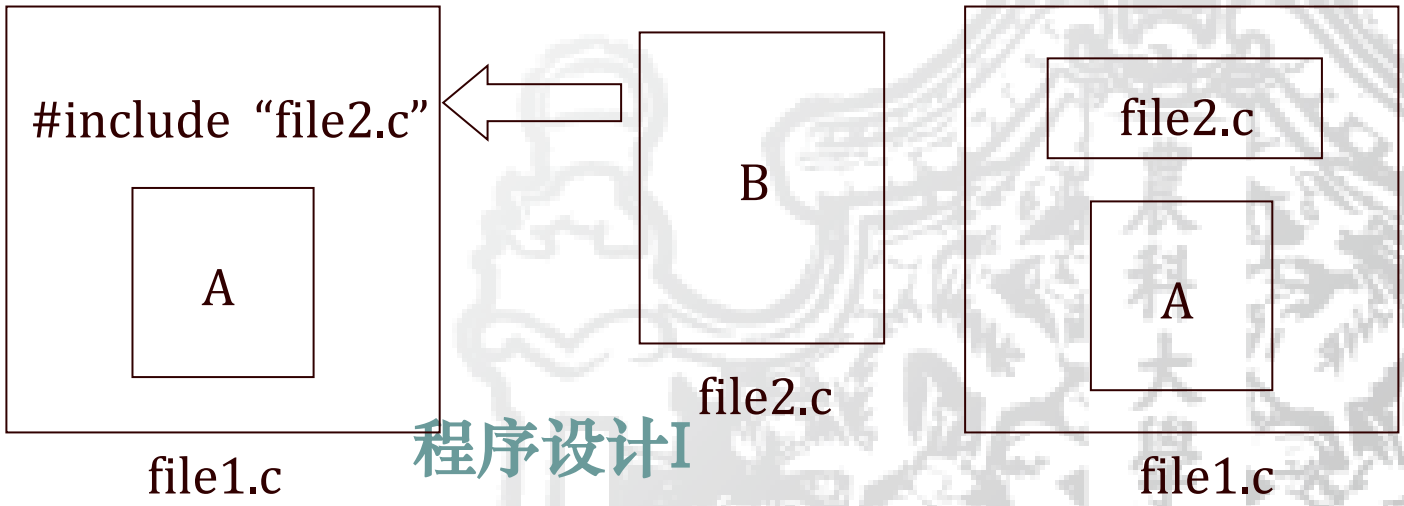
§9.2 “文件包含”处理

★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>

★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



§ 9.2 “文件包

<> 直接按标准目录搜索

“ ” 先在当前目录搜索，再搜索标准目录

可指定路径

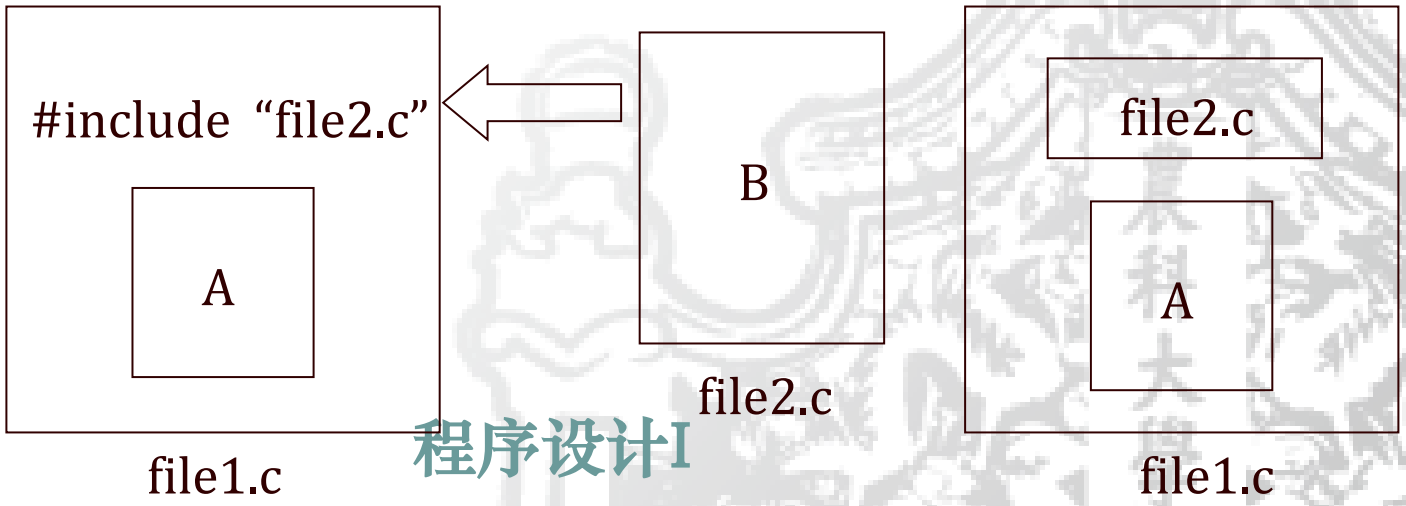
★功能：一个

全部包含进

★一般形式： #include “文件名”

或 #include <文件名>

★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

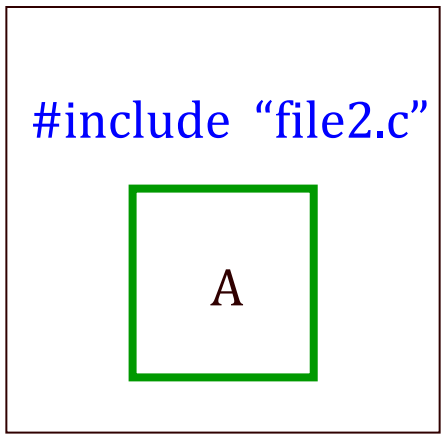


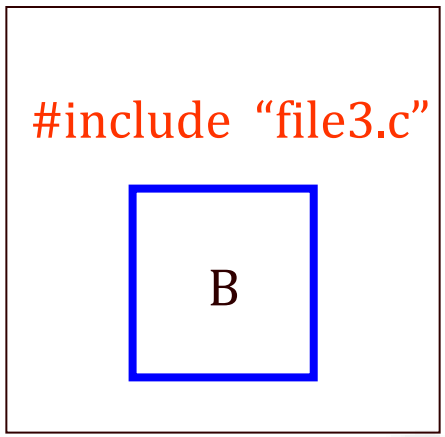
★被包含文件内容

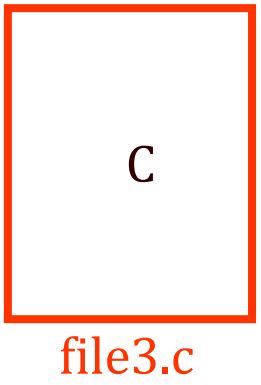
- ❖源文件(*.c)
- ❖头文件(*.h)

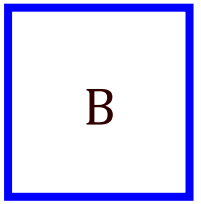
宏定义
数据结构定义
函数说明等

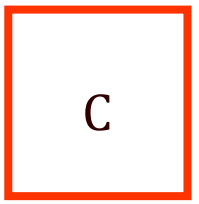
★文件包含可嵌套

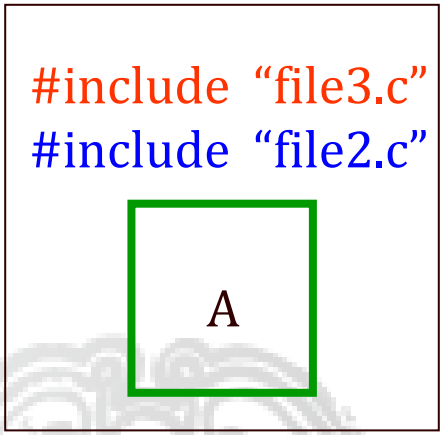












★预编译后已成为一个文件，因此file2.c中的全局静态变量在file1.c中有效，不必再用extern声明。

§ 9.3 条件编译

★功能：当文件中的部分内容在满足一定条件才进行编译

★几种形式：

❖形式1：

- 标识符已被#define命令定义过，则编译程序段1，反之编译程序段2

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

可以没有

例 调试信息的输出

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

例6

```
#include <stdio.h>
#define  LETTER  1
int main()
{ char str[20]="C Language",c;
  int i=0;
  while((c=str[i])!='\0')
      { i++;
        #ifdef  LETTER
           if(c>='a'&&c<='z')  c=c-32;
        #else
           if(c>='A'&&c<='Z')  c=c+32;
        #endif
        printf("%c",c);}
  printf("\n");
}
```

如果 LETTER被定义了，那么久输出大写，否则输出小写

在此例子中，

```
#define  LETTER  1
#define  LETTER  0
#define  LETTER
```

三者产生的结果相同

❖形式2：

- 与形式1相反，标识符未被#define命令定义过，则编译程序段1，反之编译程序段2

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

❖形式3：

- 表达式为真，则编译程序段1，反之编译程序段2

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

例7 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

```
#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");
}
```

★本章要求

- ❖熟悉宏定义与宏展开的区别，宏与函数的区别。
- ❖熟悉文件包含命令#include的作用及预处理方法。
- ❖熟悉条件编译的使用。