**编译预处理**

1. **宏定义**

1、不带参数的宏定义

* 1. 用指定的表示符代表一串字符
  2. 形如： #define PI 3.1415926 【我们就用PI 代替了3.1515926】
  3. 好处
  4. 方便修改，我们不用动一发而牵动全身【如：我们多次用到PI，拿如果我们后期需求变了，我们就只需要改变 PI 的宏定义即可】
  5. 注意
  6. 定义名我们一般用大写
  7. 宏定义不是C语句，所以不用加 ； ，如果有加 ； 则连 ； 一起编译
  8. 不能跨文件使用

2、带参数的宏定义

* + 1. 经过我们上面的讲解我们就有些了解了
    2. 形如： #define 宏名（参数列表） 字符串
    3. 我们举个例子
    4. define S(a, b) a\*b ---> 我们调用S(2, 3) = 2\*3 = 6

**二、文件包含**

1、将另一个文件得内容包含到本文件中 #include “文件名.h” 我们一般是 .h 文件即头文件 [那就是说明是可以进行嵌套的]

2、我们常常把宏定义，全局变量，函数说明，#include 写在头文件，在#include 到main 文件当中来，使得 main 函数尽量简洁

3、代码

(2) 比如我们创建一个 head.h 头文件

#define PI 3.1415926

(3)我们在主函数 main 中 #include “head.h” 我们就可以进行用了

**三、条件编译**

1、格式 # ifdef 和 ifndef

(1) #ifdef xxx

{

}

#else

{

}

#endif // xxx

* 1. 例如我们的程序，有调试和发布，那我们调试的时候需要输出一些信息，发布不需要输出信息

1. 代码

#define DEBUG

int main(){

#ifdef DEBUG

printf("调试阶段");

#else

printf("发布阶段");

#endif // DEBUG

}

1. 格式 #if #else #endif，根据条件来指定执行的代码块
   1. 代码

#if DEBUG

{

}

#else

{

}

#endif // DEBUG

1. **总结**
2. 好处
   * 1. 减少生成目标文件的长度
     2. 增加程序的可移植性（跨平台）例如（\_WIN32=[window系统] \_\_linux\_\_[Linux系统]， 这是系统定义的哦， 不是自己定义的，可以直接用）