1. 宏定义

例：程序preprocessor\_test

#define S 3 + 2

int main(int argc, char\* argv[])

{

int a = 0;

a = S \* 3;

printf("%d\n", a);

return 0;

}

输出的结果a = 9，原因是预编译时将宏名直接替换了字符串，如下所示：

a = 3 + 2 \* 3;

例：程序preprocessor\_test

#define SQUARE(a) ((a) \* (a))

int b = 4;

int c = 0;

c = SQUARE(b++);

printf("%d\n", c);

输出的结果c = 16，展开后c = ((b++) \* (b++))，先执行b \* b，计算c = 16，然后b = 6；

例：程序preprocessor\_test

#define PR printf

#define NL "\n"

#define D "%d"

#define D1 D NL

#define D2 D D NL

int main(int argc, char\* argv[])

{

int a = 0;

PR(D1, a);

PR(D2,a, b);

return 0;

}

输出结果为：

0

04

1. 条件编译

条件编译是对源程序中某段程序通过条件来控制是否参加本次编译。

一般情况下，程序清单中的程序全部都应参加编译。但是在大型应用程序中，可能会出现某些功能不需要的情况，这时可以利用条件编译来选取需要的功能进行编译，以便生成不同的应用程序。

条件编译还可以方便程序的逐段调试。

条件编译命令的三种格式：

1. #if 条件1

程序段1

#elif 条件2

程序段2

#else

程序段n

#endif

其中：条件是常量表达式（编译器在编译时就能计算出结果的整形表达式），其值非0，则条件成立，其值为0，则条件不成立。

执行过程：在编译预处理时，若条件1成立，则编译程序段1，不执行其他的程序段了（即使其他条件也成立）；若条件1不成立，而条件2成立，则编译程序段2，不执行其他的程序段；如果所有的条件都不成立，执行程序段n，如果没有程序段n，则#else可以省略。

例：程序preprocessor\_test1

#define MAX 30

#if MAX > 10

cout<<"MAX大于10"<<endl;

#elif MAX > 20

cout<<"MAX大于20"<<endl;

#else

cout<<"MAX大于30"<<endl;

#endif

执行结果为：MAX大于10

2. #ifdef 宏名

程序段1

#else

程序段2

#endif

其中：宏名是标识符。可以是前面已定义过的宏名，也可以是前面没定义过的宏名。

执行过程：在编译预处理时，判断宏名是否在前面已定义过。若前面已定义过，则编译程序段1，不编译程序段2；如前面没有定义过，则编译程序段2，不编译程序段1。

如果没有程序段2，则#else可以省略。

例：程序preprocessor\_test1

#define USER 0

#ifdef USER

cout<<"宏USER之前已定义过"<<endl;

#else

cout<<"宏USER之前没有定义过"<<endl;

#endif

执行结果为：宏USER之前已定义过

3. #ifndef 宏名

程序段1

#else

程序段2

#endif

其中：宏名是标识符。可以是前面已定义的宏名，也可以是前面没有定义的宏名。

执行过程：在编译预处理时，判断宏名是否在前面已定义过。若前面未定义过，则编译程序段1，不编译程序段2；若前面已定义，则不编译程序段1，编译程序段2.

如果没有程序段2，则#else可以省略。

例：

#define USER 0

#ifndef USER

cout<<"宏USER之前没有定义"<<endl;

#else

cout<<"宏USER之前已定义过"<<endl;

#endif

执行结果：宏USER之前已定义过

第2种条件编译和第3种条件编译判断宏是否定义过后，执行的选择正好相反。

4. #undef USER

取消之前定义的宏USER。