

§. 编译预处理 – 条件编译

1. 条件编译

★ 编译预处理之条件编译

《C++程序设计》（第1版）

P. 131-132 4. 16. 3 条件编译

例1:

输入成绩，根据分数 打印及格或不及格	某个游戏软件，适应不同 大小的屏幕
<pre>if (...) { ... } else { ... }</pre>	<pre>if (屏幕是800*480) { ... } else { ... }</pre>

例2:

输入成绩，根据分数 打印分数等级	某个游戏软件，适应不同 大小的屏幕
<pre>if (...) { ... } else if (...) { ... } else { ... }</pre>	<pre>if (屏幕是800*480) { ... } else if (屏幕是1280*720) { ... } else { ... }</pre>

可能else if
会重复多次

可能else if
会重复多次

左右的区别:

输入成绩，根据分数打印
分数等级

对于if-else形式的双
分支(包括if-elseif-
else形式的多分支)，

每次执行时只能选择
其中的某一个分支执行
但多次执行时可能每个
分支都可能会被执行到，

因此每个分支都是有意义
的

某个游戏软件，适应不同
大小的屏幕

对于if-else形式的双
分支(包括if-elseif-
else形式的多分支)，

对于某种型号的设备，
多次执行的都是其中
的一个固定分支，

因此分支中的其他部分
对某种型号设备的可执行
文件而言是无意义的

因为多种型号设备
的公用代码很多，
因此希望维护一套
源程序代码

既希望一套源代码
又希望执行文件中
只包含有效部分

§. 编译预处理 – 条件编译

1. 条件编译

引入：某些程序在调试、兼容性、平台移植等情况下希望通过简单设置参数就能生成不同的软件

方法1：把所有可能用到的代码都写进源程序文件中，再全部编译到可执行文件中，执行时根据相应的条件选择不同的代码执行

(分支语句方式，源程序统一，可执行代码大)

方法2：把所有可能用到的代码都写进源程序文件中，在编译之前根据需要进行选择待编译的代码段，再进行编译，可执行文件中只包含需要的代码段

(条件编译方式，源程序统一，可执行代码小)

§. 编译预处理 – 条件编译

1. 条件编译

<code>#ifdef 标识符</code> 程序段1 <code>#else</code> 程序段2 <code>#endif</code>	<code>#ifndef 标识符</code> 程序段1 <code>#else</code> 程序段2 <code>#endif</code>	<code>#if 表达式</code> 程序段1 <code>#else</code> 程序段2 <code>#endif</code>
若定义了标识符 则编译程序段1 否则编译程序段2	若未定义标识符 则编译程序段1 否则编译程序段2	若表达式为真 则编译程序段1 否则编译程序段2

- ★ 该过程在编译阶段完成，最终形成的可执行文件中只包含两个程序段之一
- ★ 预编译指令中的表达式与C语言本身的表达式基本一致，逻辑运算、算术运算等均可用于预编译指令
- ★ 修改预编译条件后，每次都要**重新编译**链接才能生成新的可执行文件
- ★ 修改条件编译的方法一般是**手动修改，再次编译**
- ★ 更进一步的方法：1、将条件编译开关放在设置或makefile文件中，通过修改makefile来编译
2、利用标识不同编译器的预置宏定义来区分不同编译器
- ★ **不可能**通过执行程序时输入某值或根据程序执行时是否满足某条件的方式来选择，因为是“预编译”

<p>程序如下，假设两次的输入为7，-3，写出程序的运行结果</p> <pre> #define PROGRAM main() { int t; cin >> t; } #ifdef PROGRAM if (t>=0) cout << "t是非负整数" << endl; #else if (t<0) cout << "t是负整数" << endl; #endif cout << "End." << endl; return 0; } </pre> <div>输入为7时： t是非负整数 End</div> <div>输入为-3时： End</div>	<p>程序如下，假设两次的输入为7，-3，写出程序的运行结果</p> <pre> #define PROGRAM main() { int t; cin >> t; } #ifndef PROGRAM if (t>=0) cout << "t是非负整数" << endl; #else if (t<0) cout << "t是负整数" << endl; #endif cout << "End." << endl; return 0; } </pre> <div>未被编译进可执行文件中 cout << "t是非负整数" << endl;</div> <div>输入为7时： End</div> <div>输入为-3时： t是负整数 End</div>
<p>程序如下，假设两次的输入为7，-3，写出程序的运行结果</p> <pre> #define PROGRAM 0 main() { int t; cin >> t; } #if PROGRAM if (t>=0) cout << "t是非负整数" << endl; #else if (t<0) cout << "t是负整数" << endl; #endif cout << "End." << endl; return 0; } </pre> <div>未被编译进可执行文件中 cout << "t是非负整数" << endl;</div> <div>输入为7时： End</div> <div>输入为-3时： t是负整数 End</div>	<p>程序如下，假设两次的输入为7，-3，写出程序的运行结果</p> <pre> #define PROGRAM 1 main() { int t; cin >> t; } #if PROGRAM if (t>=0) cout << "t是非负整数" << endl; #else if (t<0) cout << "t是负整数" << endl; #endif cout << "End." << endl; return 0; } </pre> <div>输入为7时： t是非负整数 End</div> <div>输入为-3时： End</div> <div>未被编译进可执行文件中 cout << "t是负整数" << endl;</div>

§. 编译预处理 – 条件编译

思考：如何让下面程序在多编译器中均能通过

提示：利用标识不同编译器的预置宏定义来区分不同编译器([查找各编译器的预置标识](#))

```
#include <iostream>
using namespace std;

int main()
{
    char a[80];

    #if (***) || (***)                如果是Dev/CB
        gets(a);
    #elif (***)                      如果是VS2017
        gets_s(a);
    #elif (***)                      如果是Linux
        fgets(a, 10, stdin);
    #endif

    cout << a << endl;
    return 0;
}
```