



黑客派

# C 语言实现 try catch 处理

作者: [bfchen](#)

原文链接: <https://hacpai.com/article/1510981407646>

来源网站: [黑客派](#)

许可协议: [署名-相同方式共享 4.0 国际 \(CC BY-SA 4.0\)](#)

<p>我们都知道，在 Java、C# 等高级程序语言中，都自带有异常处理机制，其基本结构如下：<br>  
<pre>try{<br> 程序语句;<br>}catch(Exception ex){<br> 异常处理;<br>}<br></pre>这样做不但可以防止程序异常终止，而且在出现错误时可以及时作一些释放资源处理，对程序能继续健壮的运行下去尤其重要</p>  
<script async src="https://pagead2.googlesyndication.com/pagead/js/adsbygoogle.js"></script>  
<!-- 黑客派PC帖子内嵌-展示 -->  
<ins class="adsbygoogle" style="display:block" data-ad-client="ca-pub-5357405790190342" data-ad-slot="8316640078" data-ad-format="auto" data-full-width-responsive="true"></ins>  
<script>  
 (adsbygoogle = window.adsbygoogle || []).push({});  
</script>  
<p>但是 C 语言中没有这样异常处理机制，我们可以通过宏定义去实现类似这样的操作，这时候就用到 C 语言强大而又被我们‘嫌弃’的 goto 语句了。<br>这里我们需要定义几个宏：<br>(1) 异常处理框架<br><pre>#define HANDLE\_ERROR(msg) { \<br> goto \_error; \<br>\_error: \<br> printf(msg); \<br> return -1; \<br>}</pre>在 BEGIN\_PROC 和 END\_PROC 里处理我的程序，有异常的情况时跳到 CATCH\_ERROR 和 END\_ERROR 里统一处理<br>(2) 几种异常条  
宏定义：<br><pre>#define HANDLE\_ERROR(msg) { \<br> goto \_error; \<br>\_error: \<br> printf(msg); \<br> return -1; \<br>}</pre>  
<p>(3) 异常统一处理宏定义：<br><pre>#define HANDLE\_ERROR(msg) { \<br> goto \_error; \<br>\_error: \<br> printf(msg); \<br> return -1; \<br>}</pre>  
<p>下面我们来分析这三种情况：<br>(1) 不满足某种条件产生异常跳转：<br><pre>int main() { \<br> int arr[10]; \<br> for (int i = 0; i < 10; i++) \<br> arr[i] = i; \<br> for (int i = 0; i < 10; i++) \<br> if (arr[i] > 10) \<br> goto \_error; \<br> return 0; \<br>\_error: \<br> printf("Error: array index out of bounds\n"); \<br> return -1; \<br>}</pre>这里是判断数组是否索引是否越界，如果越界则判断跳转失败，程序如何结构：  
<pre>int main() { \<br> int arr[10]; \<br> for (int i = 0; i < 10; i++) \<br> arr[i] = i; \<br> for (int i = 0; i < 10; i++) \<br> if (arr[i] > 10) \<br> goto \_error; \<br> return 0; \<br>\_error: \<br> printf("Error: array index out of bounds\n"); \<br> return -1; \<br>}</pre>可以看到，这里打印了出错的函数名、行号和错误码  
我们可以很容易的定位出错误<br>(2) 出错的时候抛出错误，统一处理：<br><pre>int main() { \<br> int arr[10]; \<br> for (int i = 0; i < 10; i++) \<br> arr[i] = i; \<br> for (int i = 0; i < 10; i++) \<br> if (arr[i] > 10) \<br> goto \_error; \<br> return 0; \<br>\_error: \<br> printf("Error: array index out of bounds\n"); \<br> return -1; \<br>}</pre>这里我们打开文件或分配内存失败都统一到错误处理当中，而不是直接 return 返回  
这是一种很好的处理方法，不然很容易导致内存泄露或文件未关闭，<br>从上面我们可以看出，当用 return 语句时，文件未打开成功，则返回，而当文件打开成功，而内存申请失败的时候，我们很  
容易直接返回，而导致<br>文件未正常关闭。使用了框架后，我们把使用的资源都作一个统一的管理  
这样就不会轻易造成资源泄露的问题了，同时也有问题定位打印信息<br>(3) 执行函数是否成功而跳  
：<br><pre>int main() { \<br> int arr[10]; \<br> for (int i = 0; i < 10; i++) \<br> arr[i] = i; \<br> for (int i = 0; i < 10; i++) \<br> if (arr[i] > 10) \<br> goto \_error; \<br> return 0; \<br>\_error: \<br> printf("Error: array index out of bounds\n"); \<br> return -1; \<br>}</pre>当函数执行的返回值不成功时，我们也进行到异常  
转，进行统一处理<br>以上就是 C 语言的一些常见的异常处理，当然还有其他的异常处理条件，但  
基于以上框架可以很容易的做到。</p>