# 异常处理

## 课程目标

* 能够使用 try…except… 语法实现对程序异常的捕获和处理
* 能够使用except对不同类型的异常进行捕获处理
* 能够完成用户登录信息校验案例

## 课程内容

* 异常处理
* 自定义异常

### 异常处理

#### 1.1异常概念

异常是程序执行过程中出现的非正常流程现象。

异常是无法避免的，只能先预估出可能出现的异常现象，并提供对应的处理机制，在异常出现后保障程序不被中断执行。

#### 1.2异常处理基本语法

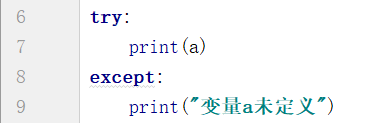
**格式一（常用）：**

try：

\_\_\_\_可能引发异常现象的代码

except：

\_\_\_\_出现异常现象的处理代码



**格式二（不常用）：**

try：

\_\_\_\_可能引发异常现象的代码

finally：

\_\_\_\_try代码块结束后运行的代码

**格式三（完整格式）：**

try：

\_\_\_\_可能引发异常现象的代码

except：

\_\_\_\_出现异常现象的处理代码

else：

\_\_\_\_未出现异常现象的处理代码

finally：

\_\_\_\_try代码块结束后运行的代码

各代码块功能与执行说明：

|  |  |  |
| --- | --- | --- |
| 名称 | 作用 | 说明 |
| try | 包裹可能引发异常的代码 | 被包裹的代码有可能出现异常也有可能不出现异常 |
| except | 对出现的异常进行处理 | 异常处理代码块可以有多个，根据不同种类的异常定义不同的处理方案 |
| else | 未出现异常的执行代码 | 当try代码块中没有出现异常时，try中代码执行完毕执行else中的代码 |
| finally | 异常处理机制结束执行代码 | 无论try代码块中是否出现异常现象，都将执行finally代码块中的内容 |

#### 1.3异常精细化处理

异常处理可以根据不同种类的异常进行不同的处理方案设计，使用异常名称来进行区分。标准格式如下：

except 异常类名1 **as** 变量名：

\_\_\_\_出现异常现象的处理代码

except 异常类名2 **as** 变量名：

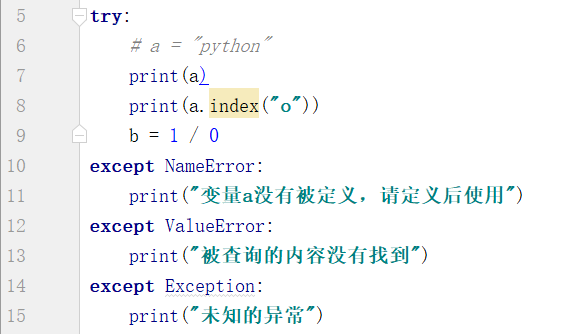
\_\_\_\_出现异常现象的处理代码

except 异常类名3 **as** 变量名：

\_\_\_\_出现异常现象的处理代码

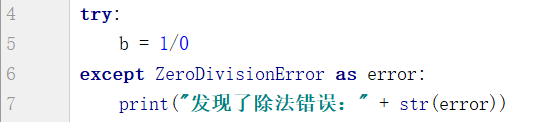
except **Exception as** 变量名：

\_\_\_\_出现异常现象的处理代码



说明：

1.通过异常类名后面的变量名可以获取到具体异常信息



2.执行了某个except中的内容后，其他except中的内容将不再执行

3.Exception类是所有异常类的父类，能够拦截所有的异常现象，通常

设置成最后一级的捕获

#### 1.4异常对象主动抛出

异常现象可以使用主动抛出的形式来构造，语法格式如下：

**raise** 异常类对象

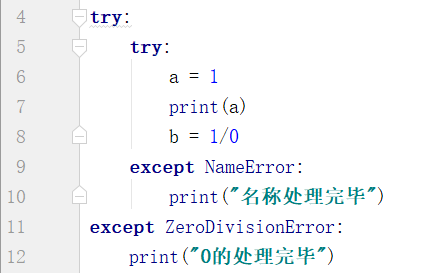


注意：异常对象使用raise抛出时，通常是满足某中情况时抛出，否则将会出现程序执行到此处必定抛出异常。

#### 1.5异常处理传递

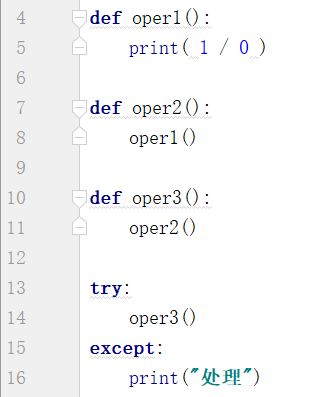
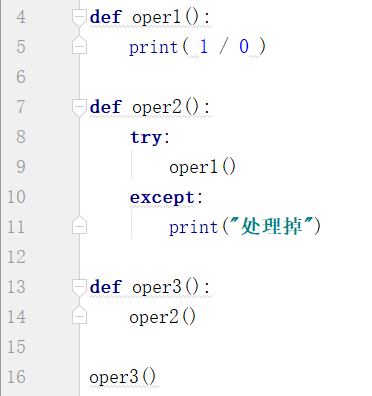
异常出现后如果没有被捕获处理，该异常将会继续向下传递，到下一次 调用位置。

情况一：使用try嵌套继续处理



情况二：函数/方法间调用，在任意调用位置处理

最后调用处理 中间调用处理

### 自定义异常

#### 2.1自定义异常基本语法

class 自定义异常类名(Exception):

pass

#### 2.2自定义异常主动触发

raise 异常类对象

### 案例：用户登录信息校验

"""  
案例：用户登录信息校验  
要求：  
 用户输入用户名，密码后对信息进行校验  
 1.用户名长度在3-8个字符  
 2.用户名中只能出现英文字母和数字  
 3.密码长度必须是6位  
 4.密码必须由纯数字组成  
分析：  
1.信息使用input操作获取  
2.定义自定义异常，描述非法信息  
3.提供检测的函数，针对情况进行处理  
4.执行程序中要使用try结构完成检测  
"""  
  
class NameQuestion(Exception):  
 pass  
class PasswordQuestion(Exception):  
 pass  
  
def check\_login(name,pwd):  
 """检测用户名和密码是否合理"""  
 if len(name) <3 or len(name) > 8:  
 raise NameQuestion("用户名长度必须在3到8个字符之间")  
 if not name.isalnum():  
 raise NameQuestion("用户名中必须使用英文字母和数字组成")  
 if len(pwd) != 6:  
 raise PasswordQuestion("密码长度必须是6位")  
 if not pwd.isnumeric():  
 raise PasswordQuestion("密码必须由纯数字组成")  
  
name = input("请输入用户名:")  
pwd = input("请输入密码:")  
  
try:  
 check\_login(name,pwd)  
except NameQuestion as e:  
 print(str(e))  
except PasswordQuestion as e:  
 print(str(e))  
else:  
 print("正常登陆，请使用")