# 15 异常处理

@(SIGAI课程录制)

**什么是异常?**

- 错误：比如代码语法有问题，程序无法启动；比如试图在除法中除以0
- 小概率事件：比如图像识别API遇到了图像尺寸为2x2的图片

**为什么程序会出现异常?**

- 程序的某一部分不由程序编写者控制（使用别人的数据；数据等待外部输入；程序运行环境一致性问题）
- 程序编写者难以考虑到全部情况并预先提供处理方式

**通常如何处理?**

- 条件语句：`if/else`
- 异常处理：`try/except/else/finally`

**Python中的异常及相关语法**

- `Exception`：Python内置的异常类
- `raise`：抛出异常
- `try`：尝试运行以下语句
- `except`：在`try`语句之后，捕获某个异常，为空则捕获全部异常（很危险，难以debug）
- `else`：在`try`语句之后，如果没有捕获到异常，则执行
- `finally`：在`try`语句之后，无论是否捕获到异常，均执行

**案例：主动抛出异常**

```
>>> raise Exception
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
Exception

>>> raise Exception("Hello, I'm Exception.")
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
Exception: Hello, I'm Exception.
```

**案例：被动遇到异常**

```
>>> 1 / 0
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
ZeroDivisionError: division by zero
```

**案例：异常处理语句 - 未遇到异常**

```
try:
    print("Enter try.")
except:
    print("Enter except.")
else:
    print("Enter else.")
finally:
    print("Enter finally.")
```

输出：

```
Enter try.
Enter else.
Enter finally.
```

**案例：异常处理语句 - 捕获全部异常**

```
try:
    print("Enter try.")
    1 / 0
except:
    print("Enter except.")
else:
    print("Enter else.")
finally:
    print("Enter finally.")
```

输出：

```
Enter try.
Enter except.
Enter finally.
```

**案例：异常处理语句 - 捕获指定异常**

```
try:
    print("Enter try.")
    1 / 0
except ZeroDivisionError:
    print("Enter except ZeroDivisionError.")
except ArithmeticError:
    print("Enter except ArithmeticError.")
except:
    print("Enter except.")
else:
    print("Enter else.")
finally:
    print("Enter finally.")
```

输出：

```
Enter try.
Enter except ZeroDivisionError.
Enter finally.
```

**案例：异常处理语句 - 捕获异常后仍抛出**

```
MODE = "DEBUG"

try:
    print("Enter try.")
    1 / 0
except:
    print("Enter except.")
    if MODE == "DEBUG":
        raise
else:
    print("Enter else.")
finally:
    print("Enter finally.")
```

输出：

```
Enter try.
Enter except.
Enter finally.
Traceback (most recent call last):
  File "Exception-1.py", line 5, in <module>
    1 / 0
ZeroDivisionError: division by zero
```

**案例：异常处理语句 - 捕获异常后显示异常信息但不抛出异常**

```
MODE = "WARN"

try:
    print("Enter try.")
    1 / 0
except Exception as e:
    print("Enter except.")
    if MODE == "DEBUG":
        raise
    elif MODE == "WARN":
        print(e)
else:
    print("Enter else.")
finally:
    print("Enter finally.")
```

输出：

```
Enter try.
Enter except.
division by zero
Enter finally.
```

**案例：异常语句与循环的连用**

```
MODE = "WARN"

def main():
    while True:
        try:
            print("Please input your Python 3 Expression.")
            exp = input()
            print("The result of your expression is", eval(exp))
        except Exception as e:
            if MODE == "DEBUG":
                raise
            elif MODE == "WARN":
                print(e)
            elif MODE == "STABLE":
                print("Something wrong, please input again.")
        else:
            break

if __name__ == "__main__":
    main()
```

输出：

```
Please input your Python 3 Expression.
1 / 0
division by zero
Please input your Python 3 Expression.
1 = 1
invalid syntax (<string>, line 1)
Please input your Python 3 Expression.
2 ** 10
The result of your expression is 1024
```

**异常的传播**

```
def f1():
    try:
        return 1 / 0
    except Exception as e:
        print("This is f1.")
        print(e)

def f2():
    try:
        f1()
    except Exception as e:
        print("This is f2.")
        print(e)

def f3():
    try:
        f2()
    except Exception as e:
        print("This is f3.")
        print(e)

def main():
    try:
        f3()
    except Exception as e:
        print("This is main.")
        print(e)

if __name__ == "__main__":
    try:
        main()
    except Exception as e:
        print("This is __main__.")
        print(e)
```

输出：

```
This is f1.
division by zero
```

---

```python
def f1():
    try:
        return 1 / 0
    except Exception as e:
        print("This is f1.")
        print(e)
        raise

def f2():
    try:
        f1()
    except Exception as e:
        print("This is f2.")
        print(e)

def f3():
    try:
        f2()
    except Exception as e:
        print("This is f3.")
        print(e)

def main():
    try:
        f3()
    except Exception as e:
        print("This is main.")
        print(e)

if __name__ == "__main__":
    try:
        main()
    except Exception as e:
        print("This is __main__.")
        print(e)
```

输出:

```
This is f1.
division by zero
This is f2.
division by zero
```

---

```
def f1():
    try:
        return 1 / 0
    except Exception as e:
        print("This is f1.")
        print(e)
        raise

def f2():
    try:
        f1()
    except Exception as e:
        print("This is f2.")
        print(e)
        raise

def f3():
    try:
        f2()
    except Exception as e:
        print("This is f3.")
        print(e)
        raise

def main():
    try:
        f3()
    except Exception as e:
        print("This is main.")
        print(e)
        raise

if __name__ == "__main__":
    try:
        main()
    except Exception as e:
        print("This is __main__.")
        print(e)
```

输出：

```
This is f1.
division by zero
This is f2.
division by zero
This is f3.
division by zero
This is main.
division by zero
This is __main__.
division by zero
```

---

```python
def f1():
    try:
        return 1 / 0
    except Exception as e:
        print("This is f1.")
        print(e)
        raise

def f2():
    try:
        f1()
    except Exception as e:
        print("This is f2.")
        print(e)
        raise

def f3():
    try:
        f2()
    except Exception as e:
        print("This is f3.")
        print(e)
        raise

def main():
    try:
        f3()
    except Exception as e:
        print("This is main.")
        print(e)
        raise

if __name__ == "__main__":
    try:
        main()
    except Exception as e:
```

```
        print("This is __main__.")
        print(e)
        raise
```

输出：

```
This is f1.
division by zero
This is f2.
division by zero
This is f3.
division by zero
This is main.
division by zero
This is __main__.
division by zero
Traceback (most recent call last):
  File "Exception-2.py", line 35, in <module>
    main()
  File "Exception-2.py", line 27, in main
    f3()
  File "Exception-2.py", line 19, in f3
    f2()
  File "Exception-2.py", line 11, in f2
    f1()
  File "Exception-2.py", line 3, in f1
    return 1 / 0
ZeroDivisionError: division by zero
```

---

**关于异常的后续学习**

1. 内置异常有很多类型，感兴趣可以参考链接：快速浏览点我 官方文档点我
2. 异常是可以自定义的，但需要先了解内置的异常类，然后选择合适的内置异常类进行继承即可