课程：异常

目标

- 了解异常
- 捕获异常
- 异常的else
- 异常finally
- 异常的传递
- 自定义异常

一. 了解异常

当检测到一个错误时，解释器就无法继续执行了，反而出现了一些错误的提示，这就是所谓的"异常"。

例如：以`r`方式打开一个不存在的文件。

```
1 open('test.txt', 'r')
```

```
C:\Users\黑马程序员\AppData\Local\Programs\Python\Python37\python.exe
Traceback (most recent call last):
  File "C:/Users/黑马程序员/Desktop/code/异常.py", line 1, in <module>
    open('test.txt', 'r')
FileNotFoundError: [Errno 2] No such file or directory: 'test.txt'

Process finished with exit code 1
```

二. 异常的写法

2.1 语法

```
1 try:
2     可能发生错误的代码
3 except:
4     如果出现异常执行的代码
```

2.2 快速体验

需求：尝试以`r`模式打开文件，如果文件不存在，则以`w`方式打开。

```
try:
    f = open('test.txt', 'r')
except:
    f = open('test.txt', 'w')
```

2.3 捕获指定异常

2.3.1 语法

```
try:
    可能发生错误的代码
except 异常类型:
    如果捕获到该异常类型执行的代码
```

2.3.2 体验

```
try:
    print(num)
except NameError:
    print('有错误')
```

> 注意：
>
> 1. 如果尝试执行的代码的异常类型和要捕获的异常类型不一致，则无法捕获异常。
> 2. 一般try下方只放一行尝试执行的代码。

2.3.3 捕获多个指定异常

当捕获多个异常时，可以把要捕获的异常类型的名字，放到except 后，并使用元组的方式进行书写。

```
try:
    print(1/0)

except (NameError, ZeroDivisionError):
    print('有错误')
```

2.3.4 捕获异常描述信息

```
try:
    print(num)
except (NameError, ZeroDivisionError) as result:
    print(result)
```

2.3.5 捕获所有异常

Exception是所有程序异常类的父类。

```
try:
    print(num)
except Exception as result:
    print(result)
```

2.4 异常的else

else表示的是如果没有异常要执行的代码。

```
try:
    print(1)
except Exception as result:
    print(result)
else:
    print('我是else，是没有异常的时候执行的代码')
```

2.5 异常的finally

finally表示的是无论是否异常都要执行的代码，例如关闭文件。

```
try:
    f = open('test.txt', 'r')
except Exception as result:
    f = open('test.txt', 'w')
else:
    print('没有异常，真开心')
finally:
    f.close()
```

三. 异常的传递

体验异常传递

需求：

1. 尝试只读方式打开test.txt文件，如果文件存在则读取文件内容，文件不存在则提示用户即可。

2. 读取内容要求：尝试循环读取内容，读取过程中如果检测到用户意外终止程序，则`except`捕获异常并提示用户。

```
import time
try:
    f = open('test.txt')
```

```
    try:
        while True:
            content = f.readline()
            if len(content) == 0:
                break
            time.sleep(2)
            print(content)
    except:
        # 如果在读取文件的过程中，产生了异常，那么就会捕获到
        # 比如 按下了 ctrl+c
        print('意外终止了读取数据')
    finally:
        f.close()
        print('关闭文件')
except:
    print("没有这个文件")
```

四. 自定义异常

在Python中，抛出自定义异常的语法为`raise 异常类对象`。

需求：密码长度不足，则报异常（用户输入密码，如果输入的长度不足3位，则报错，即抛出自定义异常，并捕获该异常）。

```
# 自定义异常类，继承Exception
class ShortInputError(Exception):
    def __init__(self, length, min_len):
        self.length = length
        self.min_len = min_len

    # 设置抛出异常的描述信息
    def __str__(self):
        return f'你输入的长度是{self.length}，不能少于{self.min_len}个字符'


def main():
    try:
        con = input('请输入密码：')
        if len(con) < 3:
            raise ShortInputError(len(con), 3)
    except Exception as result:
        print(result)
    else:
        print('密码已经输入完成')


main()
```

五. 总结

- 异常语法

```
try:
    可能发生异常的代码
except:
    如果出现异常执行的代码
else:
    没有异常执行的代码
finally:
    无论是否异常都要执行的代码
```

- 捕获异常

```
except 异常类型:
    代码

except 异常类型 as xx:
    代码
```

- 自定义异常

```
# 1. 自定义异常类
class 异常类类名(Exception):
    代码

    # 设置抛出异常的描述信息
    def __str__(self):
      return ...


# 2. 抛出异常
raise 异常类名()

# 捕获异常
except Exception...
```