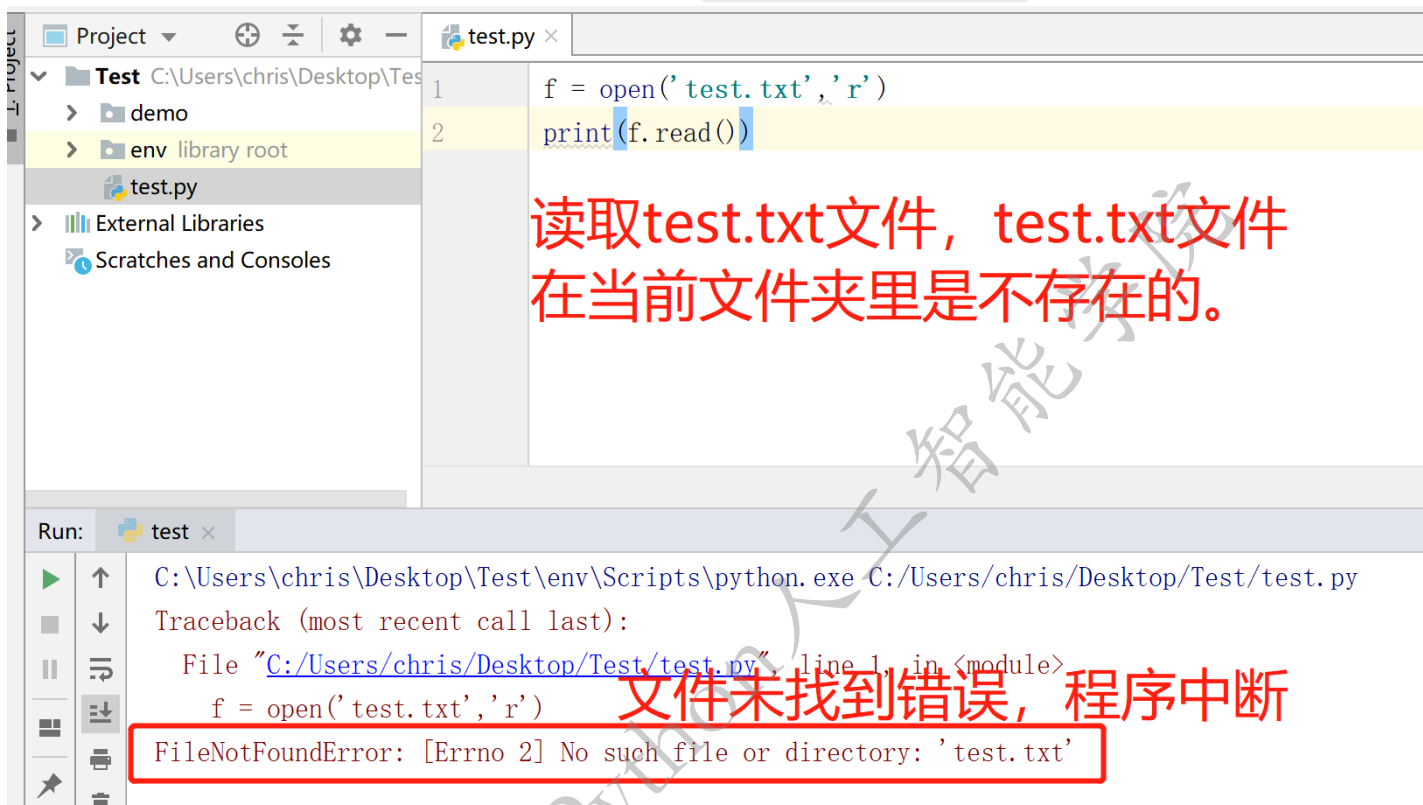


# 异常的概念

程序在运行过程中，由于我们的编码不规范，或者其他原因一些客观原因，导致我们的程序无法继续运行，此时，程序就会出现异常。如果我们不对异常进行处理，程序可能会由于异常直接中断掉。为了保证程序的健壮性，我们在程序设计里提出了异常处理这个概念。

## 读取文件异常

在读取一个文件时，如果这个文件不存在，则会报出 `FileNotFoundError` 错误。



程序在运行过程中会经常遇到类似的异常，如果我们不进行处理，此时程序就会中断并退出。为了提高程序的健壮性，我们可以使用异常处理机制来解决程序运行过程中可能出现的问题。

## try...except语句

try...except语句可以对代码运行过程中可能出现的异常进行处理。语法结构:


```
try:
    可能会出现异常的代码块
except 异常的类型:
    出现异常以后的处理语句
```

示例:

```
try:
    f = open('test.txt', 'r')
    print(f.read())
except FileNotFoundError:
    print('文件没有找到, 请检查文件名称是否正确')
```

## try...else语句

咱们应该对else并不陌生，在if中，它的作用是当条件不满足时执行的实行；同样在try...except...中也是如此，即如果没有捕获到异常，那么就执行else中的事情

try: num = 100 print(num) except NameError as errorMsg: print('产生错误了:%s'%errorMsg) else: print('没有捕获到异常，真高兴') 运行结果如下: 

## try..finally语句

try...finally...语句用来表达这样的情况:

在程序中，如果一个段代码必须要执行，即无论异常是否产生都要执行，那么此时就需要使用finally。比如文件关闭，释放锁，把数据库连接返还给连接池等。

```
try:
    f = open('test.txt')
    try:
        while True:
            content = f.readline()
            if len(content) == 0:
                break
            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
        pass
    finally:
        f.close()
        print('关闭文件')
except:
    print("没有这个文件")
```

说明:

我们可以观察到KeyboardInterrupt异常被触发，程序退出。但是在程序退出之前，finally从句仍然被执行，把文件关闭。