一.路径处理

OS模块
熟记：os.path.dirname和os.path.join

1.os.path.dirname :获取文件/文件夹所在的目录路径

```
例子1：
import os
file_path =
‘/Users/zhangcaiyan/Desktop/Lemon_python/lemon_functional/lemon_05.py’
res = os.path.dirname(file_path)
print(res)
```

2.os.path.join:路径拼接的方法, 自动根据系统添加斜杠

```
例子2：
path01 = ‘/Users/zhangcaiyan/Desktop/Lemon_python/lemon_functional’
file_name = 'lemon_06.py'
file_path = os.path.join(path01,file_name) #等同于  “\”.join(path01,file_name)
print(file_path)
```

3.os.path.abspath:获取绝对路径

```
res = os.path.abspath(".")
print(res)
```

. 当前路径
.. 上一级
file: 在哪个文件里面，pycharm运行打印函数就是文件的名字

```
res1 = os.path.abspath(__file__)    #获取当前文件的绝对路径
print(res1)
```

4.os.getcwd:获取当前的工作路径（liunx pwd)

```
res2 = os.getcwd()
print(res2)
```

5.os.chdir:切换工作路径 (等同于liunx下的cd)

```
res3 = os.chdir("..")
print(res3)
```

6.os.listdir:获取当前工作路径下的文件（liunx的ls)

```
res4 = os.listdir('.')
print(res4)
```

7.os.mkdir:创建一个文件夹

os.mkdir('cathy_test')

8.os.rmdir:删除一个文件夹

os.rmdir('cathy_test)

9.判断是否是文件

```
res =
os.path.isfile('/Users/zhangcaiyan/Desktop/Lemon_python/lemon_functional')
print(res)
文件夹   打印输出 false
文件     打印输出 True
```

10.判断是否是文件夹

```
res2 =
os.path.isdir('/Users/zhangcaiyan/Desktop/Lemon_python/lemon_functional')
print(res2)
```

二. 异常

1.定义：

当程序运行中检测到一个错误时，解释器无法继续执行，出现一些错误提示

2.常见的异常：

1.变量没定义
2.语法错误
3.键不存在
4.没找到模块
5.类型错误

3.异常四大类

--systemExit　系统退出
--systemKeyboard
--BaseException 所有异常的基类
--Exception 常见错误的基类(重点掌握),可以捕获所有的异常类型，除了语法错误

4.异常的捕获

目的：可以提高代码的容错性
1.异常捕获 try---except:
try:里面的代码有以下几种:
1.涉及到用户的输入
2.涉及到文件操作
3.涉及到网络请求
except:捕获指定的异常类型
#所有的异常做相同处理
except Exception as e:
    print("代码的错误信息为：“,e)

else:
    #try里面的代码没有出现异常就会执行else
finally
    #不管try里面的代码是否出现异常，都会执行

例子1

```
try:
    #有可能出错的代码，放到try下面
    num = int(input(“输入一个数字：”))
    with open('test.txt','r',encoding = 'utf-8') as f:
        f.read()
except ValueError as e:
    #当try里面的代码出错了，就会执行except里面的代码，（可以在此处理异常）
    print("你输入的不是数字“)
except FileNotFoundError:
    print("文件不存在！“)
#所有的异常做相同处理
except Exception as e:
    print("代码的错误信息为：“,e)
else:
    #try里面的代码没有出现异常就会执行else
finally:
    #不管try里面的代码是否出现异常，都会执行
```