# day01

## 复习：

RN项目最后问题：商品详情图片等比例缩放问题

## 1.掌握一门新编程语言的步骤：

(1)了解背景：

历史：1989年Guido开发，1991年V1发布，2000年V2，2008年V3发布，V3不兼容V2

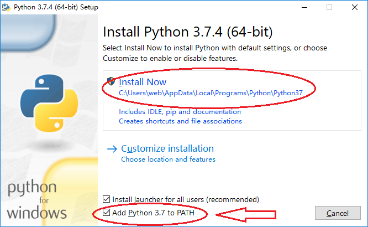
现状：Tiobe排行榜第三

特点：简单易上手，跨平台，开源，应用广泛

应用领域：运维、Web、桌面应用程序开发、游戏、爬虫、AI人工智能

(2)搭建环境：编写HelloWorld

下载并安装Python解释器



运行方式1：交互式执行

运行方式2：脚本式执行

(3)变量和常量

(4)数据类型

(5)运算符

(6)逻辑结构

(7)通用小程序

(8)函数和对象

(9)官方函数和对象

(10)第三方库和框架

|  |
| --- |
| 提示：Python的交互式执行环境下，提供了非常好的帮助手册，使用方法： help( 主题 )  例如： help( print )、 help( "keywords" ) |
| 后台编程语言老三样： Java、PHP、.NET  后台编程语言新三样： Node.js、Go、Python |

## 2.Python语法规则

①单行注释使用：# 多行注释使用： '''....'''或者"""..."""

②语句后面可以写分号，也可以省略

③Python严格区分大小写

④Python脚本中行首的TAB/空格产生的缩进是有特殊含义的！不能随便加！！

⑤Python中单引号和双引号的功能一样

## 3.定义和使用变量

提示：Python是一门弱类型语言：变量有类型，但声明时无需指定；一个变量可以先后赋值为不同类型的值

**创建变量：**

变量名 = 值

变量名1=变量名2=变量名3=值

变量名1, 变量名2 = 值1, 值2

**输出变量的类型：**

print( type(变量名) )

**输出变量的值：**

print( 变量名 )

**输出变量的标识符/内存地址：**

print( id(变量名) )

|  |
| --- |
| 变量名中可以包含数字、字母、下划线、中文；  但不能以数字开头；  且不能是关键字( 使用help("keywords")查看 ) |
| Python提供的强制类型转换函数：  int( ... )  float( ... )  bool( ... )  str( ... ) |
| print()函数输出变量的值时，如果有字符串和数字的混合输出，不能直接使用+进行拼接，可以使用如下三种方案：  ①print( '年龄为：' + str(age) )  ②print( '年龄为：', age )  ③print( '数据为：%d%.2f%s'%(age, price, name) ) |
| JS中定义常量： const PI = 3.14  Python中没有const关键字，没有创建常量的机制 |

## 4.Python中的数据类型——与JS等语言非常不同！

数据类型是一门语言的灵魂

说明手册地址：

https://docs.python.org/zh-cn/3/library/index.html

### (1)数字类型

int：整数，具有无限精度！

bool：布尔，只有True=1和False=0

float：浮点数，float( )

complex：复数

### (2)字符串类型

str: 字符串类型，不能写作string

说明：①字符串可以使用单引号、双引号、**三重引号(优点：可以换行)**括起来 ②可以用[下标]形式访问字符串中每个字符，如：[0], [-1], [2:5], [-3:] ③可以使用for..in循环遍历字符串中的每个字符 ④字符串常用API

### (3)序列类型

①list：列表，即内容可变的数组 [10,20,30]

②tuple：元组，内容不可变的数组 (10,20,30) (10,)

③range：范围，创建一个指定范围内(开始/结尾/步长)数字元素的序列，通常用于与for...in组合使用，如：

0/1/2/3.../99

50/55/60/65/..../80

10/8/6/4/2/0

|  |
| --- |
| 访问第i个元素： myList[ i ] #i不能越界  添加元素：myList.append(新值) myList.insert(下标, 新值)  删除元素：del myList[下标] myList.pop(下标)  获取长度：len( myList )  列表拼接：myList1 + myList2 myList1 \* 3  遍历数组：for tmp in myList: |

### (4)集合类型(官方提供了两个)

①set：集合，是一个无序的且不重复的数组，set( )

### (5)映射类型

①dict：字典，保存多个键值对的数组

## 5.常用的运算符

(1)算术运算： + - \* / %(取余) //(取商) \*\*(幂)

说明：//表示取商/商上取整；\*\*表示求N次方；Python中没有自增(++)和自减(--)运算

(2)比较运算：> >= < <= == !=

(3)逻辑运算：and or not

(4)位运算：

(5)赋值运算： = += -= \*= /= %=

(6)其它运算：is(是) is not(不是)

## 课后任务：

(1)整理Python中的基础语法：变量、数据类型、运算符(算术)

(2)整理字符串类型、序列类型、集合类型、映射类型常用API

(3)完成“学生成绩录入系统”：不停的读取用户的键盘输入，“请输入新的学生成绩(输入end结束输出)：”，用户输入end后输出所有的学生成绩；给每个成绩+5分，再次输出所有的学生成绩

# day02

## 复习：

掌握一门新编程的步骤：

(1)了解背景

历史：89年 1991年：V1 2000年：V2 2008年：V3

特点：简单、强大、开源、跨平台

应用领域：运维、桌面、爬虫、Web、科学、AI

(2)搭建环境

安装Python解释器，注意Add Python to PATH

交互模式、脚本模式

(3)变量常量

Python中没有常量定义法

声明变量： age = 20

输出变量： print( '年龄为：%d %s'%(age,uname) )

查看类型： type(age) <class 'int'>

查看内存： id(age)

(4)数据类型

(5)运算符

(6)逻辑结构

(7)通用小程序

(8)函数对象

(9)官方函数对象

(10)第三方函数对象

## 1.Python中的数据类型

### (1)数字类型：

int：整数，无限整数， int( )

bool：布尔，True-1/False-0

float：浮点数， float( )

complex：复数

### (2)字符串类型：

str：字符串，单引号/双引号/三引号，使用[s:e]访问字符

### (3)序列类型：

list：列表，内容可变的数组 [1, 2, 3] list()

tuple：元组，内部不可变的数组 (1, 2, 3) tuple()

range：范围，指定开始/结尾/步长的整数序列 rang()

### (4)集合类型：

set：集合，无序的没有重复元素的数组 set()

创建集合：s1 = set( 序列对象 ) s2 = { 值1, 值2,... }

添加元素：s1.add(值)

删除元素：s1.remove(值)

注意：集合中的每个元素没有下标！不能访问第i个；不存在修改第i个元素的做法！

### (5)映射类型：

dict：字典，保存多个键值对的数组，类似于JS中的对象直接量，但是下标必须用引号括起来

创建字典： d1 = { 'key':value, 'key': value, .... }

添加元素： d1['newKey'] = value

删除元素： del d1['key']

练习：创建一个商品对象（字典），让用户从键盘输入商品名称、单价、购买数量，最后输出：

“商品名称：xxx 单价：xxx 购买数量： xxx 小计：xxx”

|  |
| --- |
| Python数据类型分类（简化版）：  ①不可变类型：int / float / bool / str / tuple / range  ②可变类型：list / dict / set |

## 2.Python中的运算符

**(1)算术运算：** + - \* / %(取余) //(取商) \*\*(幂)

注意：Python中没有 ++ 和 -- 运算

**(2)比较运算：**> >= < <= == !=

注意：Python中没有=== 和!==，可以使用id()相等与否代替，或者使用is和is not运算符代替

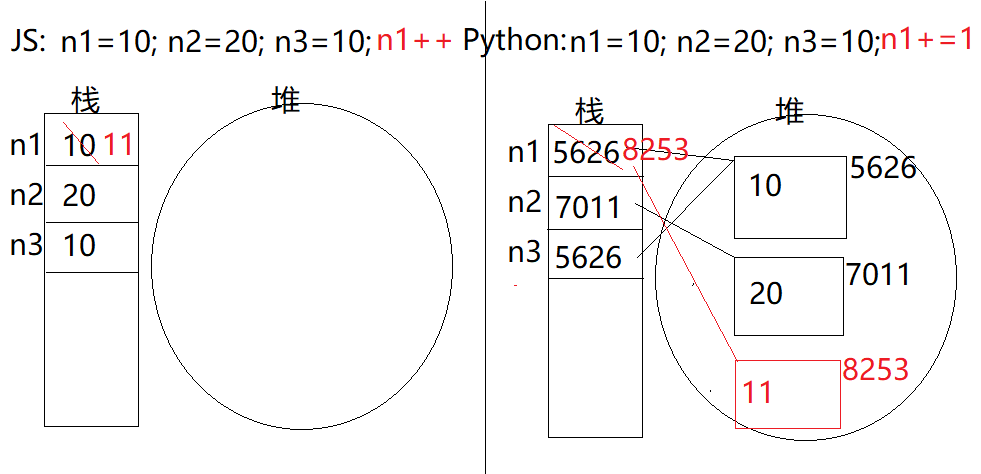
**(3)逻辑运算：** and、 or、 not

注意：Python中不能写作 &&、||、!

**(4)位运算**： &(按位与)、|(按位或)、^(按位异或)、~(按位取反)、<<(按位左移)、>>(按位右移)

**(5)赋值运算**：=、+=、-=、\*=、/=、%=、//=、\*\*=

**(6)其它运算**： is、 is not ——身份运算符 in、not in——成员运算符



练习：从键盘读取一个年份数字输入，判断该年份是否是闰年

练习：输出本世纪(2000~2100)所有的闰年

|  |
| --- |
| 数字a=5，想a变大16倍，如下哪个方法的计算速度最快？  A) a \* 16  B) a = a\*16  C) a = a << 16  D) a = a << 4 |

## 3.Python中的逻辑结构

程序 = 数据 + 算法

算法，有三种执行逻辑：

①顺序逻辑 输入>计算>输出

②选择逻辑

③循环逻辑

JS中的选择逻辑有：if...else... switch...case...

Python中的选择逻辑只有： if...else....

|  |
| --- |
| if 判断条件 :  执行体  IF外面的语句 |
| if 判断条件 :  执行体  else :  执行体  IF外面的语句 |
| if 判断条件 :  执行体  elif 判断条件 :  执行体  else :  执行体  IF外面的语句 |

练习：让用户从命令行输入登录用户名，再输入登录密码，如果用户名为admin密码为123456，则输出“登录成功”否则输出“登录失败”

练习：数据库中使用一个数字表示订单的状态，如10-待付款、 20-发货中、 30-已完成、 其它-不可识别的订单状态；创建一个数字表示订单状态，根据其值输出对应的描述文字——使用两种方法实现

JS中的循环逻辑有：for(){ } while(){ } do{}while() for(..in..) for(..of...)

Python中的循环逻辑只有： for..in.. while...(不知道循环次数/死循环)

|  |
| --- |
| for tmp in range/list/tuple/set/dict :  循环主体  FOR循环外面的语句 |
| while 循环条件 :  循环主体  WHILE循环外面的语句 |
| 注意：for..in可以遍历range/list/tuple/set/dict，但是while只能遍历list/tuple！ |
| 说明：for和while循环中，可以使用continue跳过此次循环进入下一次循环目的；可以使用break打断当前循环退出到外面的目的 |

练习：创建一个元组，其中包含若干正数、负数等数字，使用循环计算所有这些数字的和，并输出

练习：创建一个元组，其中包含若干正数、负数等数字，使用循环计算所有这些数字的和(负数不要)——continue，并输出

练习：创建一个元组，其中包含若干正数、负数等数字，使用循环计算所有这些数字的和，一旦和超过100则立即停止不再添加——break，并输出此时的和

## 4.Python中的函数

定义：一段可以反复多次使用的代码段

定义函数：

def 函数名( 形参列表 ) :

函数主体

return 返回值

调用函数

变量名 = 函数名( 实参列表 )

## 课后练习——通用小程序

(1)创建函数 print99( )，在命令行中输出九九乘法表

(2)创建函数 getLeapYears(start, end)，将指定范围内的闰年保存在一个list中，并返回

(3)创建函数 getPrime(start, end)，将start~end之间的所有质数保存到一个list中，并返回

# day03

## 复习：

见思维导图

## 1.补充：Python的函数

**定义函数：**

|  |
| --- |
| def add( num1, num2 ):  sum = num1 + num2  return sum |

**调用函数：**

|  |
| --- |
| result = add(10, 20)  print( result ) |

**带默认值的参数：**

|  |
| --- |
| #默认情况下，Python函数的实参列表必须与形参列完全一致！  #可以为形参列表尾部的参数指定默认值  def add( num1, num2=20, num3=30):  sum = num1 + num2 + num3  return sum  #调用时可以不给有默认值的形参赋值  result = add(100, 200, 300)  result = add(100, 200)  result = add(100) |

**关键字参数：**在实参赋值时为每个实参指定一个关键字（是某个形参名），这样就可以忽略该参数在形参列表中的顺序问题

|  |
| --- |
| def connect(host, user, pwd):  pass |
| connect(pwd='123456', user='root', host='127.0.0.1')  此处的实参“关键字”必须是某个形参名 |

print(123, end='\n', sep='')

print(123, sep='', end='\n')

利用关键字参数特性，上述两种写法是一样的

**可变长度的参数：**Python默认不允许多个函数名一样；否则后面的后覆盖前面的；但是有时需要给一个函数传递不同数量的实参

|  |
| --- |
| def add( \*args ) : #\*表示此参数是可变长度参数——tuple  pass  #调用函数  add() #args: ()  add(10) #args: (10,)  add(10,20,30) #args: (10, 20,30) |
| def add( \*\*args ) : #\*\*表示此参数是可变长度参数——dict  pass  #调用函数  add() #args: { }  add(n1=10) #args: {'n1':10}  add(n1=10, n2=20, n3=30) #args:{'n1':10, 'n2':20, 'n3':30} |

补充话题：变量/函数作用域

Python中提供了四种作用域：—— **L>E>G>B**

**①L**：Local，本地/局部作用域，只能在当前函数内使用

**②E**：Enclosing，闭包作用域，由外部函数提供给内部函数使用的变量作用域

**③G**：Global，全局作用域，可以在全局使用；

**④B**：Built-In，内置作用域，是由Python提供的函数和变量，如id()/type()/int().....

|  |
| --- |
| 如果函数内想**修改**全局作用域变量值，必须声明：  global 变量名  如果函数内想**修改**闭包作用域变量值，必须声明：  nonlocal 变量名 |

## 2.Python中的面向对象编程

面试题：OOP的三大基本特征：

**①封装：**用对象把数据和数据的操作整合为一个整体

object = data + method

class 类名 :

def \_\_init\_\_(self, 形参列表):

构造方法体

def 成员方法(self): #所有方法的第一个参数都是self

成员方法体

----------------------------------------

#类型实例化/创建类型的实例

变量名 = 类名( 实参列表 )

**②继承：**使子对象可以自动获得父对象的全部成员

Python支持“多继承”——一个class可以有多个直接父class

class Parent :

pass

-----------------------------------------

class Child( Parent1 ):

def \_\_init\_\_(self, 形参列表):

super().\_\_init\_\_(实参)

self.xx = xx

**③多态：**同一个方法，在不同情况下运行结果不同

说明：Python中只有一种：子类覆盖了/重写了父类的同名方法

class Parent:

def work():

pass...

class Child( Parent ) :

def work():

pass...

p1 = Parent() p1.work()

c1 = Child() c1.work()

|  |
| --- |
| 说明：  Python的class中没有静态成员(没有static关键字)，有类似的概念：**类属性：**  class 类名：  类属性 = 值 #直接声明在类中(不在构造方法中)的属性  def \_\_init\_\_(self):  self.成员属性 = 值  访问成员属性只能使用对象的引用，如e1，self，不能用类名；  访问类属性可以使用任意对象的引用，如e1、e2、self，**推荐用类名访问**——类属性在内存中只有一份，不专属于任意个实例，功能类似于其它语言中的static属性 |
| 说明：  **私有属性：**Python中没有private关键字，但也可以声明类的私有属性（不能在类外面被使用的属性）：  class 类名：  def \_\_init\_\_(self):  self.\_\_成员属性 = 值 #以两个\_开头的属性就是私有属性 |

练习：画图板程序中需要使用一个类型：Shape(形状)，其中有属性background（表示背景颜色）、foreground（前景颜色）、borderWidth（边框宽度）；并提供一个打印对象信息的方法：printInfo()试着定义该类，并创建该类的两个实例

练习：画图板程序中还需要一个类型：Rectangle(矩形)，其中除了有Shape所具有的全部特征外，还有width和height属性；定义出该类，并创建该类的两个实例

|  |
| --- |
| 面试题：Node.js/Express需要处理哪些异常？在哪里处理？ |

## 3.Python中的异常处理

异常：Exception，指程序中可能发生也可能不发生的错误情形！如果妥善处理，程序可能继续执行；但是如果不处理，程序将终止执行！！

回忆：JS中的异常处理： try.....catch....catch....finally

Python中的异常处理： try....except....except....finally

基础语法：

|  |
| --- |
| try:  //可能发生异常的代码  except ValueError as err :  print(err)  except TypeError as err :  print(err)  except Exception as err :  print(err)  finally：  print('over') #有无异常都要执行的代码 |
| except Exception as err: 这个捕捉块必须是最后一个，所有的错误都继承自Exception |

程序中，如果为了提醒函数的调用者，此处出现了不应该出现的异常情形，可以抛出自定义的异常，强制调用者加以处理：raise Exception('错误消息')

**def setAge(age)**

**if 情况下:**

**raise Exception('错误消息')**

**#此处的代码如果发生了异常则无法执行**

try:

setAge(888) #函数的调用者必须手工处理异常

except Exception as err:

print(err)

## 课后任务：

(1)整理Python函数和面向对象相关知识点

(2)自学Python装饰器相关知识：

https://www.cnblogs.com/songyue/p/5196809.html

(3)完成阶段项目“净美仕净化器”的数据库表结构设计

http://www.codeboy.com/mfresh

# day04

## 复习：

见思维导图

## 1.模块系统——模块和包

|  |
| --- |
| 回忆：  **Node.js/Common.js：**  module.exports require()  **ES6：**  ①export default import Xx from 'xx'  ②export import {mm,nn} from 'xx |

全局变量/函数会产生全局污染；解决方案——模块，每个文件都是一个模块对象，外界可以导入当前模块对象，使用其中的成员

提示：Python中模块不需要执行“导出”，其内部成员就可以供其它模块“导入”——带来的小问题：不能限制其它模块有选择性的导入

Python中模块导入有两种格式：

**①import user #./user.py**

user.login( )

print(user.uname)

把模块对象当做一个整体一次性导入

**②from user import uname, login #./user.py**

login( )

print( uname )

从指定文件模块中导入内部成员

注意：此写法有个弊端：容易导入多个不同模块中的同名成员！后导入的会覆盖之前的导入成员。解决方法：用as关键字给成员取别名，如：from user import uname as un, login

|  |
| --- |
| Node.js中：模块(module)和包(package)指的是同一种事物，具体又分为“文件模块”和“目录模块” |
| Python中：模块(module）专指文件，包(package)专指目录；包用于盛放多个功能相关的模块——就是一个目录下可以多个.py文件 |

导入包下的模块两种方法：

**①import controller.user** #此处只能导入模块名，不能是包名

controller.user.login()

**②from controller import user** #from+包名，import+模块名

user.login()

**③from controller.user import uname, login**  #from+模块名，import+成员

login()

关于模块和包需要注意：

(1)一个模块一旦被import，会自动缓存一份；如果接下来再次被import会首先使用缓存的模块对象——即多次导入同一模块只会执行一次

(2)包目录下可以声明**\_\_init\_\_.py**文件——这是包的初始化文件，其中可以声明变量\_\_all\_\_=['user', 'product']，其中声明的模块名可以被外界使用 import \* 导入；但是即使\_\_all\_\_中没声明的模块名，也可以使用import 模块名 方式强制导入

(3)每个模块中都有一个变量：\_\_name\_\_，若当前模块是启动模块，则解释器自动赋值为'\_\_main\_\_'，若当前模块不是启动模块(是被其它模块导入的)，此变量的值为'包名.模块名'。利用这一特点，可以进行模块的单元测试，而不影响其被正常导入：

if \_\_name\_\_ == '\_\_main\_\_' :

#测试代码....

练习：创建包shape，其中有两个模块rectangle和circle，其中矩形模块有两个变量width和height，以及两个方法getSize()和getPerimeter()；圆形模块有一个变量r，以及两个方法getSize()和getPerimeter()，分别对这两个模块进行单元测试，保证每个方法运行正常。 创建一个启动模块：main6，导入矩形和原型模块，并调用其中的方法得到面积、得到周长

## 2.官方提供的扩展模块

说明手册：https://docs.python.org/zh-cn/3/library/index.html

安装位置：C:\Users\web\AppData\Local\Programs\Python\Python36-32\Lib

**官方提供的常用模块：**

**①math模块**

ceil()/floor()/sin()/pi....

**②random模块**

randrange()/choice()/shuffle()

**③time模块 数字<=>本地时间对象<=>字符串**

time() #返回当前系统时间代表的整数字(单位为秒)

localtime() #把数字转换为“本地时间”对象

mktime() #把“本地时间”对象转换为数字

strftime() #把“本地时间”对象格式化为字符串

strptime() #把字符串解析为“本地时间”对象

|  |
| --- |
| 总结：把当前系统时间转换为年月日字符串  str = t.strftime( '%Y-%m-%d', t.localtime( t.time() ) ) |
| 把年月日字符串转换为时间数字  sec = t.mktime( t.strptime( '1990-10-1', '%Y-%m-%d' ) ) |

**④json模块 Python对象 <=> JSON字符串**

str = json.dumps(obj) #把Object=>字符串，等同于JSON.stringify()

obj = json.loads( str ) #把字符串=>Object，等同于JSON.parse()

**⑤os.path模块**

p = os.path.abspath( './' ) #把一个相对路径转换为绝对路径

## 3.第三方提供的模块 —— MySQL连接模块 (了解)

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  | | --- | --- | --- | | **解释器** | **核心模块** | **第三方模块管理工具** | | Node.js | http、querystring、... | NPM | | Python | math、random、json... | Pip | |

Pip：是随同Python一同安装的第三方模块包管理工具，功能类似于NPM；默认仓库地址：python.org、pythonhosted.com。PIP常用命令：

① pip install 模块名

pip install -i 非官方模块仓库 模块名

② pip uninstall 模块名

③ pip list #列出已经安装的所有模块/包

④ pip show 模块名 #显示指定模块的详情

演示：使用pip下载安装并使用MySQL数据库连接模块

说明文档：https://dev.mysql.com/doc/connector-python/en/

使用Python连接MySQL数据库的步骤：

(1)使用pip下载MySQL连接器模块

**pip install mysql-connector-python**

或者尝试豆瓣网的仓库：

**pip install -i http://pypi.douban.com/simple mysql-connector-python**

或者尝试使用清华的仓库：

**pip install -i http://e.pypi.python.org mysql-connector-python**

提示：可以使用pip show mysql-connector-python查看具体的安装地址（Python解释器的安装路径下，不是当前项目中）

(2)启动XAMPP中的MySQL数据库服务器

(3)编写Python代码导入MySQL连接器，连接数据库，提交SQL语句，关闭连接

import mysql.connector #导入第三方模块

conn = mysql.connector.connect(host='127.0.0.1', port=3306, user='root', password='', database='xz') #连接到MySQL服务器

#print(conn) #向MySQL服务器发送SQL语句

conn.close() #关闭到数据库的连接

## 课后任务：

(1)分析“净美仕项目”功能点，编写必需的数据库结构

http://www.codeboy.com/mfresh/

(2)学习第三方模块mysql-connector-python执行“增删改查”语句的方法

# pip——命令记录

## 1.目录结构：

C:\Users\web\AppData\Local\Programs\Python\Python36-32\Lib\site-packages

## 2.查看pip版本号命令与结果：

C:\Users\web>pip list

Package Version

---------------------- -------

Django 2.0.5

mysql-connector-python 8.0.17

pip 19.2.2

protobuf 3.9.1

PyMySQL 0.8.1

pytz 2018.4

setuptools 39.0.1

six 1.12.0

## 3.如果版本号低的更新命令与结果：

C:\Users\web>python -m pip install --upgrade pip

Collecting pip

Downloading https://files.pythonhosted.org/packages/8d/07/f7d7ced2f97ca3098c16565efbe6b15fafcba53e8d9bdb431e09140514b0/pip-19.2.2-py2.py3-none-any.whl (1.4MB)

100% |████████████████████████████████| 1.4MB 17kB/s

Installing collected packages: pip

Found existing installation: pip 9.0.3 //更新前的版本号

Uninstalling pip-9.0.3:

Successfully uninstalled pip-9.0.3

Successfully installed pip-19.2.2 //更新后的版本号

## 4.查看第三方模块详情的命令：

C:\Users\web>pip show mysql-connector-python

Name: mysql-connector-python

Version: 8.0.17

Summary: MySQL driver written in Python

Home-page: http://dev.mysql.com/doc/connector-python/en/index.html

Author: Oracle and/or its affiliates

Author-email: UNKNOWN

License: GNU GPLv2 (with FOSS License Exception)

Location: c:\users\web\appdata\local\programs\python\python36-32\lib\site-packages

Requires: protobuf

Required-by:

# day05