6.29：卸载3.7改用3.6；成功从cmd将pygame库pip入；

Python3.4之后的版本自带pip，可通过cmd联网直接pip需要的库；

列表 []增删、切片以及相关函数的调用；

6.30：列表推导式：[变量 循环]；

元组，(),序列解包，生成器推导式，

字典，{},相关函数 读取添加修改，有序字典；

集合，{},元素不可重复，集合推导式。

Sorted（）方法（再看）；

复杂数据结构（再看）；

7.2：字符串相关函数、格式化(format())

字符串常用方法：1.find(),rfind(),index(),rindex(),count();

1. split(),rsplit(),partition(),rpartition();
2. join()（优先使用，运行效率>“+”）;
3. lower(),upper(),capitalize(),title(),swapcase(),
4. replace();
5. maketrans(),translate();
6. strip(),rstrip(),lstrip();
7. eval()（尝试把任意字符串转化为合法表达式并求值）

（用户输入直接eval可能会产生安全问题，应先检查和过滤）

1. 关键字in,not in;
2. startswith(),endswith();
3. isalnum(),isalpha(),isdigit(),isspace(),isupper(),islower();
4. center(),ljust(),rjust()

可变字符串；

字符串应用案例：加密和生成随机信息（random库）

7.3：正则表达式，操作字符串的重要方法。（\*\*）

re模块主要方法

7.4：直接使用re模块方法；

使用正则表达式对象；

（正则表达式内容复杂，在字符串操作中有着十分重要的作用，本次仅浏览相关函数和正则表达式的表达，具体操作在相关应用中加以研究。）

函数的定义：1.无需声明形参类型和函数返回值类型

2.可以嵌套定义

3.def 函数名(参数)：

‘’’注释’’’

函数体

形参与实参：形参用逗号分隔，单向值传递

若传递的参数是可变序列，调用后的修改是可以反映到函数外的

参数类型：1.默认值参数：

def 函数名(....,形参=默认值)：

可以通过显式赋值替换默认值

“函数名.\_defaults\_” 可以查看函数默认值的当前值，其返回一个元组

定义时带有默认值的参数必须在参数表的最右端

多次调用函数且不为默认值传递参数时，默认值参数只在第一次调用进行解释

1. 关键参数：

指函数调用参数时的传递方式

按照参数名进行传递，实参形参顺序可以不一致

1. 可变长度参数：

\*参数名：接收多个任意实参并放入元组中

\*\*参数名：接收类似关键参数的显式赋值参数并放入字典中

Python允许在定义函数时混合使用参数形式，但不建议。

1. 参数传递时的序列解包：

可使用列表、元组、集合、字典以及其他可代送对象加\*作为实参，解释器会自动进行解包，务必保证元素个数与形参个数相同。

7.5：return：结束函数的执行，返回一个任意类型的值（或空值）

变量作用域：

局部变量：1.属于某个函数，使用后删除

2.若局部变量名与全局变量名重复，局部变量优先

全局变量：1.定义 global

2.可在函数内外定义

3. 可编写专门的模块在同一程序的不同模块中共享全局变量

注：局部变量运行速度快于全局变量

全局变量可能会增加不同函数之间的隐式耦合度

除非必要，应尽量避免使用全局变量

lambda表达式：声明匿名函数，只能包含一个表达式，计算结果就是返回值

可调用其他函数，支持默认值参数和关键参数

7.6：一些算法案例，注意最大公约数和最小公倍数的算法；

高级话题：

map()可将一个单参数函数依次作用到一个序列或者代送器对象的每个元素

reduce(),（import functools）

（日后补充）

7.7 面向对象程序设计

类的定义与使用：

class Name: #新式类必须包含一个基类（已有的或者已设计好的）

def 成员名():

name=Name()

name.成员名()

isinstance():测试一个对象是否为某个类的实例

关键字 pass：为软件升级预留空间而“占座”

self参数

7.8 类成员与实例成员：

实例属性 and 类属性

Python可以动态地为类和对象增加成员

方法一般指与特定实例绑定的函数，用对象调用时对象本身作为第一个参数

私有成员与公有成员：

私有属性定义：用双下划线“\_\_”开头

1. \_xxx:保护成员，不能使用import导入，只有类对象和子类对象可以访问
2. \_\_xxx\_\_:系统定义的特殊成员
3. \_\_xxx:类中的私有成员，一般只有类对象自己可以访问，子类也不可访问；

或者在对象外部通过“对象名.\_类名\_\_xxx”访问

在IDLE交互模式下,”\_”表示解释器最近一次显示内容或语句正确执行的结果

方法：（课本内容杂乱）

7.10 属性（狭义）：读∈修改∈删除

代码例

特殊方法与运算符重载（实际应用中学习）

继承机制：对已有类进行二次开发

可继承父类公有成员

super() 和“基类名.方法名()”可在派生类中调用父类方法

支持多继承，父类有相同方法名子类使用时不指定，自动左至右搜索

（类的参数、使用、成员添加等较抽象，注意阅读课本代码例）

文件操作：

文本文件和二进制文件

文件打开模式：r（读），w（写），a（追加），b（二进制），+（读、写）

文件对象属性：closed（判断文件是否关闭，文件关闭返回ture）

mode（返回文件打开方式）

name（返回文件名）

7.12 文件对象常用方法：flush()

close()

read([size])

readline()

readlines()

seek(offset[,whence])

tell()

truncate([size])

write(s)

writelines(s)

f(文件对象名)=open(文件名[,’打开方式’[,缓冲区]])

（不同路径文件需给出路径）

上下文管理关键字：with

with open() as f:

7.13 序列化：数据不丢失其类型转化为二进制形式，可反序列化

pickle模块 和 struct模块

文件级操作：处理文件路径：os.path

使用命令读取文件内容：fileinput

创建临时文件或文件夹：tempfile

表示和处理文件路径：pathlib

MD5:判断文件是否被修改

部分案例精选

异常处理结构与程序调试：

自定义异常与异常类

7.14 异常处理结构:

1. try....except....

BaseException 异常基类可以捕获所有类型异常：

try:

.....

except BaseException as e:

...... #处理所有错误

1. try....except....else

没有异常执行else

1. 带有多个except的try结构：

try：

.... #被监控语句

except Exception1:

..... #处理异常1

except Exception2:

...... #处理异常2

1. try....except....finally:

finally语句无论是否异常都会执行

常用来清理try子句申请的资源

若finally之前的语句有异常，执行完后也会抛出

finally语句中的异常也会抛出

finally中尽量不要使用return

7.15 断言与上下文管理

断言： assert expression [,reason]

当表达式expresssion为真时什么都不做，为假时抛出异常

上下文管理： 关键字：with

with context\_expr [as var]

with块

任何情况下保证资源正确释放

使用sys模块回溯最后的异常

import sys

try:

block

except:

t=sys.exe\_info()

print (t)

sys.exe\_info() 的返回值是一个三元组(type,value/message,traceback)

（traceback 为调用栈信息的对象）

使用IDLE调试代码

Debug →Debugger，观察数据变化，类似于c

7.19 MOOC学习与实例操作

计算机语言以及发展、Python的深入理解

Pythondraw.py 原始版完成并正确运行

turtle库的使用：

左上角为绘图窗体原点

turtle.setup(width,height,startx,starty)

(控制窗体长宽以及左上角相对于屏幕左上角的位置)

(后两个参数可省略，默认为在屏幕正中间)

绝对坐标：画布中心为(0,0)

海龟坐标：以海龟方向划分前后左右

turtle.goto(x,y):直线前往坐标(x,y)

turtle.fd(d):海龟直线运行

turtle.bk(d):海龟向后运行

turtle.circle(r,angle):左侧某一个点为圆心曲线运行

turtle角度体系：类似于极坐标

turtle.seth(angle):使海龟改变角度（逆时针），不行进

7.20 RGB 色彩体系：

三通道：(r,g,b)

默认(0,1)，也可以转换为（0,255）

turtle.colormode(mode),改变取值范围

画笔控制函数：

turtle.penup()（别名pu）,画笔抬起

turtle.pendown()（别名pd），画笔落下

turtle.pensize(width)（别名width），画笔大小

turtle.pencolor():

参数1：”color”（字符串类型）

参数2：”r,g,b”（默认取值(0,1)）

参数3：(x,y,z)（元组型，默认(0,1)）

运动控制函数：

turtle.forward(x)（别名fd），直线行进x个像素(x可以为负)

turtle.circle(r,extant):

r:绘制半径，圆心于左侧距离r的位置（r<0右侧）

extant:绘制角度，默认为360°

方向控制函数：

turtle.sethheading(angle)（别名shth）：转向某一绝对角度

turtle.left & turtle.right：向左转和向右转（海龟坐标）

turtle.done 类似于system(“pause”)

7.21 实例编写：MOOC实例3：天天向上的力量

感受与C在编写时语法和逻辑的不同，同时带来的编写习

惯不同，面对python代码反应没有C快

ps:Python输出函数print的具体用法需要再深入了解

time库：

获取时间：

time.time():获取当前时间戳（1970.1.1至今的时间，单位second）

time.ctime():获取可读时间

time.gmtime():获取计算机可以识别的时间（程序交互）

7.22 时间格式化：

t=time.gmtime()

time.strftime(tpl,t)（tpl为时间格式化模板）

格式化字符串：

|  |  |  |
| --- | --- | --- |
|  |  |  |
| %Y | 年份 | [0000,9999] |
| %m | 月份 | [01,12] |
| %B | 月份全称 | January-December |
| %b | 月份简称 | Jan-Dec |
| %d | 日期 | [01,31] |
| %A | 星期全称 | Monday-Sunday |
| %a | 星期简称 | Mon-Sun |
| %H | 时间（24小时） | [00,23] |
| %h | 时间（12小时） | [00,12] |
| %p | 上午，下午 | AM,PM |
| %M | 分钟 | [00,59] |
| %S | 秒 | [00,59] |

time.strptime(str,tpl)：解析时间字符串为计算机可识别时间

tpl为时间字符串str的格式化表达

程序计时：

time.perf\_counter:返回一个CPU级的时间，单位s

计时：连续调用对两个变量赋值取差

time.sleep(s):程序休眠s秒

实例4编写：文本进度条

修改1.进度条刷新，\r和修改print的end参数

修改2.时间的现实

举一反三：非线性进度的进度条设置

7.23: 实例五：BMI指数计算

If中两条语句要换行或者使用分号（不同于C）

7.24: random 库的使用:

random.seed()：设置随机数种子（可以控制一下任何函数）

random.random():获取随机数（若未指定种子，则以系统时间为种子）

取值范围(0,1)

random. randint(a,b):生成[a,b]之间的一个随机整数

random.randrange(a,b[,k]):生成[a,b]之间步长为k的一个随机整数

random.getrandbits(k):生成一个k比特长的随机整数

random.uniform(a,b):生成[a,b]之间的一个随机小数

random.choice(sep):在列表sep中随机选取一个数

random.shuffle(sep):将sep列表随机打乱

实例六：圆周率编写

\可用于过长算式的换行书写

（本机蒙特卡洛方法循环10000\*10000次用时88秒）

实例七：七段数码管的绘制

无法正确控制画笔起落，原因不明

问题解决。记住数据类型！！！！计算机是死的

7.24： 函数主体编写完成，但有调用错误，再修改

7.25 模块化设计：

函数是初步抽象，对象是进一步的抽象

紧耦合：无法独立

松耦合：相互独立

——模块内部尽可能紧耦合，模块间尽可能松耦合

递归：调用自身

关键特征：链条，基例

PyInstaller 库：程序打包库

在源文件目录下使用cmd命令行执行： pyinstaller-F <文件名.py>



 在cmd命令行下输入PyInstaller+参数

修改图标：pyinstaller -i<tubiao.ico> -F<wenjianming.py>

7.27: 实例八：科特雪花图形绘制

7.28： 组合数据类型：

集合：不可变的数据类型组合

用{}或者set()生成（空集合必须用set()）

元素唯一，逗号分隔，没有顺序

集合运算：

关系操作符：

S|T：并运算

S&T：交运算

1. T：减运算

S^T：补运算

1. T或者S>=T:返回True或False，判断包含

S<T或者S<=T:返回True或False，判断子集

增强运算符：

S|=T：更新S，将S与T的并赋给S

S-=T：更新S，将S与T的差赋给S

S&=T：更新S，将S与T的交赋给S

S^=T：更新S，将S与T的补赋给S

7.31： 集合处理：

S.add(x):往集合S中增加元素x

S.discard(x):在集合S中删除元素x，若x不存在，不报错

S.remove(x):在集合S中删除元素x，若x不存在，抛出KeyError

S.clean():移除S中所有的元素

S.pop():随机返回一个S中的元素并更新S，S为空抛出KeyError

S.copy():返回一个S的副本

Len(S):返回集合中元素个数

x (not) in S：判断x是否在S内，返回True或False

集合最典型应用：数据去重

序列数据类型：

元素之间存在顺序关系，类型可以不相同

序列是一个基类衍生出字符串、元组、列表

元组基本继承列表的相关操作，但因其不能改变所以没有特殊操作

列表：

s[i]=x:替换列表s中第i个元素为x

s[i:j:k]=t:用列表t替换列表s切片后对应元素

del s[i]：删除列表s中第i个元素

del s[i:j:k]：删除列表s中第i到j步长为k的元素

s\*n：列表s重复n次（作用于原列表）

s.append(x):在列表最后增加一个元素x

s.insert(i,x):在列表第i个位置增加一个元素x

s.pop(i):在列表第i个位置取出该元素

s.remove(x):在列表第一次出现的元素x删除

s.reverse(x):在列表的元素反转

8.1 MOOC实例九编写：基本统计数据分析

sorted(s):对列表s进行排序（改动原列表）

8.2 字典：键值对的集合，键值对之间无序

{}或者dict{}生成，键值对用:表示

{}系统保留给了空字典

字典操作函数：

del d[k]:

k in d:键k是否在d中

d.keys():返回字典d中所有的键信息

d.values():返回字典d中所有的值信息

d.items():返回字典d中所有的键值对信息

d.get(k,<default>):键k存在，则返回相应值，不存在返回default

d.pop(k,<default>):键k存在，则取出相应值，不存在返回default

d.popitem():从字典d中随机取出一对键值对，并以元组形式返回

d.clear() len(d)

jieba 库：中文操作库

精确模式：精确切分文本，无冗余

全模式：所有可能的词组合，有冗余

搜索引擎模式：精确模式基础上，再次切分长词

jieba.lcut(s):精确模式，返回一个包含分词结果的列表

jieba.lcut(s,cut\_all=True):全模式，返回一个包含分词结果的列表

jieba.lcut\_for\_search(s):搜索引擎模式，返回一个包含分词结果的列表

8.3： MOOC实例十：文本词频统计

items.sort(key=lambda x:x[1],reverse=True) 以值的大小进行排序

文件和数据格式化：

文件：

文本文件：单一二进制编码组成，可视作长字符串

二进制文件：比特0，1构成，没有统一编码

文件打开：

f=open(“f.txt”,”rt”):文本形式打开文件,只读

f=open(“f.txt”,”rb”):二进制形式打开文件

f=open(“f.txt”,”w”):文本形式，覆盖写模式

f=open(“f.txt”,”a+”):文本形式,追加写模式+读文件

f=open(“f.txt”,”x”):文本形式，创建写模式

文件读取：

<f>.read(size):读出前size个字符，默认读出全部

<f>.readline(size):读入一行，给出参数则读出前size

<f>.readlines(size):读入所有行，以每行为一个元素形成一个列表

文件写入：

<f>.write(s):向文件写入一个字符串或者字节流

<f>.writelines(s):向文件写入一个全为字符串的列表

<f>.seek(s):

s=0,返回文件开头

s=1,文件当前位置

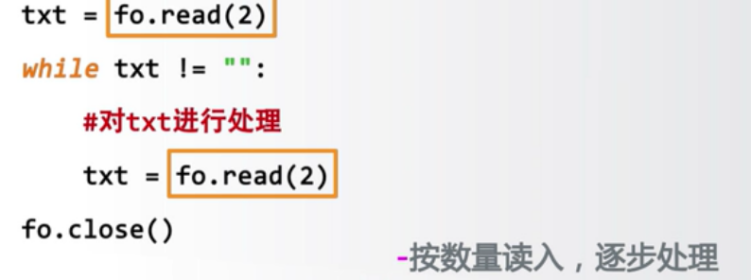
s=2,到文件结尾

文件的全文本操作：

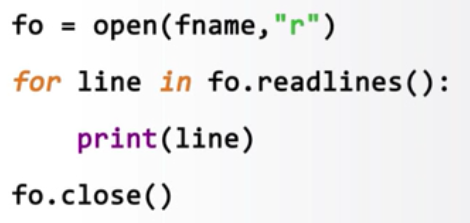
直接操作：不适用大体量文本

分量操作：

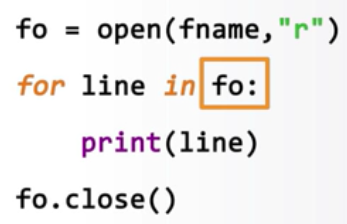
自定义量：



分行处理：

 读入全部，分行处理

分行读入，分行处理



8.4： MOOC实例十一：自动轨迹绘制

采用数据绘制图形而不是代码

基本思路：1.定义数据文件格式（接口）

1. 编写程序，根据接口解析参数并绘制图形
2. 编制数据文件

map(1,2)：将1作用于2（1,2可以是函数）

文本里面不要出现空行！会报错

自动化思维：数据和功能分离，通过数据传递进行驱动

接口化设计：格式化接口设计，间接高效

二维数据应用：应用维度组织数据

8.5： 一维数据的格式化与处理：

表示：列表或集合

存储：空格分隔，逗号分隔，特殊字符（组合字符）分隔

处理：字符串方法 join(),split()

二维数据的格式化与处理：

表示：二维列表

存储：

CSV：comma-Separated Values

国际通用一二维数据存储格式，拓展名为.csv

每行一个一维数据，逗号分隔，无空行

Excel可以输入输出，一般编辑软件都可产生

8.6 程序设计方法学：

计算思维，计算生态（论坛寻找优秀第三方库）

用户体验，进度，容错，纠正，速度，异常处理

配置化设计，程序引擎+数据

IPO，模块化，配置化，自顶向下

更大的Python世界：

<http://pypi.org/>

第三方库的安装：

pip：cmd命令行下

pip install <库名>，

pip install U-<库名>（升级库）

pip uninstall <库名>（卸载库）

pip download <库名>（下载但不安装库）

pip show <库名>（列出库的详细信息）

pip search <关键词>（根据关键词搜索库）

Anaconda：<https://www.continuum.io>

数据分析较标准的第三方

集成800多个库，包含主流计算工具

os库：通用、基本的操作系统交互

路径操作：os.path子库

os.path.getsize(path):返回路径文件大小(字节)









8.7 os.system (command): 执行指定地址的程序

执行命令command

返回0表示正确执行

可为指定程序添加参数，空格分隔

（如图片处理器打开图片）

指令以字符串形式输入

os库环境相关函数：

从数据处理到人工智能（概览）：

数据表示->数据清洗->数据统计->数据可视化->数据挖掘->人工智能

数据分析库：Numpy:Python接口，C编写，速度快，N维数组最基本库

Pandas：数据分析高层次应用

SciPy：数学、科学和工程计算功能库

数据可视化：Matplotlib：高质量二位数据可视化库

Matplotlib.pyplot可调用各个可视化效果

Seaborn：数据间分布、分类和线性关系

Mayavi:3D数据可视化

文本处理库：PyPDF2：处理pdf文件集

信息获取，分隔和整合文件，加密解密文件

NLTK:自然语言文本处理

自然文本分类、标记、语法句法、语义分析

Python-docx：创建或更新word文档

可增加并配置段落、图片、表格、文字等

机器学习库：Scikit-learn：机器学习方法工具集

提供统一化机器学习方法功能接口

聚类、分类、回归、强化学习

TensorFlow：AlphaGo背后的机器学习框架

谷歌机器学习框架

数据流图作为基础，图节点为运算，边为张量

MXNet：基于神经网络的深度学习框架

从Web解析到网络空间：

爬虫：

Requests：最友好的网络爬虫库（页面级）

简单的类HTTP协议爬虫功能

支持连接池、SSL、Cookies、HTTP代理等

Scrapy: 专业网络爬虫框架

批量和定时网页爬取、提取数据处理流程

Pyspider:提供完整的网页爬取系统构建功能

支持数据库后端、信息队列、分布式架构、优先级

Web信息提取：

Beautiful Soup：HTML和XML的解析库，可加载多种引擎

长与网络爬虫库搭配使用

Re：正则表达式的一批通用功能

Python-Goose:提取文章类型web网页的功能库

文章信息、视频等元数据的提取

Web 网站开发：

Pyramid：适合规模中等，快速构建

产品级Web开发，起步简单，可拓展性好

Flask：最简单Web系统应用构架

简单，规模小，快速，适合几个网页的开发

Django：最流行Web应用构架

模型M（model）模板T（template）视图V（Views）

Web网络应用开发：

WeRobot：微信公众号开发工具

解析信息和反馈信息，建立微信机器人

aip：访问百度AI服务的Python接口

语音、人脸、OCR、NLP、知识图谱、图像搜索

MyQR：生成二维码（基本、艺术、动态）

从人机交互到艺术设计：

图形用户界面：

PyQt5：提供创建Qt5程序的Python API接口，具有完备的GUI

wxPython：专用于Python的跨平台GUI框架

Python最主要数据分析功能库

PyGObject：使用GTK+开发GUI的库

（GTK+：跨平台用户界面）

Anaconda使用该库构建GUI

游戏开发：

PyGame：基于SDL的简单游戏开发

理解游戏对外部输入的响应机制以及角色构建

Panda3D：开源、跨平台的3D渲染和游戏开发库

提供c++和Python两种接口

迪士尼和卡耐基梅隆大学共同开发

cocos2d：基于OpenGL的游戏开发图形渲染

支持GPU加速，树形结构分层管理游戏对象类型

适用于专业2D游戏开发

虚拟现实：

图形艺术：

8.8 实例，七段数码管绘制

依然报错：AttributeError: 'str' object has no attribute 'dn'

实例13：体育竞技分析

自顶向下，自底向上

主体框架编写完成，运行正确

8.9 实例14：自动安装第三方库

面向对象特点：封装，多态，继承

最终目的是提高代码灵活性和复用

可定义同名函数

至此，MOCC第一期学习完成

8.10 MOOC第二期：网络爬虫实战（观摩）

网络爬虫原理：

开发人员界面，网页源码，监听器

大多数文本资料在Doc部分

环境安装：Chrome浏览器，其开发者选项有较多内容

anaconda安装，集成python、Ipython以及众多包，套件完全兼容

爬虫过程整体感知：获取网页传送方式，爬取内容，内容筛选，整理打包

撰写爬虫前的准备：网络基础原理，一些包的使用

8.12： 知乎相关文章：《爬虫基本原理》

import request

from bs4 import BeautifulSoup

引言：爬虫是自动抓取网页信息的代码，对象必须是已经看到的网页

URL：统一资源定位符（Uniform Resource Locator），是互联网上标准的资源（不但是网页）的地址（Address）

网页请求：通过函数request.get可将网页代码返回给指定变量

网页解析：通常使用bs4

HTML介绍：

定义：超文本标记语言

浏览器接受服务器传回的HTML源代码渲染成相应网页

文本通过特定保留标签区分和显示（标签将文本夹在中间）

提取常用方法：正则表达式

标签的属性：css样式

chrome检查工具：右键，检查

可切换入Element界面中查看网页结构