用pip安装模块：python –m pip install <module>

从<http://www.lfd.uci.edu/~gohlke/pythonlibs/> 下载whl文件 用pip install 安装

查看已安装模块：pip list

查看pip支持的版本：在shell中输入import pip; print(pip.pep425tags.get\_supported())

交互模式：cmd -> python

文件开头：

#!/usr/bin/env python3

# -\*- coding: utf-8 -\*- 按照UTF-8编码读取源代码

输出： print(‘hello’, ‘world’) 换行：\n Tab: \t

输入： name = (‘please enter your name’)

动态语言：变量赋值时不需要指定类型

除法： / -> 浮点数 // -> 取商 % -> 取余

字符->ASCII：ord() ASCII->字符：chr()

Python中的字符串：**unicode**，用于网络传输或保存磁盘时需转换成bytes:

'ABC'.encode('ascii') or （含有中文时只能用utf-8） '中文'.encode('utf-8')

bytes->str: b'ABC'.decode('ascii') or b'\xe4\xb8\xad\xe6\x96\x87'.decode('ascii')

raw string -> r'C:\Users\yunhao\Desktop\train.csv’ -> 保留原字符串不转义

计算str字符数: len()

截取空格前的字符：str.split(‘ ’)[0]

改变字符串中的字符：a = 'abc' a = a.replace('a', 'A')

格式化字符串：'Hi, %s, you have %d dollars.' % ('John', 10)

（占位符 - 整数：%d 浮点数：%f 字符串：%s）

补0：'%02d'%3 保留两位小数：'%.2f' % 3.14159

String format: print(’{} like {}’.format(‘Wendy’, ’John’))

**list**（中括号）: name = ['Michael', 'Bob', 'Tracy']

访问list: name[0] ... name[len(name)-1] or name[-1], name[-2]... （从最后一个倒数）

在末尾增加元素： name.append('Adam') 在指定位置增加元素： name.insert(1, 'Jack')

删除指定位置的元素： name.pop(i) 删除末尾的元素： name.pop()

合并list：name = name1 + name2

查找某个元素是否在list中：’Bob’ in name -> True

把某个元素替换成别的元素： name[1] = 'Sarah'

list的元素也可是list： s = ['python', 'java', ['asp', 'php'], 'scheme'] （s[2][1] -> ‘php’）

对list元素排序：改变list：a.sort() 不改变list：sorted(a)

**tuple**（小括号）: 一旦初始化就不能修改 -> 更安全 name = ('Michael', 'Bob', 'Tracy')

只有1个元素的tuple定义时必须加一个逗号： t = (1,)

多个变量同时接收一个tuple，按位置赋给对应的值：x,y,z = name

**dict**（大括号）：d = {'Michael': 95, 'Bob': 75, 'Tracy': 85} 通过key查找value：d['Michael']

通过key改变value：d['Michael'] = 88 通过in判断key是否存在：'Thomas' in d -> False

把数据放入key：d['Adam'] = 67 删除key：d.pop('Bob')

key必须是不可变对象：str, int等，不能为list

遍历key/value: for score in d.keys()/d.values()

**set**（无序、无重复key、不能放入可变对象list）：s = set([1, 2, 3])

添加元素到set：s.add(4) 删除元素：s.remove(4) 交集：s1&s2 并集：s1|s2

条件：if a>0: ... elif... else:... if 'python' in s:...

循环：for i in range(10): ...

while n>0:...

break：提前结束循环 continue：跳过当前的这次循环，直接开始下一次循环

数据类型转换：int() float() str() bool()

定义函数： def myfun (x): 占位符：pass（什么也不做，让编译通过）

参数数据类型检查：if not isinstance(x, (int, float))

（抛出错误信息） raise TypeError('bad operand type')

返回多个值（实际为返回turple）：return x,y

数学函数：import math后可以使用math.sqrt, math.sin等

默认参数必须指向不变对象：def enroll(name, gender, age = 6, city = 'Beijing'):

可变参数：def calc(\*numbers)

for n in numbers:...

关键字参数（在函数内部组装成dict）：def person(name, age, \*\*kw):

调用：person('Bob', 35, city='Beijing'，job='Engineer')

命名关键字参数（限制关键字参数的名字）：def person(name, age, \*, city, job):

切片Slice（取list/tuple/str指定索引范围）：

L[0:10:2] -> 从索引0开始，到10为止（不包括10），每2个取一个

L[:3] -> 从索引0开始，到3为止（不包括3）

倒序输出一个list：L[::-1]

列表生成式：[x \* x for x in range(1, 11) if x % 2 == 0]

[x \* y for x in range(10) for y in range(10)]

函数式编程：import functools

map（将传入的函数依次作用到序列的每个元素）：map(lambda x: x\*x, [1, 2, 3, 4, 5, 6, 7, 8, 9])

reduce把一个函数作用在一个list[x1, x2, x3, ...]上，这个函数必须接收两个参数，reduce把结果继续和序列的下一个元素做累积计算，需要用list()函数返回所有结果

filter接收一个函数和一个list，把传入的函数依次作用于每个元素，然后根据返回值是True还是False决定保留还是丢弃该元素，需要用list()函数返回所有结果

sorted()可以接收一个key函数来实现自定义的排序，key指定的函数将作用于list的每一个元素上，并根据key函数返回的结果进行排序：sorted([1,-2,-3,4], key = abs, reverse = True)

返回函数并未马上执行，因此不要引用任何后续会发生变化的量

匿名函数（返回值即为表达式结果）：lambda x: x\*x (3) -> 9

偏函数：把一个函数的某些参数固定住（设置默认值），返回一个新的函数，调用这个新函数会更简单

（str->二进制int）int2 = functools.partial(int, base=2)

使用模块： import sys (导入后利用变量sys该模块所有功能)

类似\_xxx和\_\_xxx这样的函数或变量是非公开的（private），不应该被直接引用

Python可以自由地给实例变量绑定属性：bart.name = 'Bart Simpson' （给实例bart绑定name属性）

私有变量：变量名以\_\_开头，无法从外部访问bart.\_\_name（通过set，get等方法从外部设置/获取）

类：class Animal(object):... class Dog(Animal):... 括号内为父类，继承的子类获得父类的全部功能

测试对象是否含有属性：hasattr(obj, 'x') 设置属性'y'：setattr(obj, 'y', 9) 获取属性'y'：getattr(obj, 'y')

（跳过面向对象高级编程）

错误处理：try:...except...finally:... 用logging模块记录错误，程序打印完错误信息后继续执行

断言：assert n != 0, 'n is zero!' 若断言失败，则抛出AssertionError: n is zero!

断点调试：import pdb，在可能出错的地方放置断点pdb.set\_trace()，查看变量n：(Pdb) p n，继续运行：(Pdb) c

读文件（自动close）：with open('/path/to/file', 'r') as f:

print(f.read())

写文件：with open('/Users/michael/test.txt', 'w') as f:

f.write('Hello, world!')

在内存中读写str：StringIO 二进制数据：BytesIO

查看当前目录：os.path.abspath('.')

在某个目录下创建新目录：os.path.join('/Users/michael', 'testdir')

os.mkdir('/Users/michael/testdir')

删掉一个目录：os.rmdir('/Users/michael/testdir')

拆分路径（得到文件拓展名）：os.path.splitext('/path/to/file.txt')

文件重命名：os.rename('test.txt', 'test.py')

删除文件：os.remove('test.py')

列出当前目录下的所有目录：[x for x in os.listdir('.') if os.path.isdir(x)]

列出所有.py文件：[x for x in os.listdir('.') if os.path.isfile(x) and os.path.splitext(x)[1]=='.py']

requests, urllib

正则表达式

* urllib.request.urlopen(url, data, timeout)

url: 即为URL，data: 访问URL时要传送的数据，timeout: 设置超时时间

第二三个参数是可以不传送的，data默认为空None，timeout默认为 socket.\_GLOBAL\_DEFAULT\_TIMEOUT

数据传送：

* POST：把参数写入data

data = urllib.parse.urlencode(values)

request = urllib2.Request(url,data)

* GET：直接构建一个带参数的URL

geturl = url + "?"+data

request = urllib2.Request(geturl)

设置headers：user\_agent, referer等

user\_agent = 'Mozilla/5.0 (Windows NT 6.3; WOW64; rv:51.0) Gecko/20100101 Firefox/51.0'

headers = { 'User-Agent' : user\_agent }

正则表达式：.\*? 是一个固定的搭配，.和\*代表可以匹配任意无限多个字符，？表示使用非贪婪模式进行匹配，也就是尽可能短地做匹配

(.\*?)代表一个分组

re.S标志代表在匹配时为点任意匹配模式，点. 也可以代表换行符

打开文件：file = open(title +”.txt”,”w+”)

写入数据：file.write(item)

模拟登陆，验证码识别