# Learn python 3 the hard way

## Exercise 1

Python用“print()”输出打印；

Python语法结尾不含分号“;”；

Python的单引号双引号都能用来表示要输出的字符串；

例：print(“hello world!”)

在powershell中用python命令运行python。

在powershell中用python (文件名)语法编译运行文件。

例：python ex1.py。（python文件以py作为后缀）

Python中用“#”号前缀代表注释

## Exercise 2

Python中用“#”号前缀代表注释

## Exercise 3

Python中的数学符号：+ - \* / % >= > <= <；

Python中/号是除，可以计算到小数，1/4 = 0.25；

%号是取余，8%3 = 2, 10%5 = 0；

Python的print()函数里直接用“,”分隔打印内容；

大于小于等于等等比较符的输出结果是true或者false

## Exercise 4

Python变量用小写和下划线表示

=号是赋值，==号是判断是否相等

操作符前后要加空格，方便阅读

## Exercise 5

python的print字符串前面加f表示格式化字符串，加f后可以在字符串里面使用花括号括起来的变量和表达式，如果字符串里面没有表达式，那么前面加不加f输出应该都一样.替换字段是表达式，在运行时进行评估，然后使用format()协议进行格式化。

例如：print(f ”some words {变量}”)。

在给变量赋值字符串或字符时，python里单双引号都可以。

Python取整函数round()（四舍五入）。

## Exercise 6

Python变量可以直接用format格式赋值字符串和变量

format可以嵌套使用，还有一种特殊用法.format()

例如：hilarious = False

joke\_evaluation = "Isn't that joke so funny?! {}"

print(joke\_evaluation.format(hilarious))

## Exercise 7

在print函数最后跟一段end = ‘ ’的输出，可以不换行接下一句打印，例如print(“a”,end = ‘ ‘)接print(“b”)。

## Exercise 8

Python的format函数会把所有字符串里的{}在打印时，理解为一个放变量的地方。具体例子见ex8.

## Exercise 9

和其他语言一样，python中\n也是代表换行

在打印函数中用前后三个双引号括起来的内容可以长输出也可以跨行。

例如：print(“““some thing”””).

## Exercise 10

Python中的转义字符，见ex10的表。

常用的有：\n（换行），\t（相当于tab，打印空格）

## Exercise 11

对于python3,通过input函数输入的所有内容都会作为str类型的字符串变量传入，只需要使用int和float进行强制类型转换就可以。例如：d=float(input('Please enter what is your initial balance: \n'))

## Exercise 12

Input（）函数可以在括号里写入想要显示的字符串，例如：a = input(“plz input number:”)。

可以在powershell中用python –m pydoc input查看Input函数的使用和解释，如下：

Read a string from standard input. The trailing newline is stripped. The prompt string, if given, is printed to standard output without a trailing newline before reading input.

## Exercise 13

from sys import argv, 从系统中导入（输入）参数变量（实参），这是一种python里带参数的运行写法，运行的时候必须给出指定个数的参数。

## Exercise 14

from sys import argv和input函数混合使用，通过批量输入的实参的变量，也能通过format打印，用法跟普通变量一样。

## Exercise 15

Python 中用open(文件名)函数打开文件，赋给一个变量，就可以直接用print打印出来，例如：txt = open(a.txt) print(txt.read())。

.read（）是文件的一个功能或者称为命令。用open()打开的是一个文件，而这个文件可以做很多的操作或者命令，read是其中一个。

## Exercise 16

Python中的文件操作命令：

close：关闭文件（并保存）；read：读取文件，可以赋值给一个变量；readline：读取一行；truncate：清空文件内容；write()：写文件，添加内容；seek(0)：回到文件起始位置。

打开一个文件，如果该文件不存在，那么python会在当前文件夹下面创建一个该名字的文件。对文件做写操作的时候，在打开文件的时候一定要带”w”权限（写权限），另外还有“r”权限，”a”追加。

最后记得操作了文件最后一定要调用close。

## Exercise 17

本节练习目的是把一个文本文档的内容复制到另外一个里面去，本节如果跟着作者在中文windows环境下运行可能有一个隐含的bug，可通过多种方式规避。

From os.path import exists命令引入了一个命令exists，用于判断文件是否存在，返回值为true或者false，用法exists(“文件名”)。

Len(“指向文件的一个变量”)，该命令可以计算出文件中字符的长度。

在powershell和bash下，可以用echo “字符串” > 文件名。的形式创建一个不存在的文件，并写入字符串内容和打印在标准输出屏幕上。

但是！在中文环境下powershell默认创建文件的编码模式是utf16格式（英文系统是utf8），所以当用python以“utf8”格式去去读取该文件的时候，会报错，此时最简单的方法是，不要用powershell命令去创建文件，直接手动创建一个文本文件。

## Exercise 18

Python用def来定义函数（python中称为功能）形式是

def 函数名（参数名）：

（四个空格）函数体

没有四个空格的第一句代表非函数体的其他内容。

函数参数可以是多个参数也可以是没有参数，这一点和其他语言相同，有一点不同的是，python中可以使用带星号的参数，带星号的参数可以传入N个实参，可以是0个或者多个。

## Exercise 19

本节主要教导了给函数的形参传递参数的内容，跟其他编程语言相同，可以直接传递，可以传变量，也可以变量+直接传递。

需要注意的是，python 的函数定义一定有冒号：，在本节练习中忘记了打冒号导致报错

## Exercise 20

本节内容是练习在函数定义中，在函数体内调用文件操作。

需要注意的是每一次文件操作readline都是在一行中去读取字符直到遇到第一个换行符\n，以及每次输出readline都会跟一个\n换行符

## Exercise 21

本节内容是python的函数也可以像其他语言一样，用return返回一个返回值，并且函数返回值可以做其他函数的参数，与其他语言基本相同。

## Exercise 22

本节是复习之前以上学习的内容。

## Exercise 23

这一节主要讲了文件的编码，和数据在电脑中的存储形式。数据以2进制存储在电脑中，每8位组合成一个字节，国际常用字符编码为ASCII码和UTF-8编码。

在python用open函数打开文件的时候，除了文件名这个参数，还可以带一个encoding="utf-8"的编码格式的参数。Python中想把字符串编码成某一种编码（如：UTF-8），需要用encode方法，而要解码需要用decode方法。

Strip方法可以去除字符串两端指定的内容，默认为去除字符串首尾两端的空字符和换行符。Python中的函数的返回值可以调用函数本身，加上条件判断可以做出一个有条件的循环。

详见23节的代码。

## Exercise 24

本节是做前面所有章节内容的练习，Python中比较特殊的一点是函数的返回值可以是多个，比如可以返回三个变量的值，这一点跟C语言不同，C语言只能返回一个返回值。

格式化字符串format方法调用与之前不同的是，不用在前面用f””的形式，可以在字符串后调用，形如.format(变量名)。

## Exercise 25

一系列对字符串的操作函数。

split() 通过指定分隔符对字符串进行切片，如果参数 num 有指定值，则分隔 num+1 个子字符串。语法：str.split(str="", num=string.count(str)).默认按所有空格来分，空格和\n和\t。

sorted方法排序，与sort方法有区别。

pop方法，移除列表中的一个元素，并返回这个对象，默认为最后一个元素。

每个函数下面跟一个”””字符串”””括起来的字符串，可以作为函数的说明信息，能被help（文件名.函数名）打印出来。

运行函数的时候，可以from 文件名 import \*，跟C一样，\*号是通配符，表示导入该文件里所有函数。

## Exercise 26

改错练习

## Exercise 27

基础逻辑，与（and），或（or），非（not）。

逻辑符号：！=，==，>=，<=，True，False。

## Exercise 28

基础逻辑练习。

## Exercise 29

逻辑判断语句if，python中的if语句语法：

if 表达式 ：

代码块（内容之前是4个空格，或者两个tab）

（无空格）if结束后的其他语句

## Exercise 30

Python中的逻辑选择语句，也有if和else，还有elif(else if)，语法跟if是一样的。

## Exercise 31

Python中也有if语句的嵌套，需要注意的一件事是，python中语句属于哪个if else块是按照缩进来划分的，不注意的话容易出错。

## Exercise 32

本节引入两个概念，一个是for循环，一个是讲了一下python中的list，列表。

Python中for循环用法举例：for i in range(0, 6):

注意循环只会从range的第一个数做到末尾之前的一个数，即从0到5。另外一种用法是 for 形参 in 某个List：，可以遍历这个列表。

发善一下关于list列表的内容，python的列表的作用跟C中的数组类似，但是本质上是不同的。区别如下。

Python中的列表不是C数组的固定长度，列表是可变长的。C中的数组必须规定一个类型，且只能存储该类型，而python的列表可以存储任意类型的数据。

列表的一些用法。

**（1）添加**

* 往列表a的尾部追加元素：a.append(i)

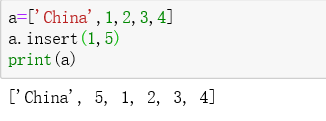
aList = [123, 'xyz', 'zara', 'abc'];

aList.append( 2009 );

print "Updated List : ", aList;

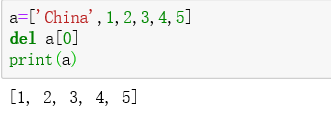
Updated List : [123, 'xyz', 'zara', 'abc', 2009]

* 往列表a中的任何位置追加元素：a.insert(i,elem),其中i表示索引位置，elem表示元素。

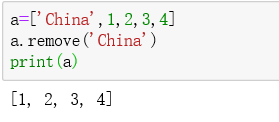


**（2）删除：**

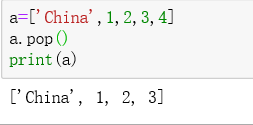
* 删除指定位置的元素：del a[i]



* 删除列表中的元素:　　a.remove(elem)

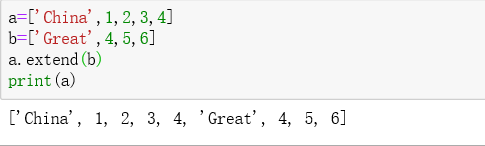


* 删除列表尾部的元素(有点像栈pop哈哈)：a.pop()

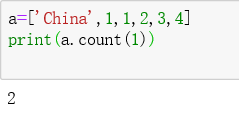


**（3）拓展：**

* 将列表b追加到a的尾部：a.extend(b)



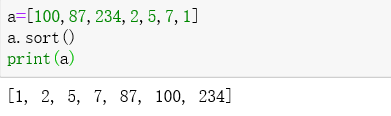
**（4）统计元素出现的次数：a.count(elem)**



**（5）排序：list.sort(self,key=None,reverse=False)其中key只在元素为字符串时才有用，表示按照关键字（如字符串长度）进行排序**

**数字：**

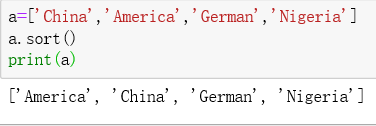
* 按照从小到大排序(默认从小到大排序)：a.sort()



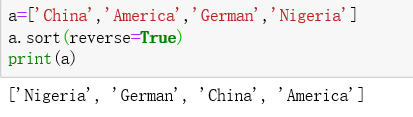
* 若想要按照从大到小排序，则为：a.sort(reverse=True)

**字符串：**

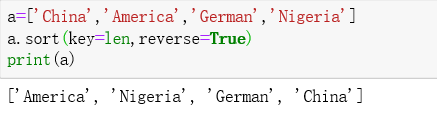
* 按照ASCII码的大小按照从小到大排序：



* 从大到小排序：a.sort(reverse=True)

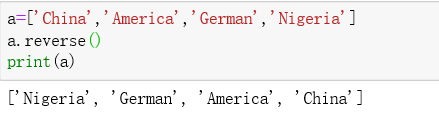


* 按照字符串的长度进行从大到小排序：a.sort(key=len,reverse=True)



**（6）翻转：**

* 列表中元素翻转：a.reverse()



**（7）返回列表a长度：len（a）**

**（8）获取元素下标：list.index(elem)**

## Exercise 33

引入python中的while循环，用法和for一样，功能和一般语言的while循环相同。

使用形式：while i < 6:

## Exercise 34

引入列表的index概念，即列表第一个数存储在位置0，list[0]。跟其他语言的数组一样。

## Exercise 35

本节练习定义了几个函数，python中函数的申明定义必须在引用之前，不可以在引用之后定义（因为python没有预编译）。而python中函数之间的调用不需要先后顺序，因为python在执行的时候，是先只记住定义了的函数的名字，不会去检查函数的内容，所以只要把所有需要调用的函数写在真正调用之前就行了，python在真正调用的时候才会去执行函数里的内容，而当在执行时再碰见别的函数，那就会再去那个相应的函数名里执行。

本节做了while True的无限循环，引入了exit这个函数，exit(0)代表无错误退出，exit(1)代表有错误退出。

本节引用了一个in的用法，例如：

if key in dict

意思是判断dict里面有没有key，注意直接从input输入的东西是字符类型，所以这里的key和dict也是按字符来比较的，例如key是“0”，那么只要dict里含有“0”字符那就算包含，比如“430”按字符算的话，就包含“0”。

## Exercise 38

本节做了一些有关于字符串和列表的练习，split()方法可以根据需求将字符串切成字符列表，例如：stuff = ten\_things.split(' ')。

而join方法可以把字符串列表根据相应的要求，拼接成一个字符串。例如：print(' '.join(stuff))。以及一些其他对列表的操作，append和pop之类的，都在之前的练习见到过。

## Exercise 39

Python一个重要概念dictionary，字典，一种类似于列表的数据存储结构。字典的每个键值(key=>value)对用冒号(:)分割，每个对之间用逗号(,)分割，整个字典包括在花括号({})中 ,格式如下所示：

d = {key1 : value1, key2 : value2 }

键必须是唯一的，但值则不必。字典的内容可以直接添加。

Python 字典(Dictionary) get() 函数返回指定键的值，如果值不在

字典中返回默认值。

dict.get(key, default=None)

* key -- 字典中要查找的键。
* default -- 如果指定键的值不存在时，返回该默认值。

## 返回值

返回指定键的值，如果值不在字典中返回默认值None。

## Python 字典(Dictionary) items() 函数以列表返回可遍历的(键, 值) 元组数组。

以下实例展示了 items()函数的使用方法：

dict = {'Google': 'www.google.com', 'Runoob': 'www.runoob.com', 'taobao': 'www.taobao.com'}

print "字典值 : %s" % dict.items()

# 遍历字典列表

for key,values in dict.items():

print key,values

以上实例输出结果为：

字典值 : [('Google', 'www.google.com'), ('taobao', 'www.taobao.com'), ('Runoob', 'www.runoob.com')]

Google www.google.com

taobao www.taobao.com

Runoob www.runoob.com

## Exercise 40

本节介绍引入了面向对象编程的概念，介绍了class类，python类的使用还是有点不同的，类定义和使用范例如下：

class MyClass:

"""一个简单的类实例"""

i = 12345

def f(self):

return 'hello world'

# 实例化类

x = MyClass()

# 访问类的属性和方法

print("MyClass 类的属性 i 为：", x.i)

print("MyClass 类的方法 f 输出为：", x.f())

其中有所不同的是，python的类一般需要定义构造函数来传递参数，用法例子如下：

实例(Python 3.0+)

#!/usr/bin/python3

class Complex:

def \_\_init\_\_(self, realpart, imagpart):

self.r = realpart

self.i = imagpart

x = Complex(3.0, -4.5)

print(x.r, x.i) # 输出结果：3.0 -4.5

其中的self也是一个特殊的用法，它代表了类的实例，而不是类定义本身，self 代表的是类的实例，代表当前对象的地址，而 self.class 则指向类。所以不用self也行，用其他形参名代表也是可以的。

## Exercise 41

练习41的代码过于复杂，本来只是作者用来练习类的读法的，用了很多很高深的写法，具体代码逻辑和函数见代码注释。

## Exercise 42

练习42教了对象，类，和实例的关系，教了父类和子类的关系，教了实例和类的关系，以及给出了一个继承父类变量的案例。注意初始化函数前后是两个下划线\_\_。

## Exercise 43

这是一个很长的节，这一节教我们如何来分析和建立一套对象。分析过程如下：

1. Write or draw about the problem.

2. Extract key concepts from 1 and research them.

3. Create a class hierarchy and object map for the concepts.

4. Code the classes and a test to run them.

5. Repeat and refine.

1写出所有相关的名词和动词，并描述他们之间的关系。最好是能整理为一个表格或者画出一个关系图。

2.要提取出第一步里描述的关键词信息，主要是各种名词，然后研究他们有没有上下级继承关系，然后删除掉类似的名词，把名词按上下级关系列成一个表。为我们的名词添加一些他们需要的动词在下面。

然后写出类的框架的代码，然后分析需要添加的内容或关系，重复以上分析的步骤。

## Exercise 44

面向对象中类的继承，一个子类可以自动继承父类所有的函数和参数变量。也可以通过重写同名的函数名来override父类中的函数。第三种方式是，当已经override了父类的函数时，可以通过调用super()函数来再次调用父类里的同名函数。

第一种特殊的方法是类组合，见代码ex44e，不需要两个类有上下继承的关系，通过类似函数调用的方法，也可以实现继承的一些功能。

## Exercise 46

创建虚拟环境的工程文件目录结构。

每次想要进入和使用虚拟环境就在windows根目录输入以下代码:

.\.venvs\lpthw\Scripts\activate

就可以进入该虚拟环境。详情见该节书内容。

## Exercise 47

47节利用上一节学习到的虚拟环境工程，新建立了一个简单的python代码，用来学习单元测试代码，通过编写测试文件，运行之后来进行自动化测试，具体内容见书本。

每次通过tests文件夹里的XXX\_tests.py文件来写入想要单元测试的代码的内容，然后在主目录输入命令nosetests来运行测试。

## Exercise 48

48节要优化我们的输入行为判断，要有更好的容错性，所以要建立一个语句的“单词表”，因为单词表内容是不变的，所以这里引入了一个新的数据结构，元组（tuple）,一个元组是类似于一个不能修改其中内容的数组。他的格式类似如下：

first\_word = ( 'verb ' , 'go ' )

以及学习如果利用先写测试代码，再反过来完成实际需要的项目的代码的编程方法。这一节有点奇怪，第一遍学习有点摸不着头脑。

## Exercise 49

第49节在48节的基础上，增加了一个语句解释器的功能，除了训练了更多的函数调用和面向对象语法之外，我不太明白这个解释器有任何的作用。对于许多语言而言，要解决用户输入文字的检测问题，可以用正则化表达这种方法，python里也有这种方法，使用match函数。

## Exercise 50

没有什么用的用Python写网页内容，在根据书中内容写好了app.py文件的代码后，进入存放该py文件的文件夹，运行以下代码，

(lpthw) $ python3.6 app.py

\* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)

就可以在浏览器<http://localhost:5000/> 地址下看到网页的内容了。

当我们的电脑使用localhost和网络端口5000来打开网页后，它会去给app.py应用程序发出一个HTTP请求（request）要求

1. 首先，我们导入了 Flask 类。这个类的实例将会是我们的 WSGI （Web服务器网关接口，Python Web Server Gateway Interface，缩写为WSGI）应用程序。

2. 接下来，我们创建一个该类的实例，第一个参数是应用模块或者包的名称。 如果你使用单一的模块（如本例），你应该使用\_\_name\_\_ ，因为模块的名称将会因其作为单独应用启动还是作为模块导入而有不同（ 也即是'\_\_main\_\_' 或实际的导入名）。这是必须的，这样 Flask 才知道到哪去找模板、静态文件等等。详情见Flask的文档。

3. 然后，我们使用route() 装饰器告诉 Flask 什么样的 URL 能触发我们的函数。（装饰器可以参考廖雪峰老师的讲解：装饰器）

由于Python的flask这个框架，实在是不想学习前端的东西，所以这三节略过