

菜鸟学Python 三周快速入门实战项目集

菜鸟学Python 三周快速入门实战项目集

一,环境安装

- 1).Sublime Text:发烧友级
- 2).Pycharm:专业级
- 3).数据挖掘利器
- 4).总结

二. 3周实战入门

- 1.学会Python可以干很多事情
- 2,三周入门大纲,脑图分析

三,实战开始

1.字符串趣味实战

- 1-1,题目:替换1-20内的数字, 3的倍数和5的倍数用不同的数字代替
- 1-2,代码思路:

2.列表,字典-综合实战应用

- 2-1,题目:寻找班级里面名字最长的人
- 2-2,代码思路:

3.基础数据实战

- 3-1,题目:九宫格用Python-14行代码搞定
- 3-2,代码思路
- 3-3,题目:猜数字游戏
- 3-4,代码思路:

4.函数练习

- 4-1,题目:用Python画一朵漂亮的花
- 4-2,代码思路:

5.融合多个技巧的文件练习

- 5-1,题目:搜索文件夹, 并统计里面的文件
- 5-2,代码思路:

6.异常处理

- 6-1,题目:迷你计算器
- 6-2 代码思路:

7.常见的文件格式处理

- 7-1 题目:把一份股票的csv数据转换写入excel文件中
- 7-2,代码思路

8.时间处理

- 8-1,题目:写一个生日提醒小程序
- 8-2 代码思路:

9 并发处理

- 9-1,题目:有道并发爬虫
- 9-2,代码思路:

一,环境安装

Python就像一本武林秘籍，想要修炼起来，一定要找一个顺手的兵器，可以说挑选一个好的开发工具是极其重要的，一个好的IDE会帮助你方便地编写Python程序，使你的编程更加舒适。

我用python已经有好几年了，中间陆陆续续用过一些IDE，我觉得有2款神器是非常适合初学者的，用了之后感觉就像行云流水，爱不释手，写起代码来非常的爽~~ 推荐给大家，不好勿喷

一个好的开发工具应该是德才兼备:长的要漂亮，功能要强大，使用要简单，下面这2个都是这样的爆款

1).Sublime Text:发烧友级

Sublime 一个字就是炫，非常的酷，用了之后视觉效果上非常享受的。现在已经到Sublime Text3了.而且现在是非常流行的编辑器，基本上前端开发HTML、CSS、JS都用它。

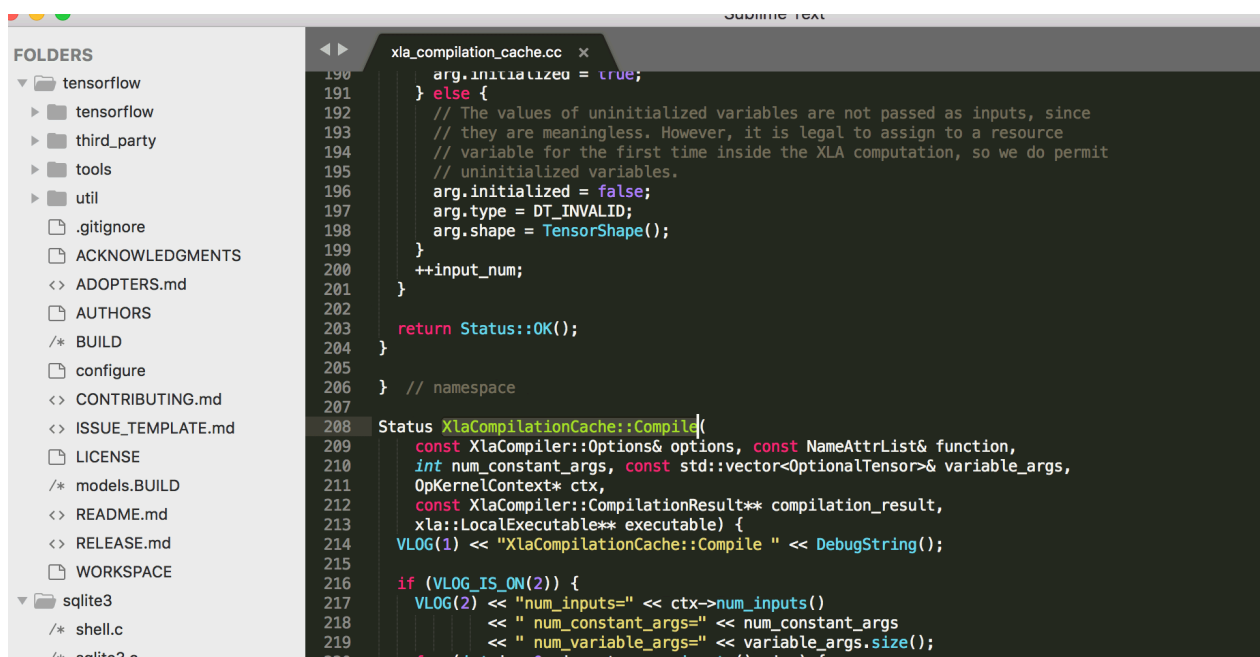
第一：它非常简洁

第二：有庞大的插件库，第三字体配色实在是太好看了(我用了这么多工具，众里寻他千百度，蓦然回首发现还是它的字体配色最好看)，一般来说python 开发必装的有:Package Control,Emmet,PyV8,SublimeREPL(终端的交互调试)，SublimeCodeIntel(代码的提示)

第三：免费的，而且启动非常快

下载地址:

<http://www.sublimetext.com/>




2).Pycharm:专业级

如果说只能推荐一款python IDE,那么非Pycharm莫属,因为它的功能非常强大,

而且很多功能(调试、语法高亮、Project管理、代码跳转、智能提示、自动完成、单元测试、版本控制)这些已经都已经内置了,不用另外安装插件。最爽的是它的代码跳转非常实用,当你写一个大几千行的项目,里面的类,函数很多的时候,就需要它方便的跳转。

而且这款神器结合了 Flask,Django,H5,AngularJS 都包含了。这样对于web开发就不要再安装其他的IDE了。

另外还有一个非常重要的功能就是支持代码重构,相信资深码农都知道代码重构的重要性,我就不多说了。



Download PyCharm

[Windows](#) [macOS](#) [Linux](#)

Professional

Version: 2018.3.4
Build: 183.5429.31
Released: January 30, 2019

Full-featured IDE for Python & Web development

[System requirements](#)
[Installation Instructions](#)
[Previous versions](#)

[DOWNLOAD](#)

Free trial

Community

Lightweight IDE for Python & Scientific development

[DOWNLOAD](#)

Free, open-source

下载链接:<http://www.jetbrains.com/pycharm/>

3).数据挖掘利器

Python中还有一个值得推荐的开发工具就是Anaconda,这一款是航母级别的开发工具,内置了很多数据开发的软件,其中鼎鼎大名的就是jupyter-notebook,当然除了这外还有spyder 和IPython。而且几乎内置了Python web开发和数据分析主流的所有的库,如果你用上面两个工具安装的话,需要一一安装,非常麻烦!

The Enterprise Data Science Platform for...



Data Scientists

Connect to a range of sources, collaborate with other users, and deploy projects with the single click of a button [Learn More >](#)



IT Professionals

Safely scale and deploy from individual laptops to collaborative teams, from a single server to thousands of nodes [Learn More >](#)



Business Leaders

Harness the power of data science, machine learning, and AI at the pace demanded by today's digital interactions [Learn More >](#)

下载链接: <https://www.anaconda.com/>

4).总结

这么多种选择对于初学者来说，应该怎么选择呢，建议下载anaconda因为它的库最全，而且包很多！然后开发的时候可以用Pycharm来关联anaconda里面的Python解释器，这样最方便。

菜鸟学Python

二. 3周实战入门

1.学会Python可以干很多事情

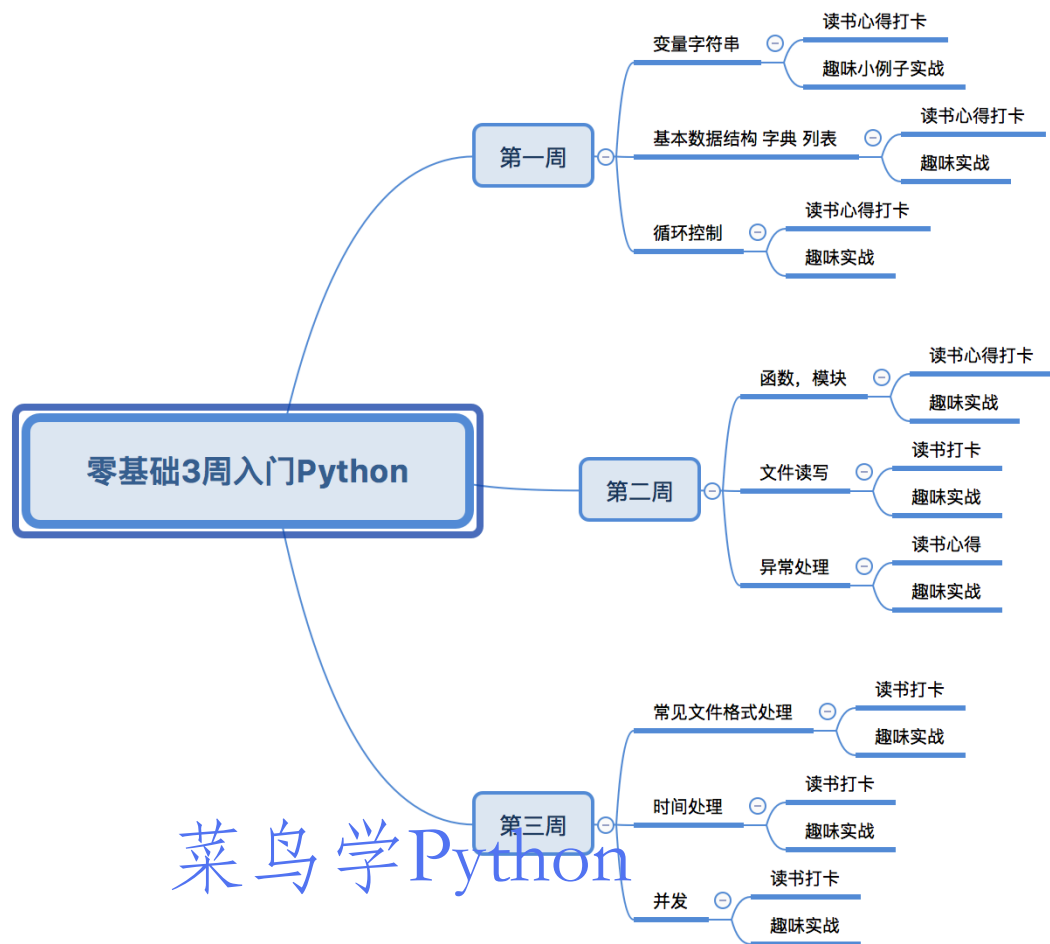
Python语法非常简单，功能强大。学好了可以做很多事情：

- 比如运维测试，目前测试岗位尤其是自动化测试或者运维岗位，Python语言已经必备技能。
- 比如数据分析，常见的数据分析岗位一共都是excel是基础，但是对于大量的数据，或者复杂的数据用Python语言可以轻松处理，并且可视化。
- 比如后端开发，Python语言里面有3大赫赫有名的web框架，其中以Flask最为有名，轻量级的神器，搭建一个小型的网站非常快速，便捷
- 比如数据挖掘，机器学习，Python语言里面最火的岗位非机器学习，数据挖掘莫属。目前人工智能相关的岗位薪资非常高，而且有越来越火的趋势，而Python是最佳的机器学习语言。

可以说很少有一门语言想Python这样简单易上手，同时有功能这么强大，学会一门可以干很多工种和胜任不同的岗位，即使你有C++/Java的基础，学会Python对你找工作也是很好的一个加分项目。

2,三周入门大纲,脑图分析

三周的时间说长不长，说短很短，如何快速的入门，到时要掌握哪些知识点呢，我把我们这次的实战训练营安排列了一张脑图，帮助大家快速总览一下：



三,实战开始

1.字符串趣味实战

1-1,题目:替换1-20内的数字, 3的倍数和5的倍数用不同的数字代替

列出1到20的数字, 若是3的倍数就用apple代替, 若是5的倍数就用orange代替, 若既是3的倍数又是5的倍数就用appleorange代替。

1-2,代码思路:

第一种普通解法,循环1-20, 然后用多个if/else进行判断:

```
def replace_num(i):
    if i%3==0 and i%5==0:#放第一个
        return "appleorange"
    if i%3==0:
        return 'apple'
    if i%5==0:
        return 'orange'
    else:
        return i
```

第二种牛逼解法:巧妙的利用列表切片

```
'apple'[i%3*len('apple')::]+'orange'[i%5*len('orange')::]or i
```

先说一下 这道题的这种解法 犹如天外有天，令人拍案叫绝！

慢动作分解1:

菜鸟学Python

```
print ('apple'[1::])
>>'pple' 表示从第2个位置开始切片
```

慢动作分解2:

既然明白了上面的，来一个稍微复杂一点的

```
>>print ('apple'[1*5::])
```

为空

为啥因为1*5是5,也就是要从第6个字符开始，apple一共为5个字符，所以输出为空

慢动作分解3:

```
for i in range(1,10):
    print ('apple'[i%3::])

>>
pple
ple
```

```
apple
pple
ple
apple
pple
ple
apple
```

会发现只有3的倍数的地方会出现完整的apple，其他地方都是残缺的，但是我们怎么把非3的倍数的地方变成空呢，简单乘以一个偏移量，这招对3的倍数没有任何影响，但是对于其他的非3的倍数有很大的影响

慢动作分解4:

```
for i in range(1,10):
    print ('apple'[i%3*len('apple')::])
```

```
>>
apple

apple

apple
```

菜鸟学Python

这个时候虽然把非3的过滤掉了，但是我们要输出数字啊，怎么办，这里又用了是一个非常巧妙的or for i in range(1,10): print ('apple'[i%3*len('apple')::] or i)

这道题是的技巧性非常高，而且很巧妙，不知道零基础的同学有没有看明白！

2.列表,字典-综合实战应用

2-1,题目:寻找班级里面名字最长的人

我有一串字符串人名:

```
names=(' Kunpen Ji, Li XIAO, Caron Li, Donl SHI, Ji ZHAO,Fia YUAN Y, Weue
DING, Xiu XU, Haiying WANG, Hai LIN,Jey JIANG, Joson WANG E, Aiyang ZHANG,Hay
MENG, Jak ZHANG E, Chang Zhang, Coro ZHANG')
```

我希望能做到下面3点:

问题1: 排序,按照名字A-Z排序

问题2: 找出里面姓"ZHANG"有几个

问题3: 找出名字里面最长的人

2-2,代码思路:

分析问题1:

首先我们要做的对字符串进行分割去掉',',然后就变成了一个长的列表, 然后对列表进行排序(注意名字前后有多余空格要去掉), 第一个问题就解决了.

```
def sort_names(names):  
    return (sorted([name.strip() for name in names.split(',')]))
```

分析问题2:

接下来, 我们需要找出姓"ZHANG", 因为名字里面有英文名字和中文名字, 有的后面还跟E/Y, 所以我们先定义一个函数, 把分割后的新的名字列表, 取出每一个名字, 然后解析, 翻转, 用推导列表形成一个新的名字列表, 然后再用字符串里的.startswith("ZHANG")取出符合的名字

```
names = ('Kunpeng Ji, Li XIAO, Caron Li, '
         'Dongjian SHI, Ji ZHAO, Fia YUAN Y, '
         'Wenxue DING, Xiu XU, Haiying WANG, Hai LIN, '
         'Jey JIANG, Joson WANG E, '
         'Aiyang ZHANG, Haiying MENG, '
         'Jack ZHANG E, Chang Zhang, Coron ZHANG')

def sort_names(names):  
    return (sorted([name.strip() for name in names.split(',')]))

def get_chinese_names(names):  
    chinese_names=[]  
    for name in sort_names(names):  
        if len(name)>=2:  
            first_name=name.split()[0].capitalize()  
            last_name=name.split()[1].capitalize()  
            chinese_names.append(last_name+' '+first_name)  
        else:  
            chinese_names.append(name)  
  
    return chinese_names

def find_name(names, key='Zhang'):  
    return [name for name in names if name.startswith(key)]
```

3.基础数据实战

3-1,题目:九宫格用Python-14行代码搞定

九宫格游戏对人们的思维锻炼有着极大的作用，从古时起人们便意识到九宫的教育意义。千百年来影响巨大，在文学、影视中都曾出现过。九宫格最早叫“洛书”，现在也叫“幻方”。

要求很简单：

1至9九个数字，横竖都有3个格，思考怎么使每行、每列两个对角线上的三数之和都等于15



算法:

九宫图的算法有很多种，我们今天讲的是最原始的算法，算是暴力破解法

- 九宫格是三行三列，每一行都是1-9中的3个数字,我们先获取1-9所有的3个数字全排列组合 [S1,S2..Sn]，差不多有 $9 \times 8 \times 7 = 504$ 种
- 这样的话3行,其实就是每一行从这个504序列里面取一个放到第一排，第二排，第三排，就形成一个矩阵
- 对这个3*3矩阵,只要判断行，列，对角线和斜对角线都是15就可以了

3-2,代码思路

step1.获取1-9数字的全排列

- Python标准库中有一个赫赫有名的模块叫做itertools

- 这模块提供了很多操作迭代对象的函数,非常方便
- 这次我们用itertools里面的permutations, 它可以方便的全排列序列中的数字,每一个组合都是3个数字:比如S1(1,2,3),S2(1,5,8)..这样我们得到一个长的列表[S1,S2...Sn]

step2:列出3*3的矩阵

- 每一行都是48个中选1个, 那么3行最大的搜索空间就是 $48 \times 48 \times 48 = 110592$
- 让电脑运算十几万搜索空间只需要几秒钟, 我们先列出110592个3*3的矩阵
- 怎么做呢, 很简单, 3个for循环搞定:

step3.计算行,列,对角线和斜对角线都是15

上面我们已经把这3*3的矩阵列出来了:

row1_1,row1_2,row1_3

row2_1,row2_2,row2_3

row3_1,row3_2,row3_3

只要判断行, 列, 对角线和斜对角线都是15就可以了

step4:过滤重复矩阵

菜鸟学Python

这样我们就可以得到'行, 列, 对角线和斜对角线都是15'的矩阵, 但是这里面会有重复的元素,比如下面这样的:

1 9 5

9 5 1

5 1 9

比如第一行是(1,5,9),第二行是(9,5,1)这样的怎么过滤呢, 简单我们用集合

把第一行和第二行都放到集合里面, 主要判断他们的交集长度为0, 则表示他们没有交集就可以了

if len(set(row1)&set(row2))==0:

```
import itertools
def jiuGongge():
    #step1
    #穷举3个数字的组合
    nums=[x for x in range(1,10)]
    seq_3nums=[p for p in itertools.permutations(nums,3)]

    #过滤出和等于15
    seq_3nums=[p for p in itertools.permutations(nums,3) if sum(p)==15]
    # print (seq_3nums)

    #step2:搜索行, 列, 对角, 斜对角, 中线均为15
    matrix=[]
    for row1_1,row1_2,row1_3 in seq_3nums:
```

```

for row2_1,row2_2,row2_3 in seq_3nums:
    for row3_1,row3_2,row3_3 in seq_3nums:
        if row1_1+row1_2+row1_3==15 \
            and row1_1+row2_2+row3_3==15 \
            and row1_1+row2_1+row3_1==15 \
            and row1_3+row2_2+row3_1==15 \
            and row1_2+row2_2+row3_2==15:

            #step3 去重
            row1=[row1_1,row1_2,row1_3]
            row2=[row2_1,row2_2,row2_3]
            row3=[row3_1, row3_2, row3_3]
            if len(set(row1) & set(row2)) == 0:
                if row1 not in matrix:
                    matrix = [row1, row2, row3]
                    print (matrix)
            print ('-'*100)

```

3-3,题目:猜数字游戏

1.让用户输入1-20, 猜数字, 可以猜5次。 2.每次有提示, 大了, 或者小了! 3.如果超过5次, 提示 game over.

3-4,代码思路:

- 1.用input提示用户输入一个数字,然后随机产生一个1-20的整数
- 2.因为最多循环5次, 所以需要用while来循环, 在循环体的内容, 来对输入的数字进行判断
- 3.分别对数字大了, 小了和正好, 三种情况进行判断, 并且retry的计数器进行加1

```

def guess_num():
    max_retry=5
    i=0
    random_num=random.randint(1,21)
    while i<max_retry:
        try:
            num=int(input("please input num :1-20\n"))
            print (f'Your input is :{num}')
            if num>random_num:
                print ('>>Biger')
            elif num<random_num:
                print ('>>Small')
            else:
                print ('!!Great,you guess the num!')

```

```
        break

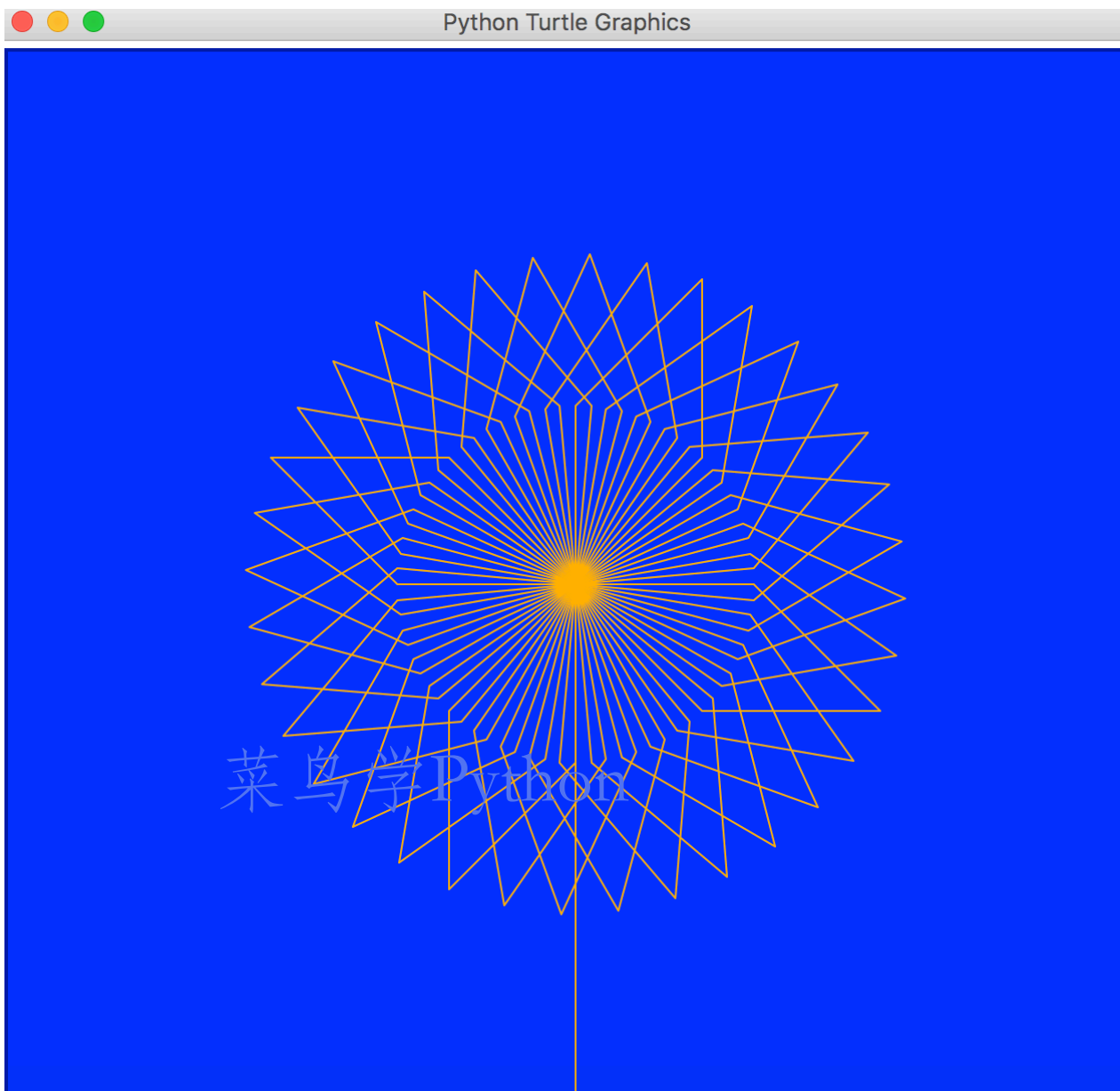
    except Exception as e:
        print (f'Your input incorrect,Please input num 1-20\n')
    finally:
        i+=1
        print (f'retry time:{max_retry-i}')
else:
    print ('Your retry time>5,Game over!')
```

4.函数练习

4-1,题目:用Python画一朵漂亮的花

关于函数和模块学起来太枯燥了，我一直想用一个**好玩有趣的小例子**趣味学习，同时也作为实战练习一下。找来找去，决定要找一个**好玩有趣的例子**来总结，总于被我找到了，这个例子**只有20几行代码**,非常适合初学者

先看一下效果:



4-2,代码思路:

1.引入模块

Python标准库里有一个非常有趣可以画画的小模块叫turtle(海龟),我们先引入这个模块

```
import turtle #从标准库里面引入turtle

def draw_art():

    window=turtle.Screen()#获得一个窗口句柄

    window.bgcolor("blue")#把背景设为蓝色

    window.exitonclick()#当点击一下窗口会自动关闭

draw_art()#调用函数
```

2.画一个小海龟出来

第一步我们已经把窗口创建好了，你可以认为是一个画布已经ok了，下面我们要让主角登场了，把海龟画出来.

```
import turtle
def draw_art():
    window=turtle.Screen()
    window.bgcolor("blue")

    #创建一个Turtle的实例这里用了类的概念，实例化一个Turtle。类的概念后面文章会讲

    brad=turtle.Turtle()
    brad.shape("turtle")#形状是一个海龟除了画海龟还可以画箭头，圆圈等等
    brad.color("orange")#颜色是橙色
    brad.speed('fast')#画的速度是快速

    window.exitonclick()#当点击一下窗口会自动关闭

draw_art()#调用函数
```

3.画一个海龟走两步的图

我们让海龟在图上走**100步**，然后再往下**45度**走100步

```
import turtle
def draw_art():
    window=turtle.Screen()
    window.bgcolor("blue")

    brad=turtle.Turtle() #创建一个Turtle的实例
    brad.shape("turtle")#形状是一个海龟除了画海龟还可以画箭头，圆圈等等
    brad.color("orange")#颜色是橙色
    brad.speed('fast')#画的速度是快速
    brad.forward(100)#向前走100步
    brad.right(45)#然后海龟头向右转45度
```

```
brad.forward(100)#继续向前走100步
brad.right(135)#然后有向右转135度

window.exitonclick()

draw_art()#调用函数
```

4.画出一个菱形

刚才我们已经画来2边，一个完整的菱形还差2边，所以我们只需要把刚才走的路循环一下，把刚才的几步抽象成一个小函数(把代码抽取成一个独立的函数，是重构经常用的技巧)，然后循环2次，就搞定了。

```
import turtle
def draw_diamond(turtle):
    for i in range(1,3):
        turtle.forward(100) #向前走100步
        turtle.right(45) #然后海龟头向右转45度
        turtle.forward(100) #继续向前走100步
        turtle.right(135) #然后有向右转135度

def draw_art():
    window=turtle.Screen()
    window.bgcolor("blue")
    brad=turtle.Turtle() #创建一个Turtle的实例
    brad.shape("turtle")#形状是一个海龟除了画海龟还可以画箭头，圆圈等等
    brad.color("orange")#颜色是橙色
    brad.speed('fast')#画的速度是快速

    draw_diamond(brad)#抽象一个新的函数，专门画菱形
    window.exitonclick()

draw_art()#调用函数
```

5.画出一朵漂亮的花

前面的4步我们已经可以画出一个菱形了，其实这个菱形是我们要画一个花瓣，接着我们主要把菱形向右旋转10度，然后继续画一个花瓣出来，这样循环一周360度，就花出一个漂亮的花了，最后当海龟回到花心的时候，我们把海龟的头向右转90度，花一根长的线就大功告成了

5.融合多个技巧的文件练习

5-1,题目:搜索文件夹，并统计里面的文件

案例:

搜索文件夹，并统计里面的文件

假如我们有一个目录里面包含若干个文件和子目录：

问题1：我们要统计该目录下有多少个文件并显示出来（包含子目录）

问题2：该目录总共的大小可以按M，也可以按K显示

问题3：该目录下最大的文件和最小的文件，以及对应的大小

目录为你的python，环境目录：

比如我的目录是：/bin/python

5-2,代码思路:

程序其实就是**数据结构+算法**，所以我们先确定自己的数据结构，还有算法

1).因为要列出**最大/最小**的文件名字和大小，数据结构我们用字典

2).**算法**的话，先把目录下的全部文件和目录列出

- 若是文件就统计大小
- 若是子目录，就继续寻找该目录下的子文件，然后不断重复刚才的过程,因为我们不知道有多少层套嵌的子目录，最好用递归

3).最后就是**显示**，要按MB,KB显示，需要我们定义一个扩展的函数入参结构，用默认位置参数

提示:当然也直接可以用**os.walk()**函数，直接帮我们递归所以目录含子目录里的文件

```
import os
def get_file_size(file):
    #判断文件是否存在，存在则取文件大小
    if os.path.exists(file):
        return os.path.getsize(file)
    else:
        return 0

def statist_files(path):
    files=[]
    #遍历路径下的所有文件,包括子目录
    for dirpath,dirnames,filenames in os.walk(path):
        for file_name in filenames:
            files.append(os.path.join(dirpath,file_name))

    #用推导列表，获取每个文件和文件大小
    return [{'name':f,
            'size':get_file_size(f)} for f in files]

def count_files_size(files_info,size_display_mode='M'):
    #计算文件大小，按照K,M,G,Byte分别计算
```



```

sizes=[file_info.get('size',0) for file_info in files_info]

if size_display_mode=='K' or size_display_mode=='k':
    return str(round(sum(sizes)/1024,3))+ 'K'
elif size_display_mode=='M' or size_display_mode=='m':
    return str(round(sum(sizes) / (1024*1024), 3)) + 'M'
elif size_display_mode=='G' or size_display_mode=='g':
    return str(round(sum(sizes) / (1024 * 1024 *1024), 3)) + 'G'
else:
    return str(round(sum(sizes)))+ 'Byte'

def get_sorted_files(files):
    #按照文件的大小排序
    sorted_files=sorted(files,key=lambda x:x['size'],reverse=True)
    return sorted_files

```

6.异常处理 菜鸟学Python

在Python程序中，分别使用未定义变量、访问列表不存在的索引、访问字典不存在的关键字观察系统提示的错误信息

```

# 使用未定义变量
a + 1

# 访问列表不存在的索引
b = ['a', 'b', 'c']
b[3]

# 访问字典不存在的key
c = {'x': 1}
c['y']

# 2. 通过Python程序产生IndexError，并用try捕获异常处理
try:
    b = ['a', 'b', 'c']
    b[3]
except IndexError:
    print('访问列表不存在的索引')

```

6-1,题目:迷你计算器

编写一个迷你的计算器，支持两个数加，减，乘，除 要求提示用户输入2个数字和一个运算符号，比如 1,2,+ 提示： 这个题目里面需要有几个地方检查 第一:是输入的参数，用户可能会乱输入，这个地方要有判断 第二:输入的参数要合法运算的适合，要考虑异常，比如9/0这样的肯定不对 第三:输入的参数，尤其是数字，可能是浮点，比如1.1,-10,-0.09

6-2 代码思路：

这个小程序虽然很小，但是需要考虑的东西还是很多的

- 第一个是输入的参数，用户可能会乱输入，我们要去输入2个数字，加一个运算符而且数字有正数和负数，而且还有小数
- 第二个是数字运算，要考虑到一些不合逻辑的运算，比如10/0

```
import re
def verify_num(n):
    #用正则判断几种数字,正数, 负数, 小数点数字
    patt=re.compile(r'^(-?\d+)(\.\d+)?$')
    return True if re.match(patt,n) else False

def verify_opt(opt):
    #判断运算符
    return opt in ['+', '-', '*', '/']

def mini_calculator():
    #用户输入数据
    my_input=input('Please two nums and operation,such as 1,2,+: \n')
    try:
        args=my_input.split(',')
        if len(args)==3:
            a,b,opt=args
            if not verify_num(str(a)) or not verify_num(str(b)) or not verify_opt(opt):
                print ('Input format is incorrect!')
                return
            #拼接表达式, 然后用eval进行求表达式值
            res=eval('{}{}{}'.format(a,opt,b))
            print ('{}{}{}={}'.format(a,opt,b,res))
        else:
            print ('Args len should be 3!')
    except ValueError as e:
        print ('Value error',e)
    except Exception as e:
        print (e)
```

7.常见的文件格式处理

7-1 题目:把一份股票的csv数据转换写入excel文件中

题目： 1.拿到平安银行一年的股票数据 csv文件

平安银行

	ts_code	trade_date	open	high	low	close	pre_close	change	pct_chg	vol	amount
0	000001.SZ	20181228	9.31	9.46	9.31	9.38	9.28	0.1	1.0776	576604	541571.004
1	000001.SZ	20181227	9.45	9.49	9.28	9.28	9.3	-0.02	-0.2151	624593.27	586343.755
2	000001.SZ	20181226	9.35	9.42	9.27	9.3	9.34	-0.04	-0.4283	421140.6	393215.14
3	000001.SZ	20181225	9.29	9.43	9.21	9.34	9.42	-0.08	-0.8493	586615.45	545235.607
4	000001.SZ	20181224	9.4	9.45	9.31	9.42	9.45	-0.03	-0.3175	509117.67	477186.904
5	000001.SZ	20181221	9.68	9.7	9.33	9.45	9.71	-0.26	-2.6777	1000616.76	944460.332
6	000001.SZ	20181220	9.92	9.97	9.63	9.71	9.94	-0.23	-2.3139	990284.79	964202.861
7	000001.SZ	20181219	10.14	10.18	9.9	9.94	10.12	-0.18	-1.7787	598007.01	600090.171
8	000001.SZ	20181218	10.2	10.32	10.1	10.12	10.29	-0.17	-1.6521	537744.3	547157.562
9	000001.SZ	20181217	10.16	10.33	10.1	10.29	10.17	0.12	1.1799	571274.87	584679.512
10	000001.SZ	20181214	10.34	10.35	10.16	10.17	10.39	-0.22	-2.1174	526675.13	539263.012

2.里面一共244个交易日，我们读取csv文件，然后找到成交量(amount)大于1百万手的交易日的数据 3.然后把大于1百万手的那天的如下数据: open high low close amount 写入excel文件

菜鸟学Python

7-2,代码思路

常见的文件格式的读取，Python都有库支持，比如csv文件有csv库；json文件有json库；操作excel的库xlrd和xlwt

下面说说思路：

- 1).利用csv库把csv文件读取并进行数据处理，然后设计一个数据结构
- 2).这里用3种不同的数据结构，一个是普通的字典带缺省，一个是顺序字典，一个迷你类namedtuple
- 3).把过滤出来的数据存入excel表中，分3步：先是创建一个 excel,然后先写表头，再一行一行写表的内容，最后保存excel

```
from collections import namedtuple, defaultdict, OrderedDict
import csv
import xlwt

headers=[ 'ts_code', 'trade_date', 'open', 'high', 'low', 'close', 'amount' ]

def read_csv(file,max_amount=1000000):
    res = []
    stock = namedtuple('stock',headers)

    with open(file, 'rt') as rf:
        csv_reader=csv.reader(rf)
        for each_line in csv_reader:
```

```

        if csv_reader.line_num==1:
            continue
        amount=float(each_line[-1])
        if amount>max_amount:
            stock.ts_code=each_line[1]
            stock.trade_date=each_line[2]
            stock.amount=each_line[-1]
            res.append(stock)
    return res

#write data into excel file
def write_data_to_excel(data=[]):
    excel=xlwt.Workbook()
    worksheet = excel.add_sheet('stock')
    #write header
    for index,header in enumerate(headers):
        worksheet.write(0, index, label=header)

    #write data
    for index_row,each_dict in enumerate(data):
        for index_col,v in enumerate(each_dict.values()):
            worksheet.write(index_row+1,index_col,v)

    excel.save('Stock.xls')

```

上面只是其中一种解法，这里是一个非常典型的字典应用场景，我们也可以namedtuple来构造一个迷人的类来处理，会更简洁：

```

from collections import namedtuple,defaultdict,OrderedDict

headers=['ts_code','trade_date', 'open', 'high','low','close','amount']
def read_csv(file,max_amount=1000000):
    res = []
    stock = namedtuple('stock',headers)

    with open(file, 'rt') as rf:
        csv_reader=csv.reader(rf)
        for each_line in csv_reader:
            if csv_reader.line_num==1:
                continue
            amount=float(each_line[-1])
            if amount>max_amount:
                stock.ts_code=each_line[1]
                stock.trade_date=each_line[2]
                stock.amount=each_line[-1]
                res.append(stock)

```

```
return res
```

当然可以利用一个第三方的库openpyxl,这个库使用非常灵活而提供了很多对excel精细话的设置，但是只支持xlsx格式的excel表格，对于xls不支持。

8.时间处理

8-1,题目:写一个生日提醒小程序

时间和日期的综合小练习 1.计算你的生日比如近20年来(1990-2019)，每年的生日是星期几,统计一下星期几出现的次数比较多 2,生日提醒,距离生日还有几天 3.计算你和你的老婆或者女友的生日差多少天 如果生日不愿意泄露，可以随机写一个比如1990-1-1

8-2 代码思路：

```
def get_birthday_weekday(birthday_str):
    weekday=datetime.strptime(birthday_str,'%Y-%m-%d').weekday()
    weekdays=['Mon','Tue','Wed','Thur','Fri','Sat','Sun']
    return weekdays[weekday]

def internal_days(name,your_birthday):
    try:
        birthday=datetime.strptime(your_birthday,'%Y-%m-%d')
        delta_days=(birthday-datetime.today()).days
        if delta_days>0:
            return (f'距离{name}的生日还有 {delta_days} 天')
        elif delta_days<0:
            return (f'距离{name}的生日已过 {abs(delta_days)} 天')
        else:
            return (f'今天是{name}的生日，生日快乐!')

    except ValueError as e:
        print ('Please input the birthday format as :1991-2-3')

def statist_birthday():
    birthday_list=[str(y)+'-1-1' for y in range(1990,2020)]
    res=[]
    for b in birthday_list:
        res.append(get_birthday_weekday(b))
    print (Counter(res))
```

时期的处理，能熟练掌握time,datetime和calendar这3个库的使用就可以了。

9 并发处理

1).Python里面的异步主要有3种方式实现，一种是多进程，一种是多线程，还有多协程。3种各有千秋，而且有对应的库：

- 多进程的库有 `from multiprocessing import Process,Pool`
- 多线程的库有 `from threading import thread`

常见的并发的任务分 cpu密集型 and IO密集型两种方式，如果对于CPU密集型任务用多线程是没有用的，因为Python因为有GIL全局解释器锁的问题，所以对于CPU密集型必须用进程，利用多核cpu来处理。

而对于 I/O密集型的任务，可以用多线程来处理，比如典型的网络请求业务，因为服务端发起网络请求，会需要等待，这里就会有等待时间，可以用多线程来提高效率。

2).Python还帮我们提供了集成好的多线程池和多线程池，使用起来更方便

多进程池 `from concurrent.futures import ProcessPoolExecutor`

多线程池 `from concurrent.futures import ThreadPoolExecutor`

3).对于协程，python3.6做了很大的改进，加入了最具野心的异步框架asyncio, 这个库的功能非常强大，不过入门阶段建议大家先不要碰，因为比较难理解。

菜鸟学Python

9-1,题目:有道并发爬虫

用并发写一个小程序，获取有道翻译上的单词的意思，比如 单词china 爬取 url:<http://dict.youdao.com/w/eng/china/> 输出的数据结构： 单词的名字 单词的发音 单词的内容 例如: {'Word': 'china', 'Proc': '英 [ˈtʃaɪnə] 美 [ˈtʃaɪ.nə]', 'Desc': 'n. 瓷器adj. 瓷制的n. (China) 中国adj. (China) 中国的'}

那么如果我有20个单词列表，如何并发去爬取 ['china']*20

9-2,代码思路：

大家可以多线程，也可以用线程池去完成(ThreadPoolExecutor) 有道的单词的url: 只要填充多个单词即可 `'http://dict.youdao.com/w/eng/{}/'.format('xxxx')`

代码思路： 1).先写一个函数去下载<http://dict.youdao.com/w/eng/china/> 2).然后解析这个页面，解析可以用pyquery,这个库非常好用，大概只要几行代码就可以解析 3).然后用多线程去处理上面的task

```
from requests import ConnectTimeout
from pyquery import PyQuery as pq
from concurrent.futures import ThreadPoolExecutor
import requests

COUNT = 1

def parse_html(doc):
```

```

proc_text = ''
desc_text = ''

for pro in doc.items('.baav .pronounce'):
    proc_text += pro.text()

for li in doc.items('#phrsListTab .trans-container ul li'):
    desc_text += li.text()

return {'Proc': proc_text, 'Desc': desc_text}

def translate_word(word):
    global COUNT
    output = {}
    url = 'http://dict.youdao.com/w/eng/{}/'.format(word)
    print('{} Fetch:{}'.format(COUNT, url))
    COUNT += 1
    headers = {
        'Cookie': 'DICT_UGC=be3af0da19b5c5e6aa4e17bd8d90b28a|;
webDict_HdAD=%7B%22req%22%3A%22http%3A//dict.youdao.com%22%2C%22width%22%3A960
%2C%22height%22%3A240%2C%22showtime%22%3A5000%2C%22fadetime%22%3A500%2C%22notS
howInterval%22%3A3%2C%22notShowInDays%22%3Afalse%2C%22lastShowDate%22%3A%22Mon
%20Nov%2008%202010%22%7D; __rl__test__cookies=1515809612366;
_ntes_nnid=7c7071e6ff37a1f321a28ebd3450279f,1503241340929;
OUTFOX_SEARCH_USER_ID_NCOO=1630604769.092356;
OUTFOX_SEARCH_USER_ID=-224682605@10.169.0.76;
JSESSIONID=abcr90lYVoIM8dOeXsTdw',
        'Host': 'dict.youdao.com',
        'Upgrade-Insecure-Requests': '1',
        'User-Agent': 'Mozilla/5.0 (Macintosh; Intel Mac OS X 10_12_6)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/63.0.3239.132 Safari/537.36'
    }
    try:
        r = requests.get(url, headers=headers)
        if r.status_code == 200:
            doc = pq(r.text)
            info = parse_html(doc)
            output['Word'] = word
            output = dict(output, **info)

    except ConnectTimeout as e:
        print('Get url:{} timeout'.format(url))
    except ConnectionError as e:
        print('Get url:{} error'.format(url))
    except Exception as e:
        print(e)

    return output

```

```
def async_task(words):  
    pool = ThreadPoolExecutor(4)  
    threads = [pool.submit(translate_word, (word)) for word in words]  
    for t in threads:  
        print(t.result())  
  
words = ['china', 'english', 'temperaments']  
async_task(words)
```

这道并发的入门练习，我们用多线程来实现。大家掌握了多线程之后，可以改成多进程，二者的使用方法非常类似。

菜鸟学Python