**1.Python从零单排简介**

　　第一次接触python是前年，那次偶遇并没有发生什么有趣的故事。后来越发觉得计算机语言对于IT从事着来说是必不可少的技能，语言都不会可不敢妄称自己是IT行业从事者，所以后来琢磨着要学一门语言，学什么语言纠结了很久，恰好有一哥们是java攻城狮，就跟他取经了，学了一段时间后放弃了，经历了一段从入门到放弃的旅程。  
　　年初注意到目前python是很火的，原因很多，如简洁易懂，比之其他语言，python很优美，而且现在用的也很广泛，像大数据分析、AI、爬虫等等都在用python，不论是从本身的测试职业出发还是未来的前景都很不错，于是自学了python，跟学一门外语其实是一样的，语法、方法等都是规定好的，都是套路，关键在于如何理解才能更快的掌握这么语言，像英语单词，有很多攻略教我们怎么记单词，给我感觉用处最大的就是两点：  
　　1.联想记忆法，就是把单词联想成我们熟悉的事物，通过关联，我们就很容易记住这个单词了，下次再看到单词的时候，联想起熟悉的事物就很快想起来；  
　　2.场景记忆法，单词、语法都是一个个点的维度概念，几一起来比较困难，举个栗子：用3秒记住这15个数字：8，9，2，3，7，5，6，3，2，4，1，0，5，1，4；记住了没，怕是3秒过去都还没看完这些数字，但是这样呢：1，2，3，4，5，6，7，8，9，0，1，2，3，4，5，是不是一眼就记住了，因为这个场景我们太熟悉了，就是数字的顺序嘛。同理，python里的每个语法点都是一个个的点，用记住每个点太难了，但是把他们串起来，学起来就轻松很多。  
　　那么如何运用到我们的这个从零单排里面呢？第一点就是在理解python的时候尽量把她关联我们所熟悉的事物，方便理解；第二点，多练题，把所有知识点都给她连起来，会轻松很多。这些就是从零单排的中心点了，会围绕这两点开启从零单排的旅程，您系好安全带了么？

**2.认识Python**

Python是一种面向对象、解释型计算机语言，特点是语法简洁、优雅、简单易学。什么是解释型的计算机语言？顾名思义，这门语言是需要解释后计算器才看得懂。比如我钱包里多了一毛钱，媳妇就抓着我要解释，为啥会多了一毛钱，因为她并不晓得我再马路边捡了一毛钱，所以我要跟她解释，然后她接着会问：为啥不交给警察叔叔呢？那么我又要解释一下，因为我找不到人，这里他问一句我回一句，在python里一样，python说一句，编译器解释一句，知道最后执行完，这就叫解释型语言，其他类似的语言还有java，shell，ruby，php等；相对解释型还有一种编译型语言，就是说我把这一串疑问全部一次就回答了，不要她问，我钱包里多了一毛钱是因为我再马路边捡的，不交给警察叔叔是因为我没找到他们，不用一个个问题回答，直接一次性说完，这就是编译型语言，说白了就是我写好代码后，直接全部编译好，然后交给计算机，让它运行，没必要计算机看一行代码再解释一行，然后运行，直接将所有编译成计算机看得懂的二进制代码然后运行。也好比看一本英文书，python这种解释型语言是看一行，百度一行翻译成中文，我们才看的懂；编译型是直接全部给百度翻译好，直接看翻译版本就可以了。显而易见，编译型语言性能上会有很大优势。

　　在2014年的时候python官方宣布2.7支持到2020年，以后不会再发行2.8版本，尽快把程序迁移到3.x版本，所以用pythons3，工具用pycharm。

**3.变量**

　　变化的参数，我们先来看下常量，不变的参数，比如磁盘里的数据就是常量，不会变化，想用直接拿过来就好了，但是python跟其他语言一样，运行时在内存里运行的，大家都知道，内存是不会存数据的，都是一些临时数据，关机就没有了，所以python在运行的时候，定义一个变量，说白了就是，先跟内存说好我存一个参数放这里，到时用的时候我来这里拿，这就是变量。  
　　python里定义变量和其他语言又不一样，变量名=变量值，直接这样写就行，变量类型是根据你的值来定的，不需要申明变量类型。但是变量名也不能随意取，最好是用代表性的英文取名，能够见名思意，比如姓名用name等等，当然也有一些规则，变量名只能是英文或者下划线开始，可以包含英文、下划线、数字，且python里一些关键字也不能取名，这些已经被python占用，不能用来取变量名的python关键字有：  
　　['and', 'as', 'assert', 'break', 'class', 'continue', 'def', 'del', 'elif', 'else', 'except', 'exec', 'finally', 'for', 'from', 'global', 'if', 'import', 'in', 'is', 'lambda', 'not', 'or', 'pass', 'print', 'raise', 'return', 'try', 'while', 'with', 'yield']

　　常用变量如下：

1 #变量，变量名=变量值，不像其他语言，直接这样写就行，变量类型是根据你的值来定的，不需要申明变量类型

2 age=18 #输入一个数字就是整型，int类型

3 height=170.2 #输入一个小数就是浮点型，float类型

4 my\_list=[1,2,3] #定义一个列表

5 my\_tuple=(1,2,3) #定义一个元祖

6 my\_dict={1:'a',2:'b',3:'c'} #顶一个字典

7 my\_set=set('xiaogang') #定义集合

8 name1='进击的菜鸟' #只要用了引号，就是字符类型，单引号

9 name2="进击的菜鸟" #双引号

10 name3=''''进击的菜鸟''' #三引号

11 print(name1,age,height)#打印变量，逗号隔开打印的参数

12

13 #引号，上述所说，含有引号的都是字符串，但是当一个字符本身就有引号咋办，比如 I'm xiaogang，这样定义

14 str1="I'm xiaogang" #当字符串有单引号时，外面用双引号

15 str2='Xiaogang say:"Hello!"' #当字符串有双引号时，外面用单引号

16 str3='''Xiaogang say:"Hello!I'm xiaogang"''' #当字符串既有有双引号又有单引号时，外面用三引号

17

18 #三引号还有注释功能，前面和后面都加三引号

19 '''

20 print('打我啊！')#被注释掉了，运行结果没有这个

21 '''

　　上面的变量定义中，可以用type方法查看变量的类型，用法：print(type(变量名))，尝试这样看下，变量名是根据变量值的类型来的。

**4.格式化输出**

输出之前，先了解下输入，python3里直接用input()，就可以接收用户输入的参数，但是要注意一点，input接收到的用户输入参数类型全部是字符类型，代码如下：

name=input('请输入您的姓名：')

print(name)

格式化输出说白了就是设计好套路，在输出的时候按这个套路来输出，主要有以下四种：

　　用+号连接：这个严格意义上不算是格式化输出，是字符串方法，连接两个字符用的。

1 #格式化输出，就是设计好套路，在输出的时候按这个套路来输出,主要有四种

2 name='进击的菜鸟'

3 money=100

4 #用+号连接

5 print("Welcome to 'From zero single row':"+name)

6 #用,号连接

7 print("Welcome to 'From zero single row':",name)

8 #用百分号占位符,%s只是先占了一个位置在这里，然后再输出的时候把后面的值依次填充到占位的位置

9 print("Welcome to 'From zero single row':[%s]! Let's study!"%name)

10 #多个占位时，这样用，需要括弧括起来，必须是按照占位的顺序才可以

11 print("Welcome to 'From zero single row':[%s]! Your have money:[%s]元"%(name,money))

12 #拓展，%s占位的是字符类型的位置，%d占位的是整数类型的位置，%.2f占位的是小数类型(会四舍五入)的位置

13 print("Welcome to 'From zero single row':[%s]! Your have money:[%d]元!Your height is[%.2f]cm"%(name,money,180.235))

14 #用format格式，这个格式，就不用担心顺序问题，但参数夺得时候用这个

15 print("Welcome to 'From zero single row':[{name}]! 'Your have money:[{money}]元'".format(money=money,name=name))

16

17 #练习，输入名字和年龄，然后格式化打印：你的名字是：xxx，你的年龄是：xx。

**5.条件判断**

　　这里不再赘述理论了，看下伪代码：　  
　　if 我比你大：  
　　　　你叫我哥  
　　else:  
　　　　我叫你哥  
　　直接上再python里的判断语句：

1 #条件判断，不像其他语言有什么case等多种判断，python里只有if判断

2 age=input('Please input your age:') #用age接收输入的年龄参数

3 age=int(age) #因为输入的参数是字符类型，所以需要转换成int类型

4 #注意，在python里只要遇到:号，下一个语句一定要缩进，这就是python的特点，简洁，不像其他语言很多标识符号，这里直接缩进就可以

5 if age<=45: #如果满足age<=45这个条件，打印下面这个语句，程序结束

6 print('你是青年') #注意缩进，python里没有其他的标点符号来标识代码，全部考缩进来处理

7 elif age<=59: #如果满足age<=59这个条件，打印下面这个语句，程序结束

8 print('你是中年')

9 elif age<=74: #如果满足age<=74这个条件，打印下面这个语句，程序结束

10 print('你是老年')

11 else : #如果上诉条件都不满足，打印下面这个语句，程序结束

12 print('你是长寿老人')

 　　if判断中的非空即真(空字符串、空列表、空字典、空集合等都符合这一语法)、非0即真、非None即真，这三个用在平常的代码里会减少代码的量，比如现在需要判断用户的输入是否为空，直接把用户输入的参数跟在判断条件后面即可，不需要调用strip和len函数来去前后空格然后取输入长度来判断参数是否为空，代码如下：

1 #非空即真

2 if '': #字符串为空，所以为假，进入下一个判断

3 print('假')

4 elif 'asd': #字符串有值不为空，所以为真，这里会打印真

5 print('真')

6 else:

7 print('over')

8

9 #非0即真

10 if 0: #值是0，所以为假，进入下一个判断

11 print('假')

12 elif 123: #值不是0，所以为真，这里会打印真

13 print('真')

14 else:

15 print('over')

16

17 #非None即真

18 if None: #值为None，所以为假，进入下一个判断，这个类似于''

19 print('假')

20 elif 'asd': #字符串有值不为空，所以为真，这里会打印真

21 print('真')

22 else:

23 print('over')

　　判断条件的短路法则，短路，顾名思义，就是程序短路了，不往后面走了，那这个在判断条件里是怎样的原理呢？这个短路法则主要是判断条件里的连接：and、or两个单词，and翻译是并且，就是左右条件都为真则这个判断结果是真，如1 and 1为真，1 and 0、0 and 1、0 and 0都为假；or翻译是或者，只要左右两边有一个条件为真则判断条件为真，如1 and 1、1 and 0、0 and 1都为真，0 and 0为假。

　　短路法则：A and B，如果A的值为真，取B，如果A的值为假，取A；A or B，如果A的值为真，取A，如果A的值为假，取B。具体我们看下代码：

>>>print(0 and 1)

0

>>>print(1 and 2)

2

>>>print(0 or 4)

4

>>>print(1 or 0)

1

**6.循环**

循环就是让计算机重复的帮我们做一件事情，人都讨厌做重复的枯燥无味的事情，像推磨，人觉的枯燥乏味，然后就有了驴推磨，一圈一圈的循环，人一圈一圈的推岂不累死，所以计算机也一样，当有重复的工作，跟计算机下命令，你重复做就好了，那么python里用来下命令的这个语法就只有while 和 for两个。

　　while循环，顾名思义，当····的时候，就是说当满足条件的时候，我运行while下面的代码，反之，当不满足的时候，我们就不运行，看到这里是不是感觉和上面的if判断是一样的？对没错，就是一样的，但是while的区别在于循环，意思就是说我运行完while下面的代码后，后跑到while开始这里来了，继续判断，如果为真运行下面的代码，为假则结束，然后运行完又回到判断条件这里来，这时你也发现了，只要条件一直为真，那么他将马不停蹄的运行下去，就跟永动机一样，这就是所谓的死循环，所以我们用来判断的这个条件，不能是一个常量，判断条件需要时变化的，不然就是一个死循环，永远也不会结束。看下伪代码：

　　while 我是菜鸟：  
　　　　加最辛苦的班，领最低的工资。

　　如果我不努力学习、进取，一直是菜鸟，我就一直循环这个状态，这就是死循环。那么如何结束掉这个死循环呢，当然是努力、进取提升自己，改变这个判断条件的值，不就结束掉了么，所以我们的这个while条件需要是一个变化的值才行，不然就一直运行下去了，在python里，常用的是用一个计数器来计数，当计数达到预期的值就终止循环，看下代码。

i=0 #定义一个变量，用来判断

while i<=10: #判断变量i的值是否<=10，满足条件往下走

i += 1 #计数，每次循环后i自增1

print(i) #打印变量i,这里会打印1-11

else: #当上面的while循环正常结束，运行else

print('over') #这里会正常结束，所以打印1-11后，打印over

　　while循环里的break和continue，顾名思义，break即破坏掉这个循环，强制结束，while循环里只要遇到break就结束，这个时候就好比驴生病了，不再循环拉磨，直接结束；continue就是结束掉本次循环，继续下一个循环，就好比磨坏了，要重新修一下，这一圈没跑完就不跑了，然后回到开始拉磨的起点，重新拉磨。代码如下。

i=0

while i<=10:

i += 1

if i==5: #当i的值等于5时，执行下面的break

break #while遇到break时，就结束掉整个循环了，所以这段代码运行的结果是 打印1-4

print(i)

else: #当上面的while循环正常结束，运行else

print('over') #这里非正常结束，所以打印1-4后，不打印over

i=0

while i<=10:

i += 1

if i==5: #当i的值等于5时，执行下面的continue

continue #while遇到continue时，就结束掉本次循环，继续下一个循环，所以这段代码运行的结果是 打印1-4，5-11

print(i)

else: #当上面的while循环正常结束，运行else

print('over') #这里会正常结束，所以打印1-4，5-11后，会打印over

**7.课后练习**

综合上面的输入、条件判断和循环等知识点，来写个猜数字的小游戏，游戏规则：产生一个1-100之间的随机数字，接收用户输入，判断输入和随机的值，相等打印猜对了，游戏结束；输入值大于随机的值，提示猜大了，继续；输入值小于于随机的值，提示猜小了，继续。产生随机数，使用模块random，方法：num=random.randint(1,101)，num就是一个随机数。