

# python2基础

- 条件判断语句：根据条件的成立与否，选择相应的逻辑分支
- 关键字：if...elif...else
- 最简格式
  - if 布尔表达式：
    - 待执行逻辑分支
    - 当布尔表达式结果为真，则执行待执行逻辑分支
- 对立条件判断
  - if 布尔表达式：
    - 待执行逻辑分支1
  - else:
    - 待执行逻辑分支2
- 多分支条件判断
  - if 布尔表达式1：
    - 待执行逻辑分支1
  - elif 布尔表达式2：
    - 待执行逻辑分支2
  - elif 布尔表达式3：
    - 待执行逻辑分支3
  - ...
  - else:
    - 以上条件均不成立时执行
    - \*\*越特殊的情况越提前判断
    - 多条件分支只会执行一个分支
- \*只有if关键字可以独立引领一个完整的条件判断逻辑链；else会和同层最近的if组成完整的逻辑链

```
44     i=int(input('请输入一个整数: '))
45     if i>100:
46         print('输入的数大于100')
47     if i==100:
48         print('输入的数等于100')
49     else:
50         print('输入的数小于100')
```

```
test01
D:\Python38\python.exe D:/PycharmProjects/test105/test01.py
请输入一个整数: 130
输入的数大于100
Process finished with exit code 0
```

- 条件判断的扩展

- 三目运算：三目运算是一个表达式，表达式的结果取决于布尔值的真假

- 三目运算的格式： $i = x \text{ if } \text{布尔表达式} \text{ else } y$

- 布尔表达式为真则值为x，否则为y

- $i = 10 \text{ if } 0 \text{ else } 20$

- `print(i)`

```
51     i=10 if 0 else 20
52     print(i)
test01
D:\Python38\python.exe D:/PycharmProjects/test105/test01.py
20
Process finished with exit code 0
```

- 多重条件判断：if语句中的待执行逻辑分支，也是一个条件判断语句

- 内层if的条件判断一定是外层条件判断的扩展

- 你大我更大，你小我更小

- 两个毫不相干的条件组合

```
53     i=10
54     if i<100:
55         if i<30:
56             print(i)
test01
D:\Python38\python.exe D:/PycharmProjects/test105/test01.py
10
Process finished with exit code 0
```

- 根据三整数，判断该三个数能否组成三角形，如果能组成，输出三角形的类型

- 提升：两边之和大于第三边：三角形的类型有：直角三角形、等腰三角形、等边三角形、等腰直三角、钝角三角形和锐角三角形

```
a=int(input('请输入第一个数:'))
b=int(input('请输入第二个数:'))
c=int(input('请输入第三个数:'))
#a,b,c=1,1,2**1/2 #能组成等腰直角三角形
l=[a,b,c]
l.sort()
if l[0]+l[1]>l[2]:
    if l[0]**2+l[1]**2==int(l[2]**2):
        if l[0]==l[1]:
            print('能组成等腰直角三角形')
        else:
            print('能组成直角三角形')
    elif l[0]==l[1] or l[1]==l[2]:
        if l[0]==l[2]:
            print('能组成等边三角形')
        else :
            print('能组成等腰三角形')
    elif l[0]**2+l[1]**2<l[2]**2:
        print('能组成钝角三角形')#4 5 7
    else:
        print('能组成锐角三角形')#7 8 9
else:
    print('输入的数不能组成三角形')
```

- 循环语句

- for循环：可以理解成为了达到某种目标的循环，目标达成循环停止
- while循环：基于条件成立的循环，条件成立循环开始，直到条件不再成立

- \*\*for循环：迭代变量，在集合中依次迭代，直到遍历完整个集合

- for循环的格式

- for 变量 in 集合
  - 待循环的语句组
- 变量从集合中第一个元素开始取值，每取值一次称为迭代一次，待循环语句组就执行一遍
- 难点：变量可以参与到待循环语句组中，也可以不参与
  - 如果只是单纯的循环做某事，通常变量不参与；如果待循环的逻辑中需要这个变量，则变量需要参与

- 范围函数：range(x,y,z)

- x,y,z分别代表起始值、终止值（不含）和步长
- 如果只有两个参数，则默认为起始值和终止值（不含），步长默认为1
- 如果只有一个参数，则为终止值（不含），起始值默认为0，步长默认为1

```
68     for i in range(0,5,1):
69         print(i)

```

test01  
D:\Python38\python.exe D:/PycharmProjects/test105/test01.py  
0  
1  
2  
3  
4  
Process finished with exit code 0

- for循环的扩展:

- \*\*\*列表推导式: 列表中的元素, 来自符合某种条件的集合中的迭代变量
  - `l=[i for i in range(10) if i%3==0]`
  - `print(l)`

```
73     l=[i for i in range(10) if i%3==0]
74     print(l)

```

test01  
D:\Python38\python.exe D:/PycharmProjects/test105/test01.py  
[0, 3, 6, 9]  
Process finished with exit code 0

# for循环的扩展:  
# \*\*\*列表推导式: 列表中的元素, 来自符合某种条件的集合中的迭代变量  
`l = [i for i in range(100) if i%3==0]`  
# 输出100以内的偶数/奇数, 不能使用步长  
# oushu = []  
# jishu = []

- 极大提升代码的效率

- 多变量迭代多集合: 通过多个变量同时迭代多个集合

- 格式 `for i,j,k,... in zip(集合1, 集合2, 集合3, ...):`
  - `zip()`: 将多个集合统一打包处理
  - 执行次数取决于木桶原理

```
70     oushu=[]
71     jishu=[]
72     for i in range(10):
73         if i%2==0:
74             oushu.append(i)
75         else:
76             jishu.append(i)
77     print('偶数:',oushu)
78     print('奇数:',jishu)
79     for x,y in zip(oushu,jishu):
80         print(x,y)

```

test01  
D:\Python38\python.exe D:/PycharmProjects/test105/test01.py  
偶数: [0, 2, 4, 6, 8]  
奇数: [1, 3, 5, 7, 9]  
0 1  
2 3  
4 5  
6 7  
8 9  
Process finished with exit code 0

```

93     for x,y in zip([1,2,3],['a','b','c','d']):
94         print(x,y)

```

test01  
D:\Python38\python.exe D:/PycharmProjects/test105/test01.py  
1 a  
2 b  
3 c  
Process finished with exit code 0

- 字典查询元素,可以双变量for循环迭代items()

```

83     d={'name':'张三','age':18,'hobby':'钓鱼','info':{'a':1,'b':2,'c':[1,2,3,4,5]}}
84     for k,v in d.items():
85         print(k,v)

```

test01  
D:\Python38\python.exe D:/PycharmProjects/test105/test01.py  
name 张三  
age 18  
hobby 钓鱼  
info {'a': 1, 'b': 2, 'c': [1, 2, 3, 4, 5]}  
Process finished with exit code 0

- 多重for循环: for循环语句中的待循环语句组, 也是一个for循环

- \*\*\*外层循环迭代一次, 内层循环执行一轮

```

86     for i in range(1,3):#i=1
87         for j in range(1,5):#j=1,2,3,4
88             print('%s:%s' %(i,j))

```

test01  
D:\Python38\python.exe D:/PycharmProjects/test105/test01.py  
1:1  
1:2  
1:3  
1:4  
2:1  
2:2  
2:3  
2:4  
Process finished with exit code 0

## 练习

- 输出1加到100的总和
- sum=0
- for i in range(101):
- sum=i+sum
- print(sum)

```

141     sum=0
142     for i in range(101):
143         sum=i+sum

```

test01  
D:\Python38\python.exe D:/PycharmProjects/test105/test01.py  
5050  
Process finished with exit code 0

- 不使用数据类型强制转换, 输出100以内包含7的数。例如: 7、17、71等
- seven=[i for i in range(100) if i%10==7 or i//10==7]
- print(seven)

```

146     seven=[i for i in range(100) if i%10==7 or i//10==7]
147     print(seven)
148

```

test01  
D:\Python38\python.exe D:/PycharmProjects/test105/test01.py  
[7, 17, 27, 37, 47, 57, 67, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 87, 97]  
Process finished with exit code 0

●

- 已知数列1,1,2,3,5,8...,观察规律，输出该数列指定任意位置的数
- ```
n=int(input('请输入斐波那契数列的第几位数：'))
```
- ```
l=[1,1]
```
- ```
for i in range(n-2):
```
- ```
    l.append(l[-1]+l[-2])
```
- ```
print(l)
```
- ```
print(l[-1])
```

The screenshot shows the PyCharm interface with a code editor and a terminal window. The code is as follows:

```
148 n=int(input('请输入斐波那契数列的第几位数：'))
149
150 l=[1,1]
151 for i in range(n-2):
152     l.append(l[-1]+l[-2])
153 print(l)
154 print(l[-1])
```

The terminal window shows the following output:

```
test01
D:\Python38\python.exe D:/PycharmProjects/test105/test01.py
请输入斐波那契数列的第几位数: 10
[1, 1, 2, 3, 5, 8, 13, 21, 34, 55]
55
Process finished with exit code 0
```

●

- 冒泡排序：一组数据，经过多轮多次相邻元素两两之间的比较，小的靠前，大的排后，最后得到一组有序数据
- 规律：
  - 1、比较的总轮次数为元素的个数减一
  - 2、随着轮次的递增，每轮比较的次数递减

● 规律：

- 1、比较的总轮次数为元素的个数减一
- 2、随着轮次的递增，每轮比较的次数递减
- ```
l=[3,7,6,9,1]
```
- ```
for i in range(len(l)-1):#i:0,1,2,3
```
- ```
    for j in range(len(l)-1-i):#5-1-0=4;5-1-1=3;5-1-2=2;5-1-3=1
```
- ```
        if(l[j]>l[j+1]):
```
- ```
            # temp=l[j]
```
- ```
            # l[j]=l[j+1]
```
- ```
            # l[j+1]=temp
```
- ```
            l[j],l[j+1]=l[j+1],l[j]
```
- ```
    print(l)
```

The screenshot shows the PyCharm interface with a code editor and a terminal window. The code is as follows:

```
130
131 l=[3,7,6,9,1]
132 for i in range(len(l)-1):#i:0,1,2,3
133     for j in range(len(l)-1-i):#5-1-0=4;5-1-1=3;5-1-2=2;5-1-3=1
134         if(l[j]>l[j+1]):
135             # temp=l[j]
136             # l[j]=l[j+1]
137             # l[j+1]=temp
138             l[],l[j+1]=l[j+1],l[j]
139 print(l)
```

The terminal window shows the following output:

```
test01
D:\Python38\python.exe D:/PycharmProjects/test105/test01.py
[1, 3, 6, 7, 9]
Process finished with exit code 0
```

●

- 九九乘法表

- `for i in range(1, 10):`
- `for j in range(1, i+1):`
- `print("%s*s=%s" %(j,i,j*i), end=' ')`
- `print()`

```

267 # 
268 for i in range(1, 10):
269     for j in range(1, i+1):
270         print("%s*s=%s" %(j,i,j*i), end=' ')
271     print()
272 
273 # class Demo:
274 #
# 
# 
```

Run test01  
D:\Python38\python.exe D:/PycharmProjects/test105/temp/test01.py

```

1*1=1
1*2=2 2*2=4
1*3=3 2*3=6 3*3=9
1*4=4 2*4=8 3*4=12 4*4=16
1*5=5 2*5=10 3*5=15 4*5=20 5*5=25
1*6=6 2*6=12 3*6=18 4*6=24 5*6=30 6*6=36
1*7=7 2*7=14 3*7=21 4*7=28 5*7=35 6*7=42 7*7=49
1*8=8 2*8=16 3*8=24 4*8=32 5*8=40 6*8=48 7*8=56 8*8=64
1*9=9 2*9=18 3*9=27 4*9=36 5*9=45 6*9=54 7*9=63 8*9=72 9*9=81
# 
```

- while循环：基于条件成立的循环，条件成立循环开始，直到条件不再成立
  - 如果没有外力干涉或内部崩溃，while通常都是死循环
- while循环的格式
  - while 布尔表达式：
    - 待循环语句组

```

124 # print('hello_world')
125 l='hello_world'
126 i=0
127 while i<len(l):
128     print(l[i])
129     i=i+1
130 
131 # l=[3,7,6,9,1]
# 
```

Run test01  
D:\Python38\python.exe D:/PycharmProjects/test105/temp/test01.py

```

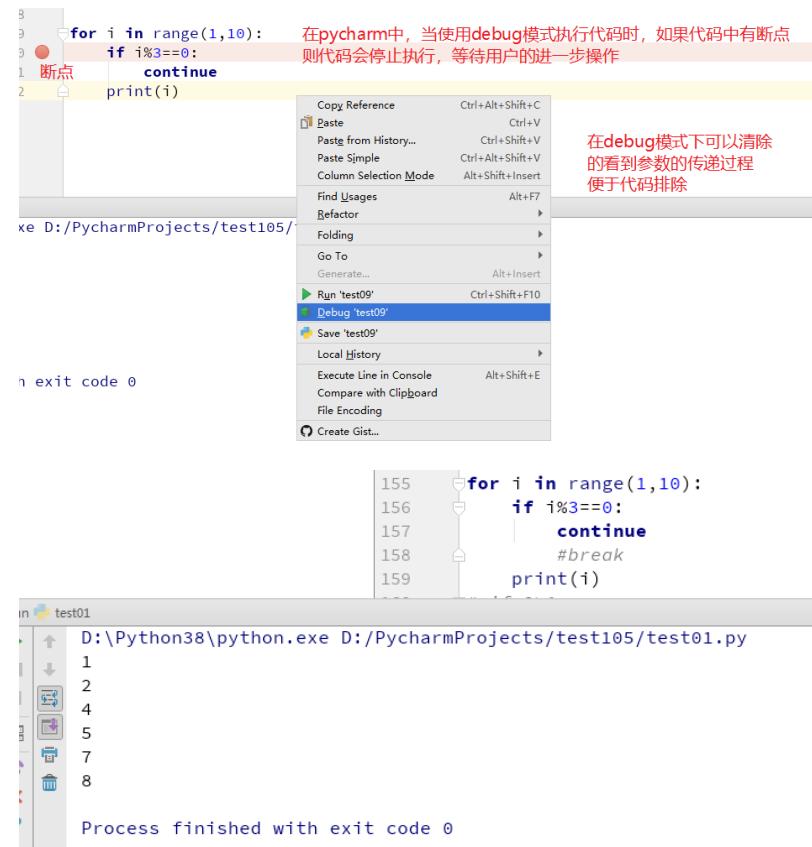
h
e
l
l
o
-
w
o
r
l
d

Process finished with exit code 0
# 
```

- while循环和for循环的区别
  - for循环有迭代变量和集合；while循环只有条件
  - \*当范围确定通常使用for循环；当条件确定通常使用while循环
  - for循环通常可以改写成while循环

- `continue`语句：配合循环使用，当代码执行到`continue`语句时，立刻回到循环开始的地方，跳过本次循环，执行下一次循环

- 通常在循环中充当例外的处理



- break语句：配合循环使用，当代码执行到break语句时，立刻摧毁循环，跳出循环之外

- 通常在循环中发现特征数据



- pass语句：没有任何逻辑的语句，通常充当代码的补齐和占位

A screenshot of the PyCharm IDE interface. The code editor shows a snippet of Python code with line numbers 155 to 159. Lines 155, 156, and 157 are highlighted in yellow, while line 158 is blue. The code is:

```
155 if 2>1:  
156     print(1)  
157 else:  
158     pass  
159
```

The run tab shows the command: D:\Python38\python.exe D:/PycharmProjects/test105/test01.py. The output window displays the result: 1. Below the output, it says Process finished with exit code 0.

- 代码入口与代码调用
- 代码的调用：可以将已有的代码复用，面向对象编程的特色之一
- 代码调用的关键字：from...import
  - 标准库或同层文件下的调用
    - 标准库：python主程序自带的所有模块
    - 引用格式：import XXX
  - 非标准库的调用
    - 非标准库：第三方模块
    - from 库/模块/类 import 模块/类/函数/变量
  - 引用的快捷方式：将需要引用的类/函数/变量写在空白处等待报错，弹出红色小提示后选择需要引入的对象
    - ALT+回车键快速导入
- 代码入口：系统隐藏变量 \_\_name\_\_
  - 在当前模块下 \_\_name\_\_ 的值为 \_\_main\_\_

A screenshot of the PyCharm IDE interface. The code editor shows a snippet of Python code with line number 183. Line 183 is highlighted in yellow and contains the code print(\_\_name\_\_). The run tab shows the command: D:\Python38\python.exe D:/PycharmProjects/test105/test01.py. The output window displays the result: \_\_main\_\_. Below the output, it says Process finished with exit code 0.
  - 被其他模块引用时， \_\_name\_\_ 的值变为模块名称

test01.py

```
174 # if 2>1:  
175 #     print(1)  
176 # else:  
177 #     pass  
178 # l=[1]  
179 # for i in l:  
180 #     l.append(i)  
181 #     print('死循环')  
182 x=0  
183 print(__name__)  
184
```

test10.py

```
1 import test01
```

test10

```
D:\Python38\python.exe D:/PycharmProjects/test105/test10.py  
test01  
Process finished with exit code 0
```

test01

```
183 if __name__=='__main__':  
184     print(__name__)  
185
```

test01

```
D:\Python38\python.exe D:/PycharmProjects/test105/test01.py  
__main__  
Process finished with exit code 0
```

- 可以根据`__name__`的特性，进行条件判断
  - 保护当前模块下的调试代码被引用时，不会被执行

- 提供一个代码执行的入口

- 作业：

- 输入一个正整数，判断该数是否为质数，如果是质数，输出该数是质数，如果不是质数，输出该数的因式，例如：输入7，输出7是质数；输入18，输出18不是质数，它的因式有[2,3,6,9]

```

190 num=10
191 l=[]
192 for i in range(2,num):
193     if num%i==0:
194         l.append(i)
195 if l:
196     print('%s不是质数，它的因式分解有: %s' %(num,l))
197 else:
198     print('%s是质数' %num)
199
200 # s='1234'
201 # x=0
202 # d={'0':0,'1':1,'2':2,'3':3,'4':4,'5':5,'6':6,'7':7,'8':8,'9':9}

```

test01  
D:\Python38\python.exe D:/PycharmProjects/test105/test01.py  
10不是质数，它的因式分解有：[2, 5]

Process finished with exit code 0

```

190 num=17
191 l=[]
192 for i in range(2,num):
193     if num%i==0:
194         l.append(i)
195 if l:
196     print('%s不是质数，它的因式分解有: %s' %(num,l))
197 else:
198     print('%s是质数' %num)
199
200 # s='1234'
201 # x=0
202 # d={'0':0,'1':1,'2':2,'3':3,'4':4,'5':5,'6':6,'7':7,'8':8,'9':9}

```

test01  
D:\Python38\python.exe D:/PycharmProjects/test105/test01.py  
17是质数

Process finished with exit code 0

- num=17
- l=[]
- for i in range(2,num):
- if num%i==0:
- l.append(i)
- if l:
- print('%s不是质数，它的因式分解有: %s' %(num,l))
- else:
- print('%s是质数' %num)
- 
- 不使用任何数据类型强制转换，将类似于\*\*'1234'\*\*字符串，转换为整数型

```

200 s='1234'
201 x=0
202 d={'0':0,'1':1,'2':2,'3':3,'4':4,'5':5,'6':6,'7':7,'8':8,'9':9}
203 for i in range(len(s)):
204     x=x+d[s[i]]*10**((len(s)-1)-i)
205     print(x,type(x))

```

test01  
D:\Python38\python.exe D:/PycharmProjects/test105/test01.py  
1234 <class 'int'>

Process finished with exit code 0

- s='1234'
- x=0
- d={'0':0,'1':1,'2':2,'3':3,'4':4,'5':5,'6':6,'7':7,'8':8,'9':9}
- for i in range(len(s)):
- x=x+d[s[i]]\*10\*\*((len(s)-1)-i)
- print(x,type(x))