

0

扫码签到，累积图书馆积分，获取定制纪念品



1

讲座预告——微信公众号“i学堂XMU”





扫一扫二维码，入群聊





<https://lecture.xmu.edu.cn>

提示:

也可通过图书馆主页获取
(主页-悦读-讲座课件)



厦门大学讲座报名系统

[首页](#)
[思明校区](#)
[翔安校区](#)
[登录](#)

[用户登录](#)

[重设密码](#)

[导航](#)

- i学堂
- 宣传月
- 新生培训
- 在线培训
- 往期讲座

[讲座日历](#)

« 2019年3月 »

日	一	二	三	四	五	六
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23

近期讲座

i学堂Excel可视化专题 (1) : Excel数据处理技巧 【思明校区】杨薇	2019-03-05(星期二) 19:30	点击报名
i学堂平面设计专题 (1) : 初识PS—基础上手指南 【翔安校区】王楷	2019-03-06(星期三) 19:30	点击报名
i学堂Excel可视化专题 (2) : Excel函数入门 【思明校区】杨薇	2019-03-07(星期四) 19:30	点击报名
i学堂平面设计专题 (2) : 设计元素的获取——PS之抠图大法 【翔安校区】王楷	2019-03-09(星期六) 19:30	点击报名

[More...](#)

讲座课件

i学堂翔安场: 详解文献管理利器 EndNote X9 (进阶篇) 【翔安校区】张妮妮	2018-12-18 19:30	i学堂翔安场: 详解文献管理软件 EndNote X9 (进阶篇).rar
i学堂本部场: MATLAB编程技巧与数据分析 【思明校区】许悦伊	2018-12-13 19:30	i学堂本部场: MATLAB编程技巧与数据分析.rar
i学堂翔安场: MATLAB编程技巧与数据分析 【翔安校区】许悦伊	2018-12-12 19:30	i学堂翔安场: MATLAB编程技巧与数据分析.rar
i学堂本部场: 详解文献管理利器 EndNote X9 (入门篇) 【思明校区】韩冬丽	2018-12-11 19:30	i学堂本部场: 详解文献管理软件 EndNote X9-韩冬丽.pptx

[More...](#)

联系我们

请使用“我的图书馆”的帐号(读者证号或学号、教工号)和密码登录本系统。

网站使用问题	讲座意见及建议(思明)	讲座意见及建议(翔安)
• 联系老师: 魏老师	• 联系老师: 麦老师	• 联系老师: 李老师
• 联系电话: 0592-2184973-810	• 联系电话: 0592-2188693	• 联系电话: 0592-2888315
• 联系邮箱: xywei@xmu.edu.cn	• 联系邮箱: mailin@xmu.edu.cn	• 联系邮箱: shining@xmu.edu.cn



- 哔哩哔哩网站 (<https://www.bilibili.com>) 搜索 “厦大图书馆”，可学习往期课程

共找到1个用户

默认排序

全部用户



厦大图书馆

LV2

+ 关注

稿件: 14 粉丝: 832

此用户没有个性签名啊啊啊

厦门大学图书馆 i学堂



i学堂-毕业论文WORD排版全攻略(2)-杨薇-202003
2020-03-17

厦门大学图书馆 i学堂



i学堂-如何运用LaTex排版论文-魏小燕-20200306
2020-03-10



i学堂-文献管理软件
EndNote X9使用入门-韩
2020-03-07

[全部14个稿件>](#)



五节课 入门Python编程!

主讲：厦门大学WISERCLUB团队

地点：思明校区图书馆总馆321教室（B站同步直播）

第三讲 Python控制流与函数

第四讲 NumPy 与Pandas

第五讲 Python数据可视化

时间：5月16日（周五）19:00

时间：5月20日（周二）19:00

时间：5月23日（周五）19:00



Python 控制结构与函数

张文豪 经济学院统计学与数据科学系

WISERCLUB 课程部

2025年5月16日



◆ 参考书籍

- 《Python语言程序设计》 刘卫国
- Python 官方文档

◆ 编程练习网站

- PythonTip
<https://edu.py2fun.com/learn#/problemset/all/1>
- Leetcode
<https://leetcode.cn/problemset/>



主要内容

- 控制结构
 - 3种控制结构
 - 分支结构之if else
 - 循环结构之for
 - 循环结构之while
 - Python语法规则
- 函数
 - 定义函数
 - 函数的调用
 - 函数的参数
 - 高阶函数



01

Part One

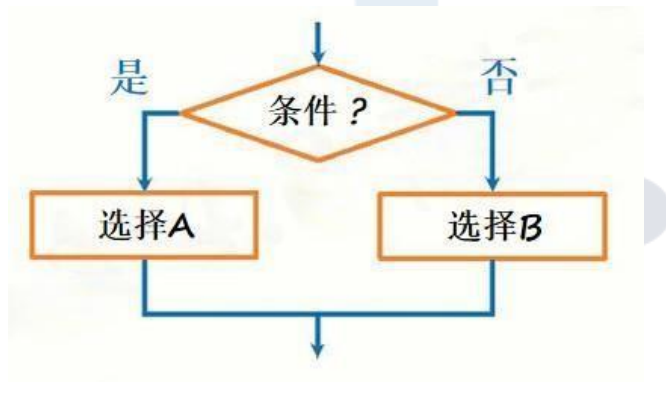
控制结构

3种控制结构

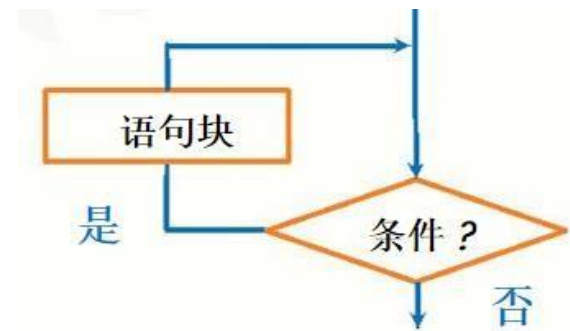
顺序结构



判断结构（分支结构）



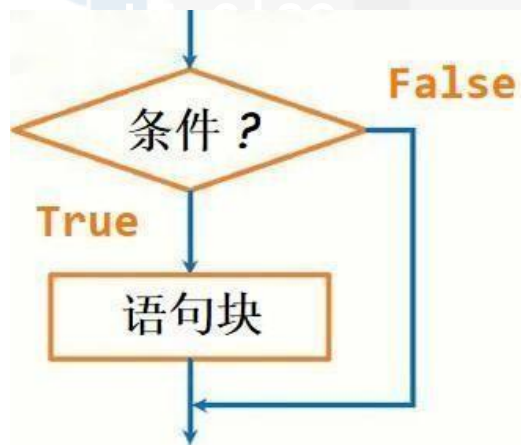
循环结构



分支结构之 if else

单分支结构

if <条件> :
 <语句块>



```
In [1]: value=1  
if value==1:  
    print('right')  
  
right
```

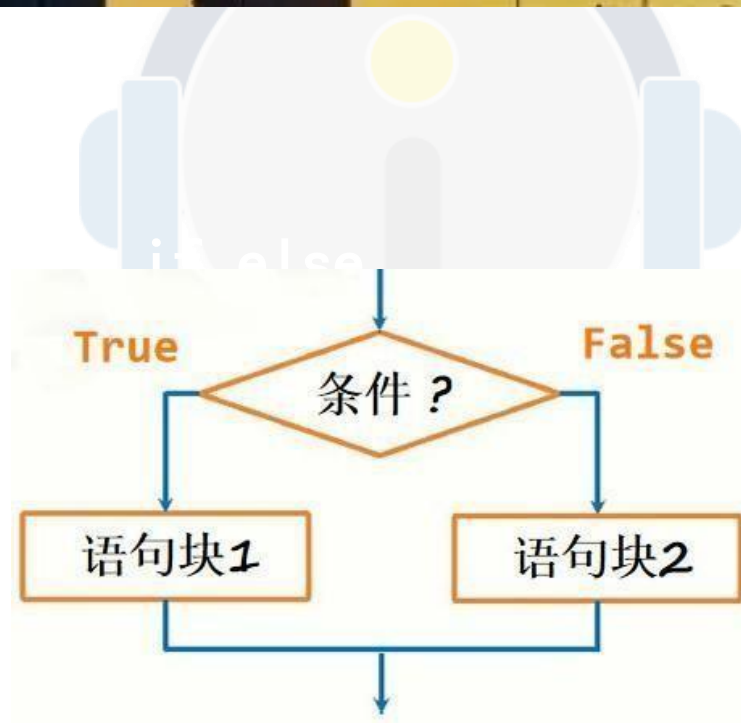
```
In [2]: if 'a' in 'apple':  
        print('right')  
  
right
```


分支结构之 if else

二分支结构

if <条件> :
 <语句块1>

else :
 <语句块2>



```
In [3]: value=1  
if value==2:  
    print('right')  
else:  
    print('false')
```

false

```
In [4]: if 'a' not in 'apple':  
        print('right')  
else:  
        print('false')
```

false

```
In [5]: if 'a' in 'apple':  
        print('right')  
else:  
        print('false')
```

right

分支结构之 if else

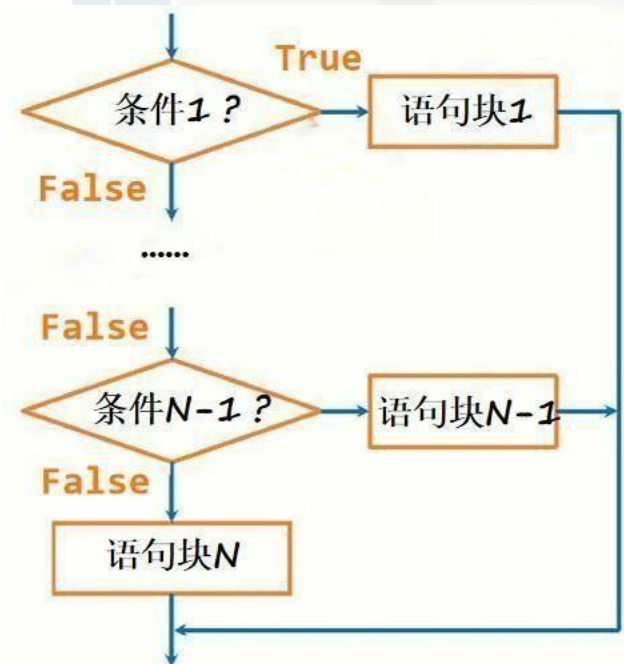
多分支结构

if <条件1> :
 <语句块1>

elif <条件2> :
 <语句块2>

.....

else :
 <语句块N>



In [6]:

```
score=61
if score < 60:
    print('不及格')
elif 60 <= score < 70:
    print('及格')
elif 70 <= score < 80:
    print('中等')
else:
    print('优秀')
```

及格

分支结构之 if else

```
In [6]: score=61
if score < 60:
    print('不及格')
elif 60 <= score < 70:
    print('及格')
elif 70 <= score < 80:
    print('中等')
else:
    print('优秀')
```

及格

01

注意缩进

语句块要比条件缩进一层，可以使用Tab键

02

注意冒号

冒号及其他符号要用英文格式打出

分支结构之 if else

03 Python比较运算符

运算符	描述
==	等于 - 比较对象是否相等
!=	不等于 - 比较两个对象是否不相等
<>	不等于 - 比较两个对象是否不相等。python3 已废弃。
>	大于 - 返回x是否大于y
<	小于 - 返回x是否小于y。所有比较运算符返回1表示真，返回0表示假。这分别与特殊的变量 True 和 False 等价。
>=	大于等于 - 返回x是否大于等于y。
<=	小于等于 - 返回x是否小于等于y。



分支结构之 if else

04

练习：判断今日厦门的空气质量

一级： 空气污染指数 ≤ 50 ， 空气质量状况属于优。

二级： 空气污染指数 ≤ 100 ， 空气质量状况属于良。

三级： 空气污染指数 ≤ 150 ， 空气质量状况属于轻微污染。

五级： 空气污染指数 ≤ 300 ， 空气质量状况属于中度污染。

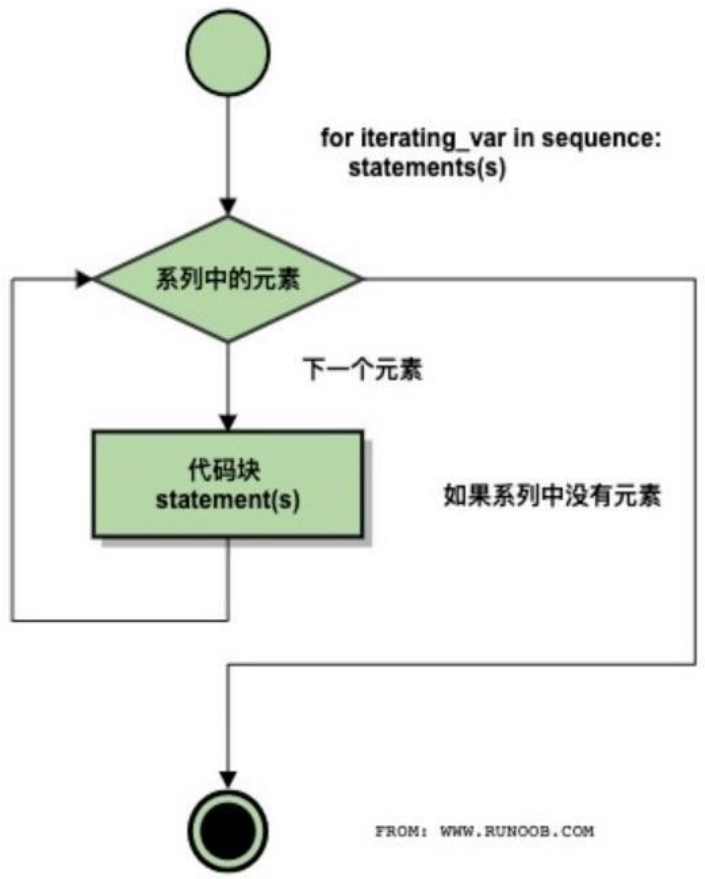
六级： 空气污染指数 > 300 ， 空气质量状况属于重度污染。

循环结构之 for

遍历循环

for <循环变量> *in* <遍历结构> :

<语句块>



```
In [11]: for i in [0, 1, 2]:  
         print(i)
```

0
1
2

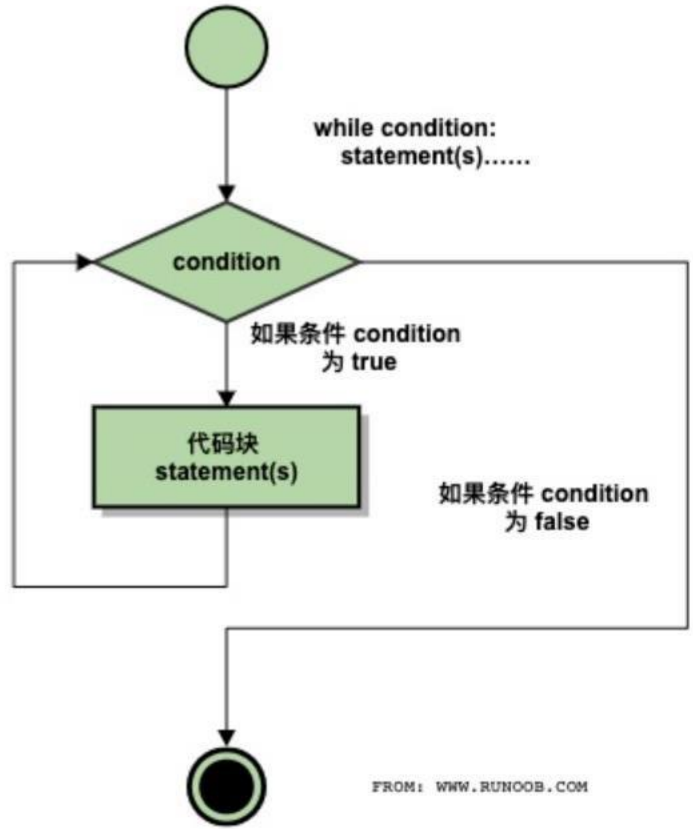
```
In [12]: for i in range(3):  
         print(i)
```

0
1
2

循环结构之 while

无限循环

while <条件> :
 <语句块>



FROM: WWW.RUNOOB.COM

```
In [13]: a=1
         while a<4:
             print(a)
             a=a+1

1
2
3
```

循环控制： continue & break

01

continue

跳过当前循环的剩余语句，然后
继续进行下一轮循环

输出数字1到4但跳过3

```
In [1]: for i in range(1,5):  
        if i == 3:  
            continue  
        print(i)
```

1
2
4

```
In [1]: a=0  
while a<4:  
    a=a+1  
    if a == 3:  
        continue  
    print(a)
```

1
2
4

循环控制： continue & break

02 break

跳出并结束当前整个循环，执行
循环后面的语句

输出小于3的数字

```
In [5]: for i in range(1,5):  
        if i == 3:  
            break  
        print(i)
```

1
2

```
In [6]: a=0  
while a<4:  
    a=a+1  
    if a == 3:  
        break  
    print(a)
```

1
2

循环结构练习

01 练习一

输出1到16中，除6以外能被三整除的数

```
In [9]: for i in range(1, 17):  
        if i == 6:  
            continue  
        if i%3==0:  
            print(i)
```

3
9
12
15

02 练习二

输入两个正整数n和m，找它们的最大公因数

```
In [10]: n = int(input("输入正整数1: "))  
m = int(input("输入正整数2: "))  
min = 0  
if m > n:  
    min = n  
else:  
    min = m  
for i in range(min, 0, -1):  
    if n % i == 0 and m % i == 0:  
        print(f"它们最大公因数为{i}")  
        break
```

输入正整数1: 56
输入正整数2: 38
它们最大公因数为2

Python语法规则

- **块和语句的边界会自动检测**：python没有大括号或者“begin/end”等字符，一般也不以分号终止。Python使用首行下的语句缩进把嵌套块内的语句组合起来。
- **文件的空白行、空格、注释都会忽略**：注释以#开头，或为""" """代码块。
- **整齐的代码风格**：Python因为通过缩进组织代码，所以对缩进有严格要求。整个代码通常看起来非常整齐。
- **不要混合使用tabs和空格**：尽管两者都可以使用，但是一旦混用，会导致在不同的编辑器上呈现不同的缩进效果，并且难以修改

◆ 小结

- if else语句
- for语句
- while语句

- continue 关键字
- break 关键字

02

PART TWO

函数





什么是函数？



- 数学上的function

$$f_1(x) = 2x$$

$$f_2(x_1, x_2) = 2x_1 + 3x_2$$

- 代码实现

```
In [1]: def f1(x):  
        y=2*x  
        return y
```

```
In [2]: f1(2)
```

```
Out[2]: 4
```

```
In [3]: f1(8)
```

```
Out[3]: 16
```

```
In [5]: def f2(x1, x2):  
        y=2*x1+3*x2  
        return y, x1, x2
```

```
In [6]: f2(2, 3)
```

```
Out[6]: (13, 2, 3)
```





什么是函数？

- 也可以一种处理行为
 - 1.把输入的单词复制两遍
 - 2.画一个圆
- 代码实现

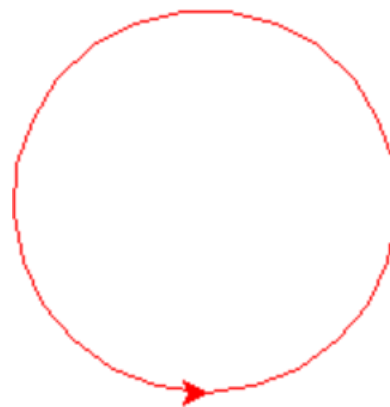
```
In [14]: def f3(x):  
         y=2*x  
         return y
```

```
In [15]: f3('apple')
```

```
Out[15]: 'appleapple'
```

```
In [50]: def circle(x):  
         import turtle  
         turtle.color('red')  
         turtle.circle(x)  
         turtle.done()
```

```
In [*]: circle(80)
```





什么是函数？

- 函数是一段具有特定功能的、可重用的语句组；函数是一种功能的抽象，一般函数表达特定功能

- 定义函数

def <函数名>(<参数(0个或多个)>) :

 <函数体>

return <返回值>





函数的调用

- 调用自己定义的函数

```
In [5]: def f2(x1, x2):  
        y=2*x1+3*x2  
        return y, x1, x2
```

```
In [6]: f2(2, 3)
```

```
Out[6]: (13, 2, 3)
```

- 调用python内置函数

Built-in Functions

A

[abs\(\)](#)
[aiter\(\)](#)
[all\(\)](#)
[anext\(\)](#)
[any\(\)](#)
[ascii\(\)](#)

E

[enumerate\(\)](#)
[eval\(\)](#)
[exec\(\)](#)

F

[filter\(\)](#)
[float\(\)](#)

L

[len\(\)](#)
[list\(\)](#)
[locals\(\)](#)

M

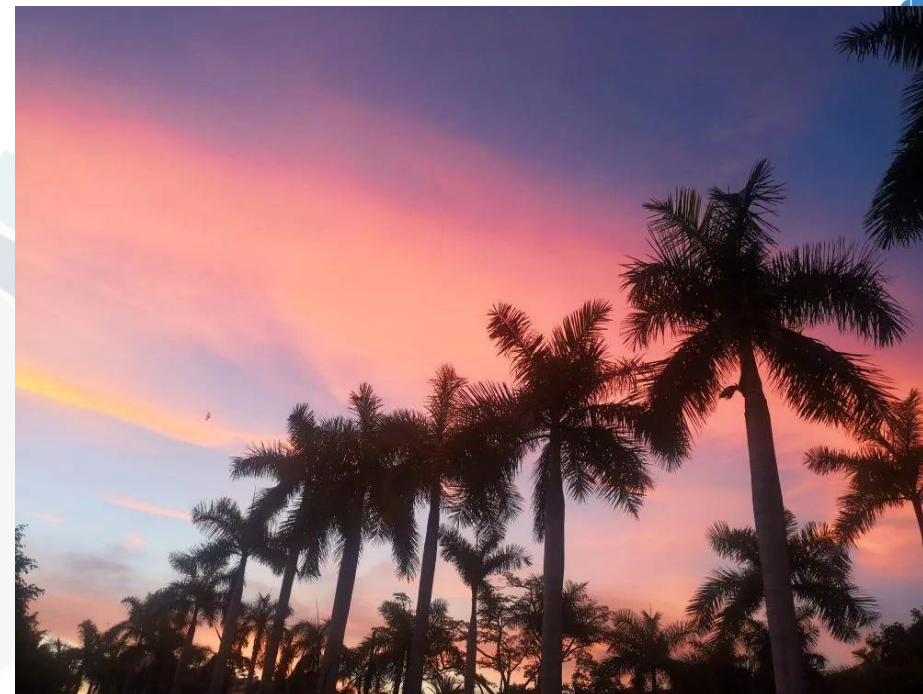
[map\(\)](#)
[max\(\)](#)

R

[range\(\)](#)
[repr\(\)](#)
[reversed\(\)](#)
[round\(\)](#)

S

[set\(\)](#)



```
In [52]: a=[1, 2, 3]
```

```
In [53]: sum(a)
```

```
Out[53]: 6
```

```
In [54]: abs(-1)
```

```
Out[54]: 1
```




函数的调用

- 调用模块（库）中的函数
 - python有自带库
 - 可以下载第三方库，然后调用

自带库的调用，用math库举例

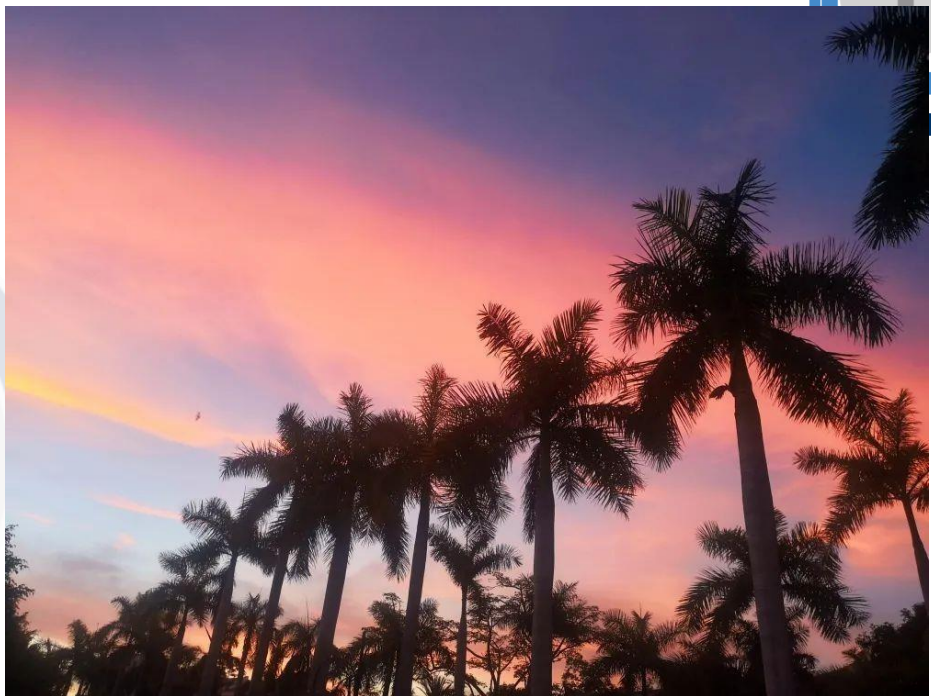
```
In [57]: import math #首先要导入库
```

```
In [58]: math.sqrt(16) #计算平方根
```

```
Out[58]: 4.0
```

```
In [59]: math.exp(5) #计算指数
```

```
Out[59]: 148.4131591025766
```



python内置的20个常用模块

os	sys	time	datetime
random	subprocess	hashlib	json
pickle	shutil	configparser	yaml
itertools	re	calendar	math
uuid	Queue	logging	copy



函数的调用



- 调用第三方库中的函数

第三方库的调用, 以pandas为例

```
In [2]: pip install pandas #首次使用时, 需要先用 pip install + “库名”, 下载; 这里的pandas库是已经下载完的
```

```
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: pandas in d:\anaconda\anaconda\lib\site-packages (1.4.2)
Requirement already satisfied: python-dateutil>=2.8.1 in d:\anaconda\anaconda\lib\site-packages (from pa
Requirement already satisfied: numpy>=1.18.5 in d:\anaconda\anaconda\lib\site-packages (from pandas) (1.
Requirement already satisfied: pytz>=2020.1 in d:\anaconda\anaconda\lib\site-packages (from pandas) (202
Requirement already satisfied: six>=1.5 in d:\anaconda\anaconda\lib\site-packages (from python-dateutil)
Note: you may need to restart the kernel to use updated packages.
```

```
In [3]: import pandas as pd #导入pandas库
```

```
In [4]: a=[[1, 2, 3], [4, 5, 6]] #写一个列表
```

```
In [5]: c=pd.DataFrame(a) #将具有两个子列表的列表a转换成矩阵形式
```

```
In [6]: c
```

Out[6]:

	0	1	2
0	1	2	3
1	4	5	6



函数-练习题



1、计算10到20中，除16以外每个数的阶乘

```
In [8]: def fn(x):    #先定义一个求阶乘的函数
        s=1
        for i in range(1, x+1):
            s=s*i
        return s
```

```
In [9]: for x in range(10, 21): #将函数带入循环中使用
        if x == 16:
            continue
        else:
            print(x, fn(x))
```

2、写一个函数fun (x) ， 其能输出1到x中，除6、9以外能被三整除的数

```
In [13]: def fun(x):
        for i in range(1, x):
            if i == 6:
                continue
            if i==9:
                continue
            if i%3==0:
                print(i)
```





函数的参数

- 函数定义时可以为某些参数指定默认值，构成可选参数（默认参数）

```
def <函数名>(<非可选参数>, <可选参数>) :  
    <函数体>  
    return <返回值>
```

如计算x的平方，这里的2就是默认参数

```
In [15]: def power(x, n=2): #这里的n=2, 就是默认参数  
        y=x**n # '**'代表幂运算  
        return y
```

```
In [16]: power(5)
```

```
Out[16]: 25
```



注意：默认参数必须在普通参数（非可选参数）后面定义

高阶函数

- 变量可以指向函数

```
In [1]: abs(-1)
```

```
Out[1]: 1
```

```
In [2]: f=abs
```

```
In [3]: f(-1)
```

```
Out[3]: 1
```

`abs`是函数本身。更确切的说，它是一段代码的名字。因此，我们可以将将这串代码赋予其他的名字。

这里变量`f`指向了`abs`的函数。





高阶函数

- 函数名也是变量

```
In [7]: def add(x, y, f):  
        return f(x)+f(y)
```

```
In [5]: add(-1, -3, abs)
```

```
Out[5]: 4
```

这里abs作为了参数传入函数中进行使用。也可以传入其他参数

```
In [11]: add([1, 2, 3], [4, 4, 6], sum)
```

```
Out[11]: 20
```



注：关于sum()的用法

```
In [13]: sum([-1, -3])
```

```
Out[13]: -4
```




扩展内容

- 匿名函数

- 匿名函数，也叫 `lambda` 函数，是一种不需要显式命名的函数。常用于写很短的、临时使用的一行函数。
- 其语法结构：`lambda 参数: 表达式`

一个例子：计算一个数的平方

```
square = lambda x: x * x  
  
# 使用匿名函数  
print(square(5)) # 输出: 25
```

这里我们没有`def`也没有`return`





扩展内容

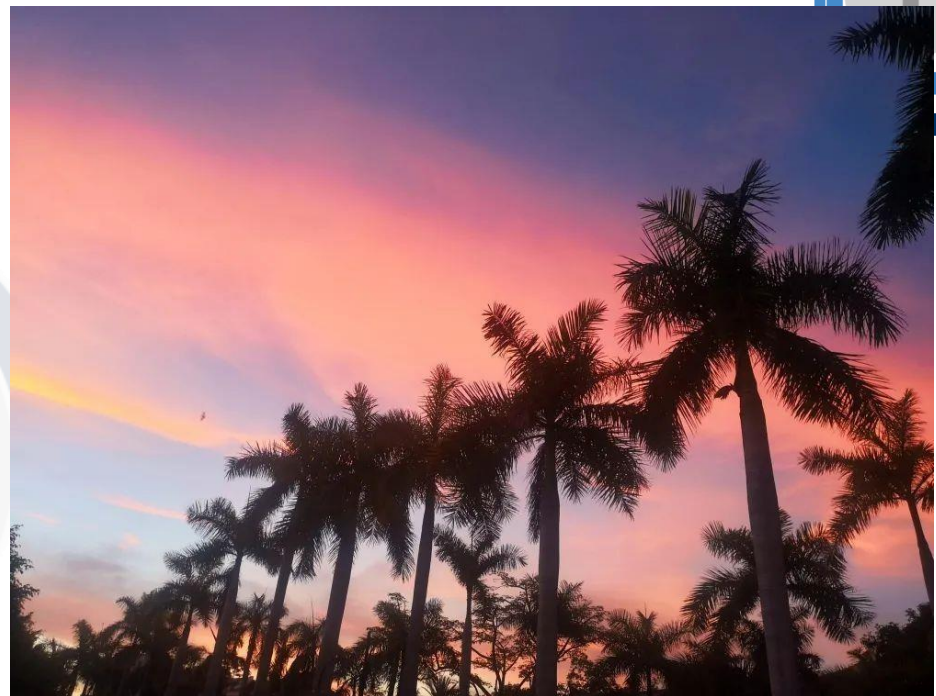
- “函数调用他自己”：递归

递归是指函数自己调用自己。它通常包含两个部分：

- 终止条件/递归出口 (Base Case) : 防止无限调用。
- 递归调用 (Recursive Case) : 缩小问题规模，逐步逼近终止条件。

一个例子：
计算阶乘

```
def factorial(n):  
    if n == 0 or n == 1:  
        return 1 # 递归终止条件  
    return n * factorial(n - 1) # 递归调用  
print(factorial(5)) # 输出: 120
```



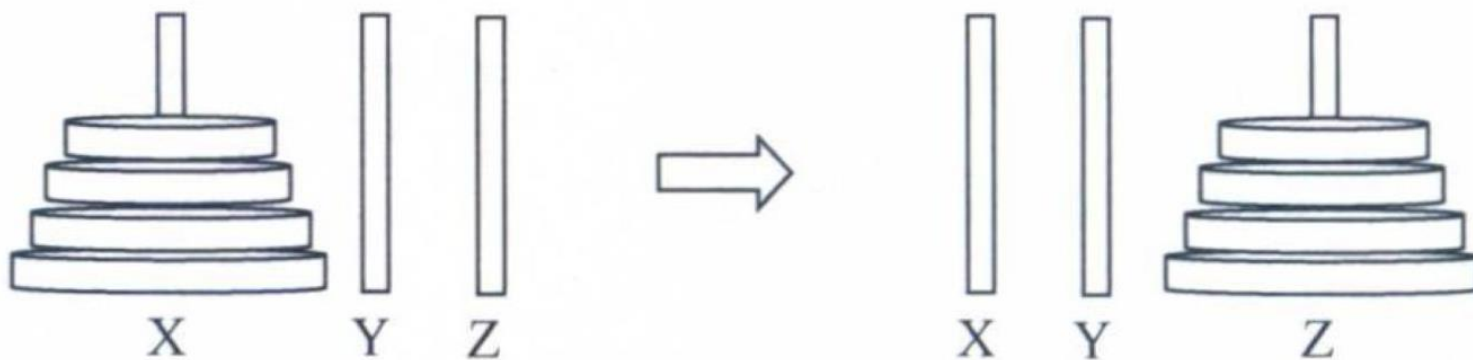


扩展内容



- 思考题：汉诺塔问题 (Tower of Hanoi)

问题描述：有三个柱子：X、Y、Z。有若干个大小不一的圆盘，最初按大小顺序叠在柱子X上，大在下，小在上。目标是将所有圆盘借助柱子Y移动到柱子Z，每次只能移动一个圆盘，并且不能把大盘子放在小盘子上面。





总结

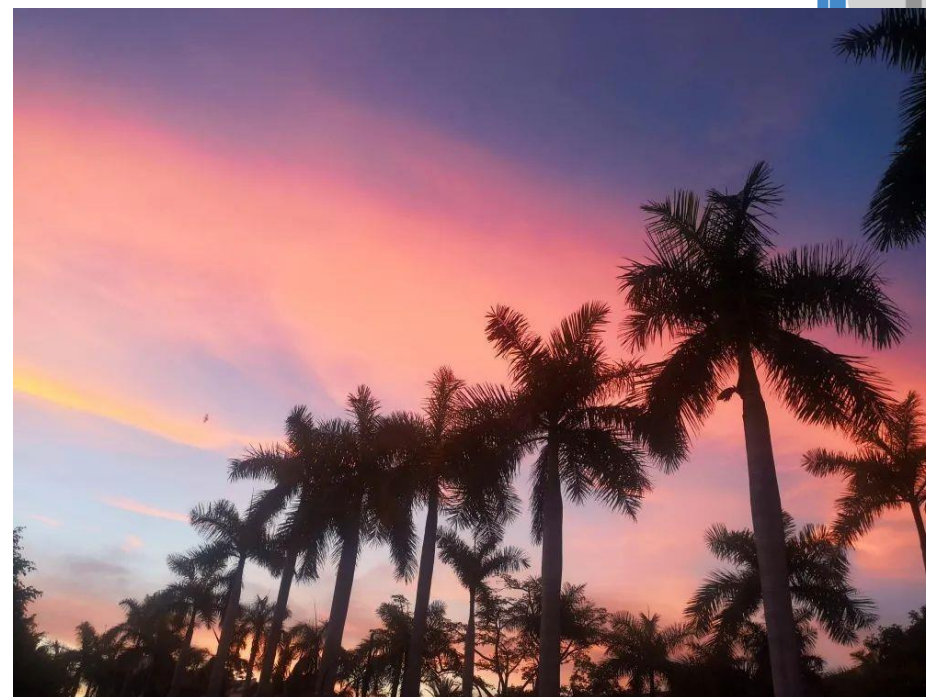


- 控制结构

- 3种控制结构
- 分支结构之if else
- 循环结构之for
- 循环结构之while
- Python语法规则

- 函数

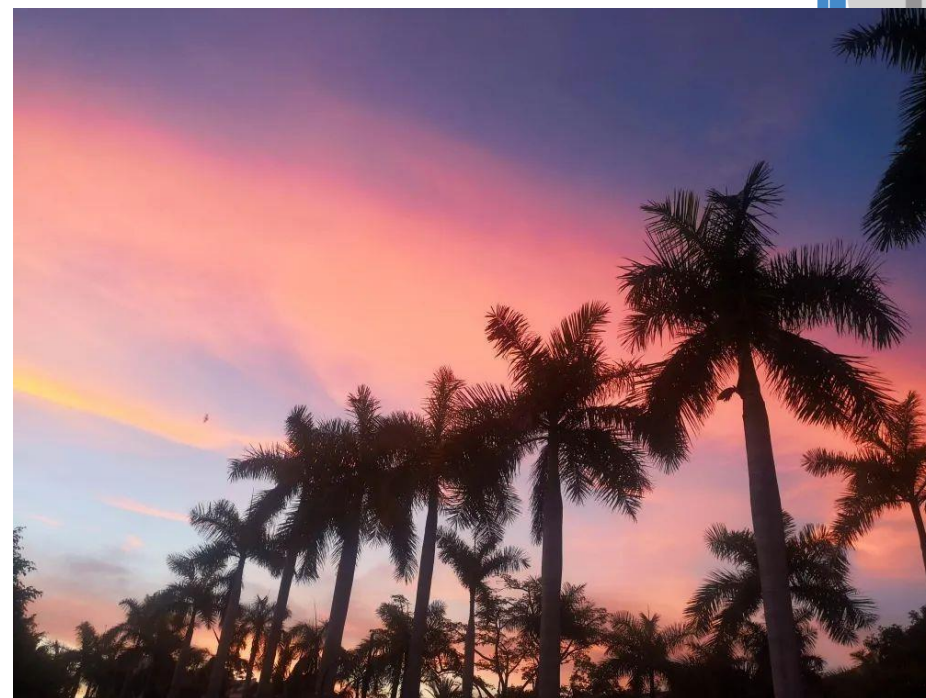
- 定义函数
- 函数的调用
- 函数的参数
- 高阶函数





课后练习

- 写一个程序，找出能被从1到给定数字（包括n）的所有数字整除的最小正数（即最小公倍数）。
- 编写一个程序来判断一个数字是否为素数。
- 编写一个程序来检查一个单词是否为同源词。同源词是指不包含任何重复字母的单词，例如 brown, fox, quick 等。
- 编写一个程序递归判断一个字符串是否为回文。（也可以试试循环的做法）回文是指从前往后读和从后往前读都一样的单词、短语、数字或其他字符序列。在本题中，空字符串也被视为回文。例如：“”，“racecar”，“abcba”
提示：如果字符串长度小于2，则认为是回文；如果字符串的首尾字符不相等，则认为不是回文；递归调用，去掉字符串的首尾字符，继续检查



有机会抽取图书馆文创纪念品!



厦门大学图书馆信息素
养教育服务调研



扫一扫或长按识别二维码