



School of Economics and Management, Beihang University

现代程序设计技术

赵吉昌

jichang@buaa.edu.cn

- Python基础
 - 关键字
 - 代码格式
 - 标识符与变量
 - 数据类型

- 关键词

- `import keyword`
- `print(keyword.kwlist)`
- `with open("test.txt", "w") as f:`
 - 上下文管理器，常在资源管理中用到，能够处理异常
- `None`和任何其他数据类型比较永远返回`False`
- `nonlocal`在函数或其他作用域中使用外层（非全局）变量

- 缩进
 - 缩进空格数可变，但是同一个代码块的语句必须包含相同的缩进空格数
- 多语句
 - 可以用；在同一行显示多条语句
 - 语句很长时可使用\来实现多行语句
 - 在 [], {}, 或 () 中的多行语句不需要使用\
- 注释
 - 单行注释用#
 - 多行注释用'''或"""

- 标识符

- 标识符由字母、数字、下划线(_)组成
- 所有标识符可以包括英文、数字以及下划线，但不能以数字开头
 - 非ASCII码亦可
- 标识符区分大小写
- 以下划线开头的标识符有特殊意义
 - 以单下划线开头代表不能直接访问的类属性
 - 以双下划线开头代表类的私有成员
 - 以双下划线开头和结尾代表 Python 里特殊方法专用的标识

- 变量

- 变量不需要声明

- 每个变量在使用前都必须赋值

- 赋值以后该变量才会被创建

- 变量没有类型

- Python是动态类型语言

- `a, b, c = 1, 2, "runoob"`

- del 语句删除对象引用

- `del var_a, var_b`

- 删除后不能再引用，除非再次赋值

- 变量
 - 获取变量所指对象的内存地址
 - `id(var)`
 - `print(hex(id(var)))`
 - `a=2`
 - `del a`
 - `print(id(2))`
 - `c=2`
 - `print(id(c))`

- 标准数据类型
 - 变量所指的内存中对象的类型
 - Number(int, bool, float, complex) : 数字
 - String(str) : 字符串
 - List(list) : 列表
 - Tuple(tuple) : 元组
 - Set(set) : 集合
 - Dictionary(dict) : 字典

- 类型的划分

- 不可变数据：Number、String、Tuple
- 可变数据：List、Dictionary、Set

- 类型的查询

- `type()` 函数可以用来查询变量所指的对象类型
- `type()` 不会认为子类是一种父类类型

- 类型的判断

- `isinstance(a, int)` 可以用来判断是否是某类型
- `isinstance()` 会认为子类是一种父类类型

- 数字(Number)类型
 - 整数、布尔型、浮点数和复数。
 - `int` (整数), 长整型
 - `bool` (布尔), 如 `True`和`False`.
 - `type(True)`和`isinstance(True, int)`的返回结果?
 - `float` (浮点数), 如3.14
 - `complex` (复数), 如`1+3j`
 - 数据类型不允许改变,这就意味着如果改变数字数据类型的值, 将重新分配内存空间
 - `a=20`
 - `b=20`
 - `id(a)==id(b)` 是否成立?

- 内存分配

- 为提高内存利用效率,对于简单对象如int对象或字符串对象等,会采取重用对象内存的办法

- `x=2`

- `print(id(2))`

- `print(id(x))`

- `x=3`

- `print(id(3))`

- `print(id(x))`

- 数字类型运算

- 常规运算比较简单

- 需要注意的一些地方

- True 和 False 关键字的值是 1 和 0，它们可以和数字相加

- if -1 if 0 if 2 如何判断

- /：除法，得到浮点数
 - //：除法，得到整数（不一定，分子分母为浮点时得到浮点）
 - **：幂
 - complex：a + bj,或者complex(a,b)表示，复数的实部a和虚部b都是浮点型
 - 比较操作可以传递（a<b==c）
 - 洪炜可以哈哈笑了

- 字符串(String)
 - 单引号和双引号使用完全相同
 - 使用三引号(''或''')可以指定一个多行字符串。
 - 转义符 '\'
 - 反斜杠可以用来转义，使用r可以让反斜杠不发生转义
 - 按字面意义级联字符串，如"this " "is "
"string"会被自动转换为this is string
 - 字符串可以用 + 运算符连接在一起，用 * 运算符重复。

- 字符串(String)
 - 字符串有两种索引方式，从左往右以 0 开始，从右往左以 -1 开始。
 - 字符串不能改变
 - `s[0] = 'a'`
 - `s += 'b'`
 - 这时还是不是同样的内存地址？
 - 没有单独的字符类型，一个字符就是长度为 1 的字符串

- 字符串(String)
 - 字符串常用的函数
 - `find()`
 - `strip()`
 - `split()`
 - `zfill(width)`

- 列表(List)

- 使用最频繁的数据类型

- `list1=['a','b','c']`
 - `list2=['d','e','f']`
 - `print(list1*2)`
 - `print(list1+list2)`

- 有步长的元素截取

- `list1[1:4:2]`
 - `n1=[0,1,2,3,4,5,6,7,8,9,10]`
 - `print(n1[0::2])` 将输出？
 - `print(n1[::2])` 将输出？

- 列表(List)

- 逆序

- `l1=['A','B','C','D','E','F','G']`
 - `print(l1[-1::-1])`
 - `print(l1.reverse())`

- 元素的删除

- `del l1[0]`
 - `l1.remove('A')`
 - `a.clear()` 相当于 `del a[:]`
 - `del a`

- 元素的增加

- `a.append(x)` 相当于 `a[len(a):]=[x]`
 - `a.extend(1)` 相当于 `a[len(a):]=1`

- 列表(List)

- 排序

- `a.sort()`

- 复制

- `a.copy()` 相当 `a[:]`

- slice copy , 在可能对原列表有修改的操作中较常见

- 推导式

- `a=[x**2 for x in range(6)]`
 - `a=list(map(lambda x:x**2,range(6)))`
 - `pis=[str(round(pi, i)) for i in range(1, 6)]`
 - `a=[(x, y) for x in [1,2,3] for y in [3,1,4] if x != y]`

- 列表(List)

- 一些复杂但可能有用的操作

- `l1=[[1,3,4],[5,6,7,8],[9,10,11]]`
 - `l1_f=[e for ele in l1 for e in ele]` #flatten list

- 矩阵交换行列

- `matrix=[[1,2,3],[2,3,4],[3,4,5]]`
 - `matrix2=[[row[i] for row in matrix] for i in range(3)]`
 - 怎么按列遍历矩阵？

- 比较大小

- 序列对象可以与其它序列对象比较

- 内存分配

- `L = [1, 2, 3]`

- `L2 = [1, 2, 3]`

- `id(L) == id(L2) ??`

- `L1 = L[:]`

- `id(L1) == id(L) ??`

- 元组 (tuple)
 - 与列表类似，不同之处在于元素不可修改
 - 可以把字符串看作一种特殊的元组
 - 虽然tuple的元素不可改变，但它可以包含可变的对象
 - 给个栗子？
 - 构造包含 0 个或 1 个元素的元组
 - `tup1 = ()` # 空元组
 - `tup2 = (20,)` # 一个元素，需要在元素后添加逗号，否则含义不明确
 - 元组中的元素值不允许删除，但`del`能删除整个元组

- 集合(Set)

- 一般用于进行成员关系测试和删除重复元素
- 可以使用大括号 `{ }` 或者 `set()` 函数创建集合

- `s1={e1,e2,e3}`
- `s2=set(value)`

- 创建空集合必须用 `set()` 而非 `{ }`

- `{ }` 用来创建一个空字典

- 集合的运算

- `-` `|` `&` `^(|-&)`
- `update()` 可实现批量添加元素

- 集合的实现

- 散列实现

- 字典(Dictionary)

- 一种映射类型，其元素是键值对

- **无序**的键(key): 值(value) 集合

- 键(key)必须使用不可变类型

- 同一个字典中键(key)必须是唯一的

- 函数 `dict()` 可以直接从键值对序列中构建字典

- 构造函数

- `dict([('sape', 4139), ('guido', 4127), ('jack', 4098)])`

- `dict(sape=4139, guido=4127, jack=4098)`

- `dict(list(enumerate(['one', 'two', 'three'], start=1))))`

- 散列实现

- 字典(Dictionary)
 - 如何有充地输出？
 - `for key in sorted(dic.keys()):`
 - `print(key, dic[key])`
 - 如何构建有序字典
 - 即对字典排序
 - `import collections`
 - `dic = collections.OrderedDict()`
 - `dic=collections.OrderedDict(sorted(unsorted_d.items(), key=lambda dc:dc[1], reverse = True))`

- 判断相等

- is 运算符

- `a is b` 相当于 `id(a) == id(b)`

- is 用于判断两个变量引用对象是否为同一个

- == 用于判断引用变量的值是否相等

- 强制类型转换

- `int(x [,base])` 将x转换为一个整数
- `float(x)` 将x转换到一个浮点数
- `complex(real [,imag])` 创建一个复数
- `str(x)` 将对象 x 转换为字符串
- `repr(x)` 返回对象 x 的字符串表达
- `eval(str)` 用来计算在字符串中的有效Python表达式,并返回一个对象
- `tuple(s)` 将序列 s 转换为一个元组
- `list(s)` 将序列 s 转换为一个列表

- 强制类型转换

- `set(s)` 转换为可变集合
- `dict(d)` 创建一个字典, `d` 必须是一个 `(key, value)` 元组序列。
- `frozenset(s)` 转换为不可变集合
- `chr(x)` 将一个整数转换为一个字符
- `ord(x)` 将一个字符转换为它的整数值
- `hex(x)` 将一个整数转换为一个十六进制字符串

- 随机数函数

- `choice(seq)` 从序列中随机挑选一个元素
- `randrange([start,] stop [,step])`
从指定范围内，按指定基数递增的集合中获取一个随机数，基数默认值为 1
- `random()` 随机生成下一个实数，在 $[0, 1)$ 范围内
- `seed([x])` 改变随机数生成器的种子seed
- `shuffle(list)` 将序列的所有元素随机排序
- `uniform(x, y)` 随机生成下一个实数，在 $[x, y]$ 范围内

- **Demo**

- 针对一个文本文档，读入并分词，过滤停用词，统计所有词的频率，排序输出前 $topn$ 个词。
- 对所有出现的词依一定规则进行过滤，得到特征词，将文档用特征词表示为向量。
- 可视化为关键词云。
- 待分析的文本文件：`doc1.txt`
- 停用词表：`stopwords_list.txt`
- 代码：`w2demo1.py`

本周作业



- 根据提供的评论数据(online_reviews_texts.txt, 见资源/data, 一行一条评论, 因此一行可以视为一个文档, 行号可以作为文档编号), 读入所有文档并分词, 统计词频, 找到高频词, 确实特征集, 为每一条评论生成向量表示, 计算一下不同评论之间的距离(自定义, 如欧氏或余弦), 能不能找到所有评论的“重心”或者所有评论中的代表性评论并输出原文? 除了词云外, 针对多文档数据还有别的可视化方式没有?