- 2-数据结构大汇总（2）
  - 序列
    - 简介
      - 在 Python 中，序列类型包括字符串、列表、元组、集合和字典，这些序列支持一些通用的操作，但比较特殊的是，集合和字典不支持索引、切片、相加和相乘操作。
    - 1-针对序列的内置函数
      - list(sub) 把一个可迭代对象转换为列表。
      - tuple(sub) 把一个可迭代对象转换为元组。
      - tuple(sub) 把一个可迭代对象转换为元组。
      - len(s) 返回对象（字符、列表、元组等）长度或元素个数。
      - max(sub)返回序列或者参数集合中的最大值
      - min(sub)返回序列或参数集合中的最小值
      - sum(iterable[, start=0]) 返回序列iterable与可选参数start的总和。
      - sorted(iterable, key=None, reverse=False) 对所有可迭代的对象进行排序操作。
        - iterable -- 可迭代对象。
          key -- 主要是用来进行比较的元素，只有一个参数，具体的函数的参数就是取自于可迭代对象中，指定可迭代对象中的一个元素来进行排序。
          reverse -- 排序规则，reverse = True 降序， reverse = False 升序（默认）。
          返回重新排序的列表。
      - reversed(seq) 函数返回一个反转的迭代器。
        - seq -- 要转换的序列，可以是 tuple, string, list 或 range。
      - enumerate(sequence, [start=0])
      - zip(iter1 [,iter2 [...]])
  - 集合
    - 1-集合的创建
      - ·在创建空集合的时候只能使用s = set()，因为s = {}创建的是空字典。
      - ·直接把一堆元素用花括号括起来{元素1, 元素2, ..., 元素n}。
        ·重复元素在set中会被自动被过滤。
      - ·使用set(value)工厂函数，把列表或元组转换成集合。
      - *去掉列表中重复的元素
    - 2-访问集合中的值
      - ·可以使用len()内建函数得到集合的大小
      - ·可以使用for把集合中的数据一个个读取出来。
      - ·可以通过in或not in判断一个元素是否在集合中已经存在
    - 3-集合的内置方法
      - set.add(elmnt)用于给集合添加元素，如果添加的元素在集合中已存在，则不执行任何操作。
      - set.update(set)用于修改当前集合，可以添加新的元素或集合到当前集合中，如果添加的元素在集合中已存在，则该元素只会出现一次，重复的会忽略。
      - set.remove(item) 用于移除集合中的指定元素。如果元素不存在，则会发生错误。
      - set.discard(value) 用于移除指定的集合元素。remove() 方法在移除一个不存在的元素时会发生错误，而 discard() 方法不会。
      - set.pop() 用于随机移除一个元素。
      - 由于 set 是无序和无重复元素的集合，所以两个或多个 set 可以做数学意义上的集合操作
        - ·set.intersection(set1, set2) 返回两个集合的交集。
          ·set1 & set2 返回两个集合的交集。
          ·set.intersection_update(set1, set2) 交集，在原始的集合上移除不重叠的元素。
        - ·set.union(set1, set2) 返回两个集合的并集。
          ·set1 | set2 返回两个集合的并集。
        - ·set.difference(set) 返回集合的差集。
          ·set1 - set2 返回集合的差集。
          ·set.difference_update(set) 集合的差集，直接在原来的集合中移除元素，没有返回值。
        - ·set.symmetric_difference(set)返回集合的异或。
          ·set1 ^ set2 返回集合的异或。
          ·set.symmetric_difference_update(set)移除当前集合中在另外一个指定集合相同的元素，并将另外一个指定集合中不同的元素插入到当前集合中。
        - ·set.issubset(set)判断集合是不是被其他集合包含，如果是则返回 True，否则返回 False。
          ·set1 <= set2 判断集合是不是被其他集合包含，如果是则返回 True，否则返回 False。
        - ·set.issuperset(set)用于判断集合是不是包含其他集合，如果是则返回 True，否则返回 False。
          ·set1 >= set2 判断集合是不是包含其他集合，如果是则返回 True，否则返回 False。
        - ·set.isdisjoint(set) 用于判断两个集合是不是不相交，如果是返回 True，否则返回 False。
    - 4-集合的转换
    - 5-不可变集合
      - Python 提供了不能改变元素的集合的实现版本，即不能增加或删除元素，类型名叫frozenset。需要注意的是frozenset仍然可以进行集合操作，只是不能用带有update的方法。
        - ·frozenset([iterable]) 返回一个冻结的集合，冻结后集合不能再添加或删除任何元素。
  - 字典
    - 1-可变类型与不可变类型
      - ·序列是以连续的整数为索引，与此不同的是，字典以"关键字"为索引，关键字可以是任意不可变类型，通常用字符串或数值。
        ·字典是 Python 唯一的一个 映射类型，字符串、元组、列表属于序列类型。
        ·那么如何快速判断一个数据类型 X 是不是可变类型的呢？两种方法：
        ·麻烦方法：用 id(X) 函数，对 X 进行某种操作，比较操作前后的 id，如果不一样，则 X 不可变，如果一样，则 X 可变。
        ·便捷方法：用 hash(X)，只要不报错，证明 X 可被哈希，即不可变，反过来不可被哈希，即可变。
      - 数值、字符和元组 都能被哈希，因此它们是不可变类型。
        列表、集合、字典不能被哈希，因此它是可变类型。
    - 2-字典的定义
      - 字典 是无序的 键:值（key:value）对集合，键必须是互不相同的（在同一个字典之内）。
        - ·dict 内部存放的顺序和 key 放入的顺序是没有关系的。
          ·dict 查找和插入的速度极快，不会随着 key 的增加而增加，但是需要占用大量的内存。
      - 字典 定义语法为 {元素1, 元素2, ..., 元素n}
        - ·其中每一个元素是一个「键值对」-- 键:值 (key:value)
          ·关键点是「大括号 {}」，「逗号 ,」和「冒号 :」
          ·大括号 -- 把所有元素绑在一起
          ·逗号 -- 将每个键值对分开
          ·冒号 -- 将键和值分开
    - 3-创建和访问字典
      - 通过字符串或数值作为key来创建字典
        - 通过key直接把数据放入字典中，但一个key只能对应一个value，多次对一个key放入value，后面的值会把前面的值冲掉
      - dict() 创建一个空的字典
      - dict(mapping) new dictionary initialized from a mapping object's (key, value) pairs
      - dict(**kwargs) -> new dictionary initialized with the name=value pairs in the keyword argument list. For example: dict(one=1, two=2)
    - 4-字典的内置方法
      - dict.fromkeys(seq[, value]) 用于创建一个新字典，以序列 seq 中元素做字典的键，value 为字典所有键对应的初始值。
      - dict.keys()返回一个可迭代对象，可以使用 list() 来转换为列表，列表为字典中的所有键。
      - dict.values()返回一个迭代器，可以使用 list() 来转换为列表，列表为字典中的所有值。
      - dict.items()以列表返回可遍历的 (键, 值) 元组数组。
      - dict.get(key, default=None) 返回指定键的值，如果值不在字典中返回默认值。
      - dict.setdefault(key, default=None)和get()方法 类似, 如果键不存在于字典中，将会添加键并将值设为默认值。
      - key in dict in 操作符用于判断键是否存在于字典中，如果键在字典 dict 里返回true，否则返回false。而not in操作符刚好相反，如果键在字典 dict 里返回false，否则返回true。
      - dict.pop(key[,default])删除字典给定键 key 所对应的值，返回值为被删除的值。key值必须给出。若key不存在，则返回 default 值。
        del dict[key] 删除字典给定键 key 所对应的值
      - dict.popitem()随机返回并删除字典中的一对键和值，如果字典已经为空，却调用了此方法，就报出KeyError异常。
      - dict.clear()用于删除字典内所有元素。
      - dict.copy()返回一个字典的浅复制。
      - dict.update(dict2)把字典参数 dict2 的 key:value对 更新到字典 dict 里。