组合数据对象-2

- 简单对象和组合对象
- 可变和不变对象
- 不变序列操作
- 可变序列操作和表操作
- 表描述式（推导式，生成式）

简单对象和组合对象

- 程序中处理的对象可分为两类：
  - 简单对象，具有原子性的个体。如整数/浮点数
  - 组合对象（复合对象），由一些成分组成
    - 成分可能是其他类型的对象，成分可能独立存在和使用
    - 表是一种组合对象，各种序列类型的对象都是组合对象
- 组合对象的性质
  - 整体是对象，可以赋值/取值，传入/传出函数
  - 其元素也是对象，可以取出单独使用（或可以修改）
  - 对整体对象的操作一般通过对元素的操作实现
  - 有可能替换元素，或者修改对象的整体结构

组合对象类型：表

- 表 list 是一个类型，其对象是组合对象
- 一个表对象是一个整体
  - 其中包含一些（任意多个，包括0个）元素
  - 表的元素可以是任何对象，一个表里可以有不同类型的元素
  - 作为整体，可以赋给变量，传入/传出函数等
- 对于表的操作，一部分是操作和使用表的元素
  - 元素基本操作是取元素值，元素的重新赋值（修改表的元素）
- 表的结构也可以修改，例如
  - **lst.append(x)** 修改了 lst，增加了一个元素
  - 下面还会介绍另一些操作，其中有些也修改表的结构

所有序列类型（字符串/表等）支持的操作

- **x in s**　　如果 ***s*** 有元素等于 ***x*** 则 **True,** 否则 **False**
- **x not in s**　　如果 ***s*** 有元素等于 ***x*** 则 **False,** 否则 **True**
- **s + t**　　***s*** 和 ***t*** 的拼接
- **s * n 或 n * s**　　***s*** 的 ***n*** 个拷贝的拼接
- **s[i]**　　***s*** 的第 ***i*** 个元素, 从 **0** 开始，**s[i:j]**，**s[i:j:k]** 类似
- **len(s)**　　***s*** 的长度
- **min(s)**　　***s*** 的最小元素
- **max(s)**　　***s*** 的最大元素
- **s.index(x[, i[, j]])**　　***x*** 在 ***s*** 里首次出现的下标 **(**从 ***i*** 开始到 ***j*****)**

  [ ] 表示可选：**s.index(x), s.index(x, i), s.index(x, i, j)**
- **s.count(x)**　　***x*** 在 ***s*** 里出现的总次数

不变类型和不变对象(immutable)

- 一些类型称为不变类型，其对象创建后不会变化（不能修改），也称为不变对象
  - 基本类型的对象都是不变对象，如各种数对象
  - 不变对象的操作只有取得对象的相关信息和生成新对象
- **str** 是一种不变序列类型，其对象（字符串）是不变对象

  作为不变序列，对 **str** 类似的对象（字符串），可以使用上述所有公共序列操作，如 **max("hfuhrewufhu")** 得到其中编码最大的字符（**"w"**）
- **range(m, n, d)** 得到一个 **range** 类型的对象。**range** 是一种元素值为整数的不变序列类型，但它只支持一些序列操作，例如
  - **range(10)[2]** 的值是 **2**
  - **range(100, 200)[3:84:6]** 相当于 **range(103, 184, 6)**

可变对象

- 创建后可以变化（可以修改）的对象称为可变对象
  - 变动：还是这个对象，但它（的内容）变了
  - 造成变动的原因一般是成分被修改（重新赋值），或者是结构（和内容）的改变（例如 **append** 新元素）
- **list** 类型的对象是可变对象
  - 可以修改表元素。例：**lst[0] = 2**
  - 对于可变的序列类型的对象，除了可以使用前面序列对象的公共操作外，还有一组“变动”操作

可变序列操作

- **s[i] = x**　用 ***x*** 取代 ***s*** 里下标 ***i*** 的元素，
- **s[i:j] = t**　用可迭代对象 ***t*** 的内容替代 ***s*** 从 ***i*** 到 ***j*** 的切片；**s[i:j:k] = t** 类似，**i, j, k** 可省略表示默认值，这里要求 **t** 元素个数正合适
- **del s[i]** 删一个元素，**s[i:j]** 相当于 **s[i:j] = []**，**del s[i:j:k]** 类似
- **s.append(x)**　把 ***x*** 接到序列最后 (同 **s[len(s):len(s)] = [x]**)
- **s.clear()** 清除 **s** 的所有元素 (同 **del s[:]**)
- **s.copy()** 创建 **s** 的一个拷贝 (同 **s[:]**)
- **s.extend(t)**　用 ***t*** 的内容扩展 ***s*** (同 **s[len(s):len(s)] = t**)
- **s.insert(i, x)** 把 ***x*** 插入 ***s*** 里由下标 ***i*** 确定的位置 (同 **s[i:i] = [x]**)
- **s.pop(), s.pop(i)** 取 **s** 里下标 ***i*** (默认为最后)的元素并将其从 ***s*** 删除
- **s.remove(x)** 删除 ***s*** 里第一个满足 **s[i] == x** 的元素
- **s.reverse()**　反转 ***s*** 里的所有元素（的位置）

表操作

- 所有序列操作（包括变动操作）都可以用于表
- 对表 lst 使用带下标的操作时，下标不能超出 lst 的范围
  - 对 lst[i] = n，　i 取值范围为 [0, len(lst))
  - 对 lst[i:j] = t，　i, j 取值范围为 [0, len(lst)]

    lst[0:0] = t, lst[len(lst):len(lst)] = t 表示在表头/尾加一段

    所用的 t 必须是一个可迭代对象（序列或迭代器）
- 表只有一个特殊操作
  - lst.sort() 按 < 关系对 lst 的元素排序（修改 lst）
  - lst.sort(reverse=True) 将 lst 按 < 的逆序排序
  - sort 还有一个 key 关键字参数，可以指定一个从元素算出值的函数，要求将元素按这个函数的值排序（见手册）

表操作

下面比较两对可用于表的操作：

- 标准内置排序函数 **sorted** 可用于所有序列或迭代器对象，得到一个元素排序的表，其中是该序列或迭代器的元素
  - **sorted(lst)** 得到表 **lst** 的排序拷贝（另一个新表），**lst** 不变
  - **lst.sort()** 将 **lst** 的内容排序（改变 **lst**，其中元素重新排列）
- 标准内置的序列反转函数 **reversed** 可用于序列对象，得到一个迭代器（不是序列），可以用于 **for** 头部：
  - **reversed(lst)** 从表 **lst** 得到一个反向迭代器，**lst** 不变
  - **lst.reverse()** 把 **lst** 的内容反转，所有元素逐对对调位置
- **lst[:]** 做 **lst** 的拷贝（默认切片是做整个的拷贝，切片的下界缺省值是 0，上界是序列长度），**list(lst)** 的结果一样

表描述式

- 用表描述式（list comprehension，内涵描述）构造

  基本形式 [表达式 for 变量 in 可迭代对象]

  变量由可迭代对象取值，对该变量的每个取值计算表达式，以得到的值作为表的一个元素

  注意：序列对象和迭代器都可以用做可迭代对象

  带条件形式[表达式 for 变量 in可迭代对象 if 条件表达式]

  选择满足条件表达式要求的变量取值，去计算表元素

  在基本形式之后可以有任意多个 for 片段或 if 片段

- 用描述式生成表的实例
- 通过表描述式，可以直接生成比较复杂的表

  应该学会使用，对于其他序列对象，也有类似形式的描述式