Python程序设计 基本数据类型、常量和变量

苏州大学计算机科学与技术学院

问题引入

- A=1
- A=3.14
- A='China'
- A=[1,2,3,4]
- A={'NAME': 'TOM', 'SEX':'MALE', 'AGE':19}



Python是一种强数据类型的编程语言!

基本的Python数据类型

- 数字
- 字符串
- 列表
- 元组
- 字典
- 集合

数字-整型(int)

- 2. X中区分标准整型(int)和长整型(long),但3.x中就不再区分了,可以表示任意大的数值
- >>> print(a, type(a))
- 支持不同的进制表示 0x0A 0o10 0b1010都表示十进制的10

数字-布尔型(bool)

- 布尔类型有两种取值: True和False
- 对于一些非布尔类型的数值在需要使用布尔值的时候,值为0的数字以及空集(字符串、列表、元组、集合等)都将被理解为False

```
>>> bool(1)
True
>>> bool('a')
True
>>> bool(0)
False
>>> bool('')
False
```

数字-浮点型(float)

- Python的浮点型不区分单精度和双精度
- Python的浮点型数据在内存中占据24个字节

```
>>> import sys
>>> a=1.0
>>> sys.getsizeof(a)
24
```

数字-复数型(complex)

```
• Python支持复数类型
>>> a = 1+1j
>>> a
         (1+1j)
>>> type(a)
         <class 'complex'>
>>> import sys
>>> sys.getsizeof(a)
         32
>>> a = complex(2.2)
>>> a
         (2.2+0j)
>>> sys.getsizeof(a)
>>> a.real
         2.2
>>> a.imag
```

字符串 (str)

- 字符串是不可变数据类型,就是说改变一个字符串的元素需要新建一个新的字符串。
- 字符串的元素也是字符串

```
>>> s='abcd'
```

>>> type(s)

<class 'str'>

>>> type(s[0])

<class 'str'>

- Python使用单、双和三引号来作为字符串的标示,Python没有字符(char)这一数据类型
- 字符串支持索引、切片、长度、遍历、删除、分割、清除空白、大小写转换、判断以什么开头等操作,详细内容请参见相应教材
- Python的字符串支持正则表达式的功能,这是进行文本搜索等操作的重要基础

序列(列表、元组、字典、集合)

- 序列是一块用来存放多个值的连续内存空间。常见的序列有列表、元组、字典和集合。
- 可变与不可变特性
 - 可变: 列表、字典和集合
 - 不可变: 元组
- 有序与无序特性
 - 有序: 列表、元组和字典
 - 无序: 集合
- 上述序列结构中并不一定要求所有元素都是相同类型的
- 可以利用集合来完成去重处理
- 可以利用字典的哈希特性来完成大数据量的查找和统计的速度
- 列表推导式的功能要重点贯彻

序列(列表、元组、字典、集合)

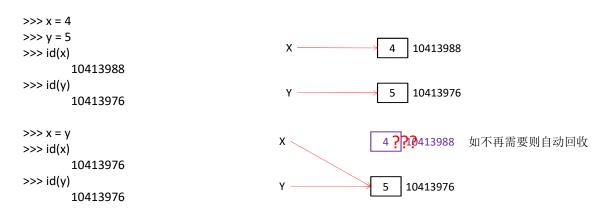
- 序列是一块用来存放多个值的连续内存空间。常见的序列有列表、元组、字典和集合。
- 可变与不可变特性
 - 可变: 列表、字典和集合
 - 不可变: 元组
- 有序与无序特性
 - 有序: 列表、元组和字典
 - 无序: 集合
- 上述序列结构中并不一定要求所有元素都是相同类型的
- 可以利用集合来完成去重处理
- 可以利用字典的哈希特性来完成大数据量的查找和统计的速度
- 列表推导式的功能要重点贯彻

数据类型的可变与不可变特性

- 不可变数据类型: 当该数据类型的对应变量的值发生了改变,那么它对应的内存地址也会发生改变,对于这种数据类型,就称不可变数据类型。
- 可变数据类型 : 当该数据类型的对应变量的值发生了改变,那么它对应的内存地址不发生改变,对于这种数据类型,就称可变数据类型。
- 不可变数据类型:
 - 数字
 - 字符串
 - 元组
- 可变数据类型
 - 列表
 - 字典
 - 集合

Python中变量的内存空间管理机制

• Python变量定义的过程相对于是对一个常量存储空间的引用关系



Python中变量的内存空间管理机制

• Python的可变数据类型变量在使用过程中的特殊现象

>>> x=[1,2,3] >>> x=[1,2,3] >>> y=[1,2,3] >>> y=x >>> id(x) >>> id(x) 2996505893704 2996505893704 >>> id(y) >>> id(y) 2996505894216 2996505893704 >>> x[0]=10 >>> x[0]=11 >>> x >>> y [10, 2, 3][11, 2, 3] >>> y >>> y [1, 2, 3][11, 2, 3]



让我们大展Python身手吧!

