
# 内置数据类型1

北京理工大学计算机学院　高玉金

2019年3月

# Python简介

- 从“小”数据到大数据，从简单数据到复杂数据
- Python是一门弱类型的面向对象的解释语言
- Python语言简单易学，功能强大，是一门适合用于大数据和人工智能的语言
- 安装和使用Python3.X
  - 交互环境IDLE
  - 文件编辑环境
  - 可以使用自己喜欢的编辑器，如Sublime等
  - 解释执行：python sample.py
- 建议安装Aanconda，并使用spyder编辑器

```
In [9]: a = 5

In [10]: type(a)
Out[10]: int

In [11]: a=5.0

In [12]: type(a)
Out[12]: float

In [13]: a="5"

In [14]: type(a)
Out[14]: str

In [15]: a
Out[15]: '5'
```

# Python内置类型对象int

- 整数类型的四种进制表示
  - 十进制，不需要引导符，0-9
  - 二进制，0b或0B，0-1
  - 八进制，0o或0O，0-7
  - 十六进制，0x或0X，0-9到A-F
- 普通用户无需区分有符号和无符号数
- 整数类型理论上没有取值范围限制，与C语言不同
- 支持算术运算

# Python内置类型对象float

- 所有浮点数必须带小数部分，小数部分可以为零或省略，如5.0或5.
- 浮点数和整数在计算机内部表示不同，如0.0和0，没有double类型
- 浮点数表示方法
  - 十进制，如3.1415
  - 科学计数法，如4.3e-5
- Python浮点数的取值范围和小数精度受不同计算机系统的限制
- 可以用sys.float_info列出所用计算机系统的各项参数，一般为15个数字的准确度
- Python支持无限制且准确的整数计算，建议使用整数而非浮点数
- 支持算术运算

# Python内置类型对象complex

- 表示数学中的复数概念
- 实部+虚部，表示为：a+bj
- 实部和虚部均为浮点数
- 对于复数z，可以用z.real和z.imag获取实部和虚部

```
>>> 1.0 + 3.3j.imag
4.3
>>> (1.0+3.3j).imag
3.3
```

```
>>> type(1.+3.j)
<class 'complex'>
```

# Python内置类型对象bool

- python 中布尔值使用常量True 和 False来表示，注意大小写
- 用于逻辑判定，比较，常用在if和while语句中
- bool是int的子类（继承自int）， True==1 False==0 返回Ture
- 下列表达式均会被判定为False
  - None
  - False
  - 任意数字类型的0，如 0，0.0，0j等
  - 任意空的序列结构，如‘’， ()，[]，空字典等
  - 用户自定义类中给出了 __bool__() 或 __len__() 方法，并返回整数0或布尔值False

```
>>> type(0)
<class 'int'>
>>> type(False)
<class 'bool'>
>>> type(0 == False)
<class 'bool'>
```

# Python内置类型对象str

- 字符串对象
  - 单引号，双引号，三引号
- 字符串和其他类型的区别之一 ：长度差异很大
- Python内部以Unicode编码存储字符串，Unicode是真正的字符串，而用ASCII、UTF-8、GBK等字符编码表示的是字节串
- 可以用len(s)计算出字符串的长度

```
In [20]: len("北京理工大学")
Out[20]: 6
```

```
In [25]: s="北京理工大学"

In [26]: g= s.encode("gbk")
    ...:

In [27]: g
Out[27]: b'\xb1\xb1\xbe\xa9\xc0\xed
\xb9\xa4\xb4\xf3\xd1\xa7'

In [28]: s
Out[28]: '北京理工大学'

In [29]: len(g)
Out[29]: 12

In [30]: len(s)
Out[30]: 6
```

# 查看类型对象

- dir查看对象类型，了解可用方法，用help来获取帮助
- 判断字符是否为数字：ch.isdigit()

```
>>> help(str.upper)
Help on method_descriptor:

upper(...)
    S.upper() -> str

    Return a copy of S converted to uppercase.
```

```
>>> dir(str)
['__add__', '__class__', '__contains__', '__delattr__', '__
dir__', '__doc__', '__eq__', '__format__', '__ge__', '__get
attribute__', '__getitem__', '__getnewargs__', '__gt__', '_
_hash__', '__init__', '__init_subclass__', '__iter__', '__l
e__', '__len__', '__lt__', '__mod__', '__mul__', '__ne__',
'__new__', '__reduce__', '__reduce_ex__', '__repr__', '__rm
od__', '__rmul__', '__setattr__', '__sizeof__', '__str__',
'__subclasshook__', 'capitalize', 'casefold', 'center', 'co
unt', 'encode', 'endswith', 'expandtabs', 'find', 'format',
'format_map', 'index', 'isalnum', 'isalpha', 'isdecimal',
'isdigit', 'isidentifier', 'islower', 'isnumeric', 'isprinta
ble', 'isspace', 'istitle', 'isupper', 'join', 'ljust', 'lo
wer', 'lstrip', 'maketrans', 'partition', 'replace', 'rfind
', 'rindex', 'rjust', 'rpartition', 'rsplit', 'rstrip', 'sp
lit', 'splitlines', 'startswith', 'strip', 'swapcase', 'tit
le', 'translate', 'upper', 'zfill']
```

# 数据类型与输入输出

- input()：从键盘输入的任何信息，都被转换为字符串
- eval()：去引号，计算Python表达式，返回运算结果（对应的对象）
- type()：查看对应的数据类型

```
s=input("请输入：")
print(type(s))
```

```
请输入：4.5
<class 'str'>
```

**eval("4:5")**

```
>>> a=eval(input("请输入："))
请输入：wer
Traceback (most recent call last):
  File "<pyshell#29>", line 1, in <module>
    a=eval(input("请输入："))
  File "<string>", line 1, in <module>
NameError: name 'wer' is not defined
```

# 数据类型与输入输出

```
>>> s=input("请输入：")
请输入：3+4
>>> s
'3+4'
```

```
>>> s=eval(input("请输入："))
请输入：3+4
>>> s
7
```

```
>>> b,c=eval(input("请输入："))
请输入：4,5
>>> b,c
(4, 5)
>>> b+c
9
```

# 强制类型转换

- int()
- float()
- str()
- complex()
- chr()支持ASCII码和Unicode
- ord()取得ASCII /Unicode值
- hex()
- oct()

```
>>> oct(9)
'0o11'
>>> hex(9)
'0x9'
```

```
>>> ord("高")
39640
```

```
>>> ord('9')
57
>>> chr(59)
';'
```

```
>>> chr(9999)
'✏'
```

```
>>> bool(9)
True
>>> int('3')
3
>>> str(9)
'9'
>>> complex(9)
(9+0j)
>>> float(3)
3.0
>>> int(True)
1
```