# 3.基本数据类型

## 3.1.数值类型

### 3.1.1.数值类型的组成

**整数型——不同进制的转换**

默认输入十进制

二进制0b、八进制0o、十六进制0x

16 == 0b10000 == 0o20 == 0x10

True

十进制与其他进制的转换

a = bin(16) *# 转二进制*

b = oct(16) *# 转八进制*

c = hex(16) *# 转十六进制*

print(a, b, c)

0b10000 0o20 0x10

*# 注意：上述转换后结果为字符串类型*

type(a)

str

*# 其他进制转十进制*

d = int(a, 2) *# 二进制转十进制*

e = int(b, 8) *# 八进制转十进制*

f = int(c, 16) *# 十六进制转十进制*

print(d, e, f)

16 16 16

**浮点数——不确定性**

不确定小数问题

(0.1+0.2) == 0.3

False

0.1+0.2

0.30000000000000004

计算机采用二进制小数来表示浮点数的小数部分

部分小数不能用二进制小数完全表示

通常情况下不会影响计算精度

四舍五入获得精确解

a = 3\*0.1

print(a)

0.30000000000000004

b = round(a, 1)

print(b)

b == 0.3

0.3

True

**复数——a+bj**

大写J或小写j均可

例如：3+4j 2+5J

虚部系数为1时，需要显式写出

例如：2+1j

### 3.1.2.运算符

加减乘除运算   +  -  /  \*

例如：(1+3-4\*2)/5 # -0.8

取反  -

例如：x = 1 -x # -1

乘方运算  \*\*

例如：2\*\*3 # 8

整数商//  和  模运算%

例如：

13//5 # 整数商 x/y 向下取整数 2

13 % 5 # 模运算 余数 13=2\*5+3

注意：

整数与浮点数运算结果是浮点数

除法运算的结果是浮点数

### 3.1.3.数字运算操作函数

求绝对值 abs()

例如：

abs(-5) # 5

abs(3+4j) # 5，对于复数，求的是模，即

幂次方 pow(x,n)

例如：pow(2, 5) # pow(x,n) x的n次方 等价于x\*\*n 32

四舍五入 round(x,n)

a = 1.618

print(round(a)) *# 默认四舍五入为整数*

2

print(round(a, 2)) *# 参数2表示四舍五入后保留2位小数*

1.62

print(round(a, 5)) *# 位数不足，无需补齐*

1.618

整数商和模运算 divmod(x,y)

等价于返回二元元组（x//y,x % y）

divmod(13, 5) *# 较（x//y,x % y）更快，只执行了一次x/y*

(2, 3)

序列最大/最小值 max( )  min( )

max(3, 2, 3, 6, 9, 4, 5)

9

a = [3, 2, 3, 6, 9, 4, 5]

print("max:", max(a))

print("min:", min(a))

max: 9

min: 2

求和sum(x)

sum((1, 2, 3, 4, 5))

15

借助科学计算库 math\scipy\numpy

**import** **math** *# 导入库*

print(math.exp(1)) *# 指数运算 e^x*

print(math.log2(2)) *# 对数运算*

print(math.sqrt(4)) *# 开平方运算 等价于4^0.5*

2.718281828459045

1.0

2.0

**import** **numpy** **as** **np**

a = [1, 2, 3, 4, 5]

print(np.mean(a)) *# 求均值*

print(np.median(a)) *# 求中位数*

print(np.std(a)) *# 求标准差*

3.0

3.0

1.4142135623730951

## 3.2.字符串类型

### 3.2.1.字符串的表达

用双引号""或单引号''括起来的任意字符

print("Python")

print('Python')

Python

Python

字符串中有双引号或单引号的情况

双中有单

print("I'm 18 years old")

I'm 18 years old

单中有双

print('"Python" is good')

"Python" is good

双中有双，单中有单——转义符  \

*# print(""Python" is good")*

print("**\"**Python**\"** is good") *# \ 我是个字符呀*

"Python" is good

转义符可以用来换行继续输入

*# 等等，我还没完事！*

s = "py**\**

thon"

print(s)

python

### 3.2.2.字符串的性质

字符串的索引

通过字符的位置对字符进行访问

格式：变量名[位置编号]

正向索引，从零开始递增

反向索引——从-1开始递减

位置编号不能超过字符串的长度

s = "My name is Tom"

print(s[0])

print(s[2])

print(s[5])

M

m

print(s[-1])

print(s[-3])

print(s[-5])

m

T

s

字符串的切片

索引只能获得一个字符，如何获得多个字符？

可以对字符串进行切片

格式：变量名[开始位置：结束位置：切片间隔]

切片间隔如不设置默认为1，可省略

切片范围含头不含尾

s = "Python"

print(s[0:3:1])

Pyt

print(s[0:3])

Pyt

print(s[0:3:2])

Pt

起始位置是0 可以省略

结束位置省略，代表可以取到最后一个字符

可以使用反向索引

s = "Python"

print(s[0:6])

Python

print(s[:6])

Python

print(s[:])

Python

print(s[-6:])

Python

反向切片

起始位置是-1也可以省略

结束位置省略，代表可以取到第一个字符

s = "123456789"

print(s[-1:-10:-1])

987654321

print(s[:-10:-1])

987654321

print(s[::-1])

987654321

### 3.2.3.字符串操作符

字符串的拼接

字符串1+字符串2

a = "I love "

b = "my wife "

a+b

'I love my wife '

字符串的成倍复制

字符串 \* n   n \* 字符串

c = a+b

print(c\*3)

print(3\*c)

I love my wife I love my wife I love my wife

I love my wife I love my wife I love my wife

成员运算

子集in全集   任何一个连续的切片都是原字符串的子集

folk\_singers = "Peter, Tom and Mary"

"Peter" **in** folk\_singers

True

"PPM" **in** folk\_singers

False

遍历字符串字符   for 字符 in 字符串

**for** s **in** "Python":

print(s)

P

y

t

h

o

n

### 3.2.4.字符串处理函数

字符串的长度

所含字符的个数

s = "python"

len(s)

6

字符编码

将中文字符，英文字母、数字、特殊字符等转化成计算机可识别的二进制数

每个单一字符对应一个唯一的互不重复的二进制编码

Python 中使用的是Unicode编码

将字符转化为Unicode码——ord(字符)

print(ord("1"))

print(ord("a"))

print(ord("\*"))

print(ord("中"))

print(ord("国"))

49

97

42

20013

22269

将Unicode码转化为字符——chr(Unicode码)

print(chr(1010))

print(chr(10000))

print(chr(12345))

print(chr(23456))

ϲ

✐

〹

宠

### 3.2.5.字符串的处理方法

字符串的分割——字符串.split(分割字符)

返回一个列表

原字符串不变

languages = "Python C C++ Java PHP R"

languages\_list = languages.split(" ")

print(languages\_list)

print(languages)

['Python', 'C', 'C++', 'Java', 'PHP', 'R']

Python C C++ Java PHP R

字符串的聚合——“聚合字符”.join(可迭代数据类型)

可迭代类型 如：字符串、列表

s = "12345"

s\_join = ",".join(s)

s\_join

'1,2,3,4,5'

序列类型的元素必须是字符类型

*# s = [1, 2, 3, 4, 5]*

s = ["1", "2", "3", "4", "5"]

"\*".join(s)

'1\*2\*3\*4\*5'

删除两端特定字符——字符串.strip(删除字符)

strip从两侧开始搜索，遇到指定字符执行删除，遇到非指定字符，搜索停止

类似的还有左删除lstrip和右删除rstrip

s = " I have many blanks "

print(s.strip(" ")) *# 还有吗？hahaha*

print(s.lstrip(" "))

print(s.rstrip(" "))

print(s)

I have many blanks

I have many blanks

I have many blanks

I have many blanks

字符串的替换——字符串.replace("被替换"，"替换成")

s = "Python is coming"

s1 = s.replace("Python","Py")

print(s1)

Py is coming

字符串统计——字符串.count("待统计字符串")

统计字符出现的频率

s = "Python is an excellent language"

print("an:", s.count("an"))

print("e:", s.count("e"))

an: 2

e: 4

字符串字母大小写

字符串.upper() 字母全部大写

s = "Python"

s.upper()

'PYTHON'

字符串.lower() 字母全部小写

print(s.lower())

print(s)

python

Python

字符串.title()首字母大写

将字符串中每个单词的首字母大写

s.title()

'Python'

## 3.3.布尔类型

布尔类型只有两个值，分别为True和False

表示逻辑运算的结果

a = 10

print(a > 8)

print(a == 12)

print(a < 5)

True

False

False

any()和all()

any()表示出现的元素只要有非零则为True

all()表示出现的元素只要有零则为False

print(any([**False**,1,0,**None**])) *# 0 False None 都是零*

print(all([**False**,1,0,**None**]))

True

False

指示条件

True和break配合使用的猜数字游戏

n = 2800

**while** **True**:

m = eval(input("请输入一个正整数："))

**if** m == n:

print("你猜对啦")

**break**

**elif** m > n:

print("太大了")

**else**:

print("太小了")

请输入一个正整数：28

太小了

请输入一个正整数：2800

你猜对啦

作为掩码

**import** **numpy** **as** **np**

x = np.array([[1, 3, 2, 5, 7]]) *# 定义 numpy数组*

print(x > 3)

x[x > 3]

[[False False False True True]]

array([5, 7])

## 3.4.类型判别及类型转换

### 3.4.1.类型判别

判断类型

type(变量)

age = 20

name = "Ada"

print(type(age))

print(type(name))

<class 'int'>

<class 'str'>

isinstance(变量,预判类型) 判断变量是否属于预判类型

变量类型是预判类型的子类型，则为真，否则为假

print(isinstance(age, int)) *# 承认继承*

True

print(isinstance(age, object))

print(isinstance(name, object)) *# object 是所有对象的父类*

True

True

字符串检查方法

字符串.isdigit() 字符是否只有数字组成

age = "20"

name = "Ada"

age.isdigit()

True

name.isdigit()

False

字符串.isalpha()字符是否只有字母组成

name.isalpha()

True

age.isalpha()

False

字符串.isalnum()字符是否只有数字和字母组成

"Ada20".isalnum() *# 比如可用于判断用户名是否合法*

True

### 3.4.2.类型转换

数字类型转字符串  str(数字类型)

age = 20

print("My age is "+str(age))

My age is 20

仅有数字组成的字符串转数字  int()  float()  eval()

s1 = "20"

s2 = "10.1"

int(s1) *# 仅整型*

*# int(s2)*

20

float(s1)

20.0

float(s2)

10.1

eval(s1)

20

eval(s2)

10.1

## 3.5.作业练习

填空题：

1、下列代码的输出结果是\_\_\_\_\_\_\_。

x=20

y=7

print(x%y，x\*\*2)

2、下列代码的输出结果分别是\_\_\_\_\_\_\_和\_\_\_\_\_\_\_。

x=15

y=6

print(x/y，x//y)

print(divmod（x，y))

3、下列代码的输出结果是\_\_\_\_\_\_\_。

x=3.1415926

print(round(x，2)，round(x))

操作题

s="python is a good language"

4、分别用正向索引和负向索引从s中获取字符串“python”；

5、将s首字母大写；

6、统计s中字母o出现的次数；

7、将s中的“python”替换为“C++”；

8、将s以“ ”（空格）进行划分，输出新的字符串序列；

9、将8中生产的字符串序列用“\*”聚合成新的字符串。

10、将数字255分别转换成二进制、八进制和十六进制，然后反向转回十进制的形式。

判断以下输出结果为True or False：

a=10

b=5

c="6"

d="nine"

e="peppa pig123"

11、(a>10)and(b<8)

12、(a>10)or(b<8)

13、isinstance(a，float)

14、c.isdigit()

15、d.isalpha)

16、e.isalnum()

答案：

1. 6 400
2. 2.5 2 和 (2, 3)
3. 3.14 3
4. s[0:6] s[:-(len(s)-6:1]
5. 不能使用s.title，因为会将每个单词首字母大写，str(s[0:1].upper())+str(s[1::])
6. s.count(“o”)
7. s.replace(“python”,”C++”)
8. s.split(“ “)
9. “\*”.join(s.split(“ “))
10. bin(255) oct(255) hex(255), int(hex(255),16)...
11. False
12. True
13. False
14. True
15. True
16. False