Python面向对象语法精讲

嵩 天

Python类的运算

嵩 天

Python类的运算

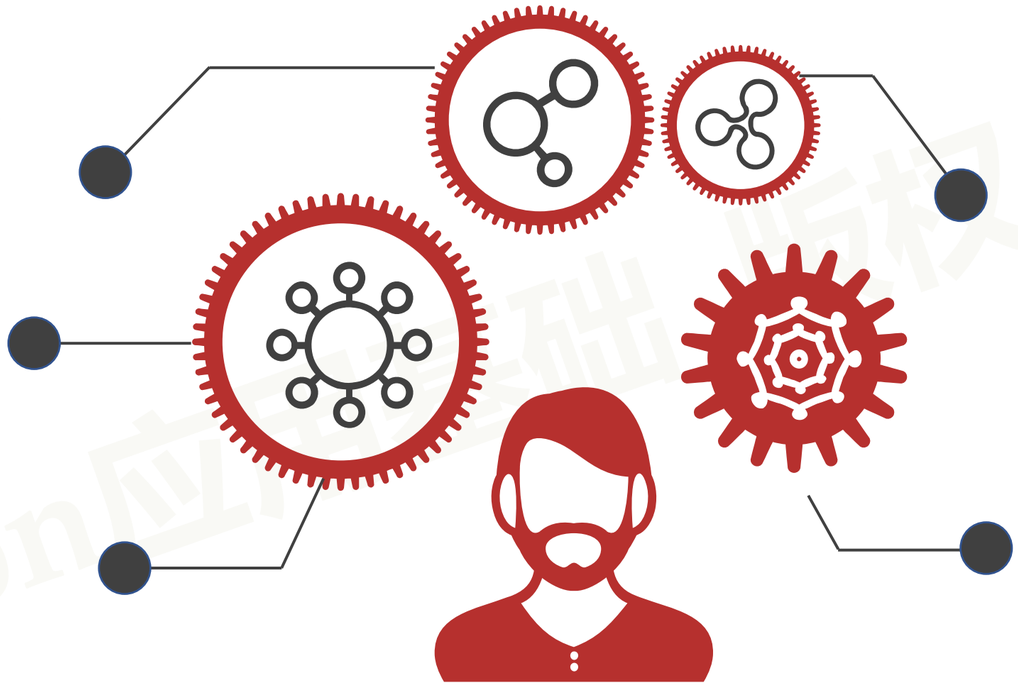
单元开篇

单元开篇

(3) 比较运算的重载

(2) 算术运算的重载

(1) 运算的理解



(4) 成员运算的重载

(5) 其他运算的重载

Python类的运算

Python类的运算

运算的理解

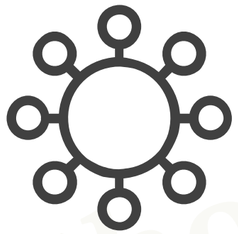
运算的理解

运算 Operation：操作逻辑的抽象

- 运算体现一种操作逻辑，广义角度，任何程序都被认为是运算
- Python解释器通过保留方法预留了一批运算的接口，需要重载
- 保留方法一般对应运算符，Python中运算体现为运算符的重载

运算的理解

运算 Operation：操作逻辑的抽象



类(Class)

- 算术运算 → 体现 交互关系
- 比较运算 → 体现 包含关系
- 成员运算 → 体现 包含关系
- 其他运算 → 体现 常规关系

运算的理解

运算重载的限制

- 不能重载Python语言内置类型的运算符
- 不能新建运算符，只能通过重载完成
- is and not or 不能被重载

Python类的运算

算术运算的重载

算术运算的重载

算术运算的种类

- 一元算术运算符：+、-、~
- 二元算术运算符：+、-、*、/、//、%、divmod()、pow()、**

  <<、>>、&、^、|

运算的重载

算术运算符的重载：一元算术运算符

保留方法	对应操作	描述
.__neg__(self)	-obj	定义对象取负的运算逻辑
.__pos__(self)	+obj	定义对象取正的运算逻辑
.__abs__(self)	abs(obj)	定义对象绝对值的运算逻辑
.__invert__(self)	~obj	定义对象取反的运算逻辑

运算的重载

算术运算符的重载：二元算术运算符 (1)

保留方法	对应操作	描述
.__add__(self, other)	obj + other	定义两个对象加法的运算逻辑
.__sub__(self, other)	obj - other	定义两个对象减法的运算逻辑
.__mul__(self, other)	obj * other	定义两个对象乘法的运算逻辑
.__truediv__(self, other)	obj / other	定义两个对象除法的运算逻辑

运算的重载

算术运算符的重载：二元算术运算符 (2)

保留方法	对应操作	描述
.__floordiv__(self, other)	obj // other	定义两个对象整数除的运算逻辑
.__mod__(self, other)	obj % other	定义两个对象模的运算逻辑
.__divmod__(self, other)	divmod(obj, other)	定义两个对象除模的运算逻辑

运算的重载

算术运算符的重载：二元算术运算符 (3)

保留方法	对应操作	描述
.__pow__(self, other)	obj ** other	定义对象幂的运算逻辑
.__lshift__(self, other)	obj << other	定义对象左移的运算逻辑
.__rshift__(self, other)	obj >> other	定义对象右移的运算逻辑

运算的重载

算术运算符的重载：二元算术运算符 (4)

保留方法	对应操作	描述
.__and__(self, other)	obj & other	定义两个对象位与的运算逻辑
.__xor__(self, other)	obj ^ other	定义两个对象位异或的运算逻辑
.__or__(self, other)	obj & other	定义两个对象位或的运算逻辑

运算的重载

```
class NewList(list):
    def __add__(self, other):
        result = []
        for i in range(len(self)):
            try:
                result.append(self[i] + other[i])
            except:
                result.append(self[i])
        return result

ls = NewList([1,2,3,4,5,6])
lt = NewList([1,2,3,4])
print(ls + lt)
```

继承list类型的新类型

重载其中的加法运算

运算的重载

```
class NewList(list):
    def __add__(self, other):
        result = []
        for i in range(len(self)):
            try:
                result.append(self[i] + other[i])
            except:
                result.append(self[i])
        return result

ls = NewList([1,2,3,4,5,6])
lt = NewList([1,2,3,4])
print(ls + lt)
```

对两个对象进行加运算

运算的重载

```
class NewList(list):
    def __add__(self, other):
        result = []
        for i in range(len(self)):
            try:
                result.append(self[i] + other[i])
            except:
                result.append(self[i])
        return result

ls = NewList([1,2,3,4,5,6])
lt = NewList([1,2,3,4])
print(ls + lt)
```

```
[2, 4, 6, 8, 5, 6]
```

小结：算术运算的重载

算术运算的种类

- 一元算术运算符：+、-、~
- 二元算术运算符：+、-、*、/、//、%、divmod()、pow()、**

  <<、>>、&、^、|

Python类的运算

比较运算的重载

比较运算的重载

比较运算的种类

- 比较运算：<、<=、==、!=、>、>=

运算的重载

比较运算的重载

保留方法	对应操作	描述
.__lt__(self, other) .__le__(self, other) .__eq__(self, other) .__ne__(self, other) .__gt__(self, other) .__ge__(self, other)	obj < other obj <= other obj == other obj != other obj > other obj >= other	两个对象比较操作的运算重载

运算的重载

```
class NewList(list):
    def __lt__(self, other):
        "以各元素算术和为比较依据"
        s, t = 0, 0
        for c in self:
            s += c
        for c in other:
            t += c
        return True if s < t else False

ls = NewList([6,1,2,3])
lt = NewList([1,2,3,99])
print([6,1,2,3] < [1,2,3,99])
print(ls < lt)
```

继承list类型的新类型

重载其中的小于比较运算

运算的重载

```
class NewList(list):
    def __lt__(self, other):
        "以各元素算术和为比较依据"
        s, t = 0, 0
        for c in self:
            s += c
        for c in other:
            t += c
        return True if s < t else False

ls = NewList([6,1,2,3])
lt = NewList([1,2,3,99])
print([6,1,2,3] < [1,2,3,99])
print(ls < lt)
```

对两个对象进行小于运算

同时比较列表的小于运算

运算的重载

```
class NewList(list):
    def __lt__(self, other):
        "以各元素算术和为比较依据"
        s, t = 0, 0
        for c in self:
            s += c
        for c in other:
            t += c
        return True if s < t else False

ls = NewList([6,1,2,3])
lt = NewList([1,2,3,99])
print([6,1,2,3] < [1,2,3,99])
print(ls < lt)
```

```
False
True
```

小结：比较运算的重载

比较运算的种类

- 比较运算：<、<=、==、!=、>、>=

Python类的运算

成员运算的重载

成员运算的重载

成员运算的种类

- 成员获取：[]、del、.reversed()
- 成员判断：in、not in

运算的重载

成员运算符的重载：成员获取

保留方法	对应操作	描述
.__getitem__(self, key)	obj[k]	定义获取对象中序号k元素的运算逻辑，k为整数
.__setitem__(self, key, v)	obj[k] = v	定义赋值对象中序号k元素的运算逻辑
.__delitem__(self, key)	del obj[k]	定义删除对象中序号k元素的运算逻辑
.__reversed__(self)	obj.reversed()	定义对象逆序的运算逻辑

运算的重载

成员运算符的重载：成员判断

保留方法	对应操作	描述
.__contains__(self, item)	item in obj	**定义in操作符对应的运算逻辑**

运算的重载

```
class NewList(list):
    def __contains__(self, item):
        "各元素算术和也作为成员"
        s = 0
        for c in self:
            s += c
        if super().__contains__(item) or item == s:
            return True
        else:
            return False

ls = NewList([6,1,2,3])
print(6 in ls, 12 in ls)
```

继承list类型的新类型

重载其中的成员判断运算

运算的重载

```
class NewList(list):
    def __contains__(self, item):
        "各元素算术和也作为成员"
        s = 0
        for c in self:
            s += c
        if super().__contains__(item) or item == s:
            return True
        else:
            return False

ls = NewList([6,1,2,3])
print(6 in ls, 12 in ls)
```

对2个元素进行成员判断

运算的重载

```
class NewList(list):
    def __contains__(self, item):
        "各元素算术和也作为成员"
        s = 0
        for c in self:
            s += c
        if super().__contains__(item) or item == s:
            return True
        else:
            return False

ls = NewList([6,1,2,3])
print(6 in ls, 12 in ls)
```

```
True True
```

小结：成员运算的重载

成员运算的种类

- 成员获取：[]、del、.reversed()
- 成员判断：in、not in

Python类的运算

其他运算的重载

其他运算的重载

其他运算的种类

- Python内置函数：repr()、str()、len()、int()、float()、complex()、round()、bytes()、bool()、format()
- 类的常用方法：.format()

运算的重载

其他运算符的重载（1）

保留方法	对应操作	描述
.__repr__(self)	repr(obj)	定义对象可打印字符串的运算逻辑
.__str__(self)	str(obj)	定义对象字符串转换操作的运算逻辑
.__len__(self)	len(obj)	定义对象长度操作的运算逻辑

运算的重载

其他运算符的重载（2）

保留方法	对应操作	描述
.__int__(self)	int(obj)	定义对象整数转换的运算逻辑
.__float__(self)	float(obj)	定义对象浮点数转换的运算逻辑
.__complex__(self)	complex(obj)	定义对象复数转换的运算逻辑

运算的重载

其他运算符的重载（3）

保留方法	对应操作	描述
.__round__(self)	round(obj)	定义对象四舍五入的运算逻辑
.__bytes__(self)	bytes(obj)	定义对象字节串转换的运算逻辑
.__bool__()	bool(obj)	定义对象布尔运算的运算逻辑

运算的重载

其他运算符的重载（4）

保留方法	对应操作	描述
.__format__(self, format_spec)	obj.format() format(obj)	**定义对象格式化输出的运算逻辑**

运算的重载

```
class NewList(list):
    def __format__(self, format_spec):
        "格式化输出，以逗号分隔"
        t = []
        for c in self:
            if type(c) == type("字符串"):
                t.append(c)
            else:
                t.append(str(c))
        return ", ".join(t)

ls = NewList([1,2,3,4])
print(format([1,2,3,4]))
print(format(ls))
```

继承list类型的新类型

重载其中的格式化运算

运算的重载

```
class NewList(list):
    def __format__(self, format_spec):
        "格式化输出，以逗号分隔"
        t = []
        for c in self:
            if type(c) == type("字符串"):
                t.append(c)
            else:
                t.append(str(c))
        return ", ".join(t)

ls = NewList([1,2,3,4])
print(format([1,2,3,4]))
print(format(ls))
```

格式化输出

并与列表类型比较

运算的重载

```python
class NewList(list):
    def __format__(self, format_spec):
        "格式化输出，以逗号分隔"
        t = []
        for c in self:
            if type(c) == type("字符串"):
                t.append(c)
            else:
                t.append(str(c))
        return ", ".join(t)

ls = NewList([1,2,3,4])
print(format([1,2,3,4]))
print(format(ls))
```

```
[1, 2, 3, 4]
1, 2, 3, 4
```

小结：其他运算的重载

其他运算的种类

- Python内置函数：repr()、str()、len()、int()、float()、complex()、round()、bytes()、bool()、format()
- 类的常用方法：.format()

Python类的运算

单元小结

单元小结

(3) 比较运算的重载

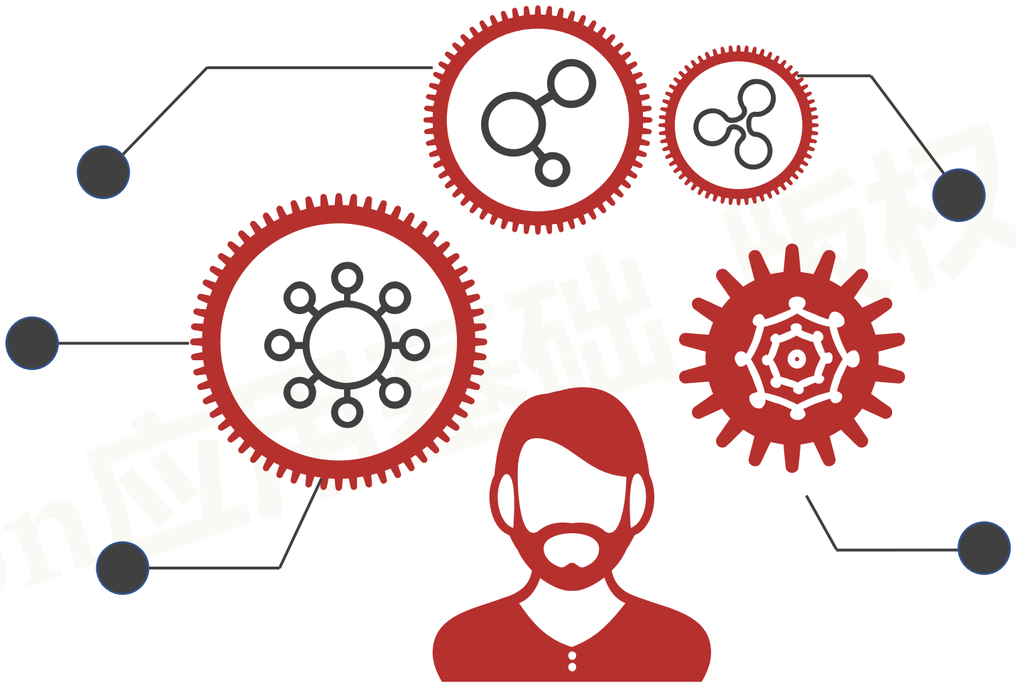
<、<=、==、!=、>、>=

(2) 算术运算的重载

一元/二元算术运算符

(1) 运算的理解

运算的种类、运算符



(4) 成员运算的重载

成员获取/判断操作符

(5) 其他运算的重载

内置函数相关的运算符

Python类的运算


Python 123
Thank you