

Day4-基本运算符与注释的多种方法

一、 学习目标

掌握 Python 中常用的各类基本运算符，能够熟练运用多种注释方法，能够理解并实践良好的代码注释规范。

二、 知识点讲解

1. Python 基本运算符

运算符是用于对变量或值执行特定操作的符号。Python 提供了丰富的运算符类型。

① 算术运算符：用于执行基本的数学计算。如下表所示。

+	加法	<code>a + b</code>	13	
-	减法	<code>a - b</code>	7	
*	乘法	<code>a * b</code>	30	
/	除法	<code>a / b</code>	3.333...	浮点除法，结果总是为浮点数。
//	整除	<code>a // b</code>	3	整数除法，结果向下取整。
%	取余	<code>a % b</code>	1	返回除法的余数。
**	幂运算	<code>a ** b</code>	1000	计算 a 的 b 次方 (10 ³)。

表 1：算术运算符

② 比较运算符：用于比较两个值，返回布尔值 True 或 False。如下表所示。

<code>==</code>	等于	<code>a == b</code>	<code>False</code>	判断值是否相等。
<code>!=</code>	不等于	<code>a != b</code>	<code>True</code>	判断值是否不等。
<code>></code>	大于	<code>a > b</code>	<code>True</code>	
<code><</code>	小于	<code>a < b</code>	<code>False</code>	
<code>>=</code>	大于等于	<code>a >= 5</code>	<code>True</code>	
<code><=</code>	小于等于	<code>a <= 3</code>	<code>False</code>	

表 2：比较运算符

③ 赋值运算符：用于给变量赋值，可以与算术运算结合，简化代码。如下表所示。

<code>=</code>	<code>x = 5</code>	<code>x = 5</code>	基本赋值。
<code>+=</code>	<code>x += 3</code>	<code>x = x + 3</code>	加后赋值。
<code>-=</code>	<code>x -= 2</code>	<code>x = x - 2</code>	减后赋值。
<code>*=</code>	<code>x *= 4</code>	<code>x = x * 4</code>	乘后赋值。

<code>/=</code>	<code>x /= 2</code>	<code>x = x / 2</code>	除后赋值（结果为浮点数）。
<code>//=</code>	<code>x //= 3</code>	<code>x = x // 3</code>	整除后赋值。
<code>%=</code>	<code>x %= 2</code>	<code>x = x % 2</code>	取余后赋值。
<code>**=</code>	<code>x **= 3</code>	<code>x = x ** 3</code>	幂运算后赋值。

表 3：赋值运算符

④ 逻辑运算符：用于组合多个条件判断，返回布尔值。如下表所示。

<code>and</code>	与	<code>x > 0 and x < 10</code>	全真才真	两边条件都为 True 时，结果才为 True。
<code>or</code>	或	<code>x < 0 or x > 100</code>	一真即真	两边条件任意一个为 True，结果即为 True。
<code>not</code>	非	<code>not (x == 5)</code>	取反	将条件的结果取反。

表 4：逻辑运算符

⑤ 成员运算符：检查一个值是否存在于一个序列（如字符串、列表、元组）中。如下表所示。

in	是否包含	'a' in 'apple'	True	如果值在序列中，返回 True。
not in	是否不包 含	'x' not in [1, 2, 3]	True	如果值不在序列中，返回 True。

表 5：成员运算符

⑥ 身份运算符：检查两个变量是否指向内存中的同一个对象。如下表所示。

is	是同一个对象	a is b	如果 a 和 b 是同一个对象（内存地址相同）， 返回 True。
is not	不是同一个对 象	a is not b	如果 a 和 b 不是同一个对象，返回 True。

表 6：身份运算符

注意：is 与 == 不同。== 比较的是值是否相等，而 is 比较的是身份（内存地址）是否相同。

2. Python 注释的多种方法

注释是代码中不会被程序运行的说明性文字，其核心目的是提高代码的可读性和可维护性。

① 单行注释（#）

```
# 这是一个单行注释
x = 10 # 变量 x 赋值为 10
# print("这行被注释了，不会执行")

#用途：解释变量、函数、逻辑等。
```

图 1：单行注释

② 多行注释 (' ' ' 或 " " ")

```
"""
这是一个多行注释。
可以写很多行。
用于说明复杂逻辑或函数用途。
"""

'''
也可以这样写注释。
常用于模块或函数说明（文档字符串 docstring）。
'''
```

图 2：多行注释

③ 快速注释/取消注释代码块：

1. 先选中多行代码，
2. 按下快捷键：Windows/Linux: Ctrl + /, Mac: Cmd + /。
3. 编辑器会自动在每行前面加上 # 号。再次按此快捷键可以取消注释。
4. 解释“为什么”，而不是“做什么”，代码本身已经说明了“做什么”，注释应解释背后的设计意图、逻辑原因或复杂算法。

三、 作业安排

任务描述：

假设你是一名老师，需要计算一个学生的最终成绩。

作业平均分 (homework_avg) = 85

期中考试分 (midterm) = 78

期末考试分 (final) = 92

成绩计算规则：最终成绩 = 作业平均分 * 30% + 期中考试分 * 30% + 期末考试分 * 40%

请编写一个 Python 程序，计算该学生的最终成绩（保留一位小数），并判断他是否通过了考试（最终成绩 ≥ 60 ）。

要求：

1. 使用变量存储上述分数。
2. 使用算术运算符 (*, +, /) 计算最终成绩。
3. 使用比较运算符 (\geq) 判断是否通过。
4. 使用 print() 函数输出结果。

最后，在代码中添加适当的注释，描述每一步操作的功能。

四、 总结

今天我们系统学习了 Python 编程中的基本运算符和代码注释。

基本运算符是我们进行数据处理和逻辑判断的工具：

算术运算符：能进行加减乘除等数学计算；

比较运算符：比较数值大小或相等性，返回 `True` 或 `False`；

赋值运算符：给变量赋值，特别是复合赋值运算符（如 `+=`）能简化代码；

逻辑运算符：用 `and`、`or`、`not` 组合多个条件；

成员运算符：用 `in` 和 `not in` 检查元素是否存在于序列中；

身份运算符：用 `is` 和 `is not` 检查两个变量是否指向同一个对象。

代码注释是编写高质量代码的关键。我们学习了：

使用 `#` 进行单行注释。

使用 `""" ... """` 或 `''' ... '''` 编写多行注释。

利用编辑器快捷键 `Ctrl + /`、`Cmd + /` 快速注释/取消注释代码块。