让数字正确参与计算与判断

第 2.5-2.7 节｜数字类型、类型转换、运算符与综合实训

Python 零基础培训

90 分钟课堂

今天把“数字”变成可靠的业务结果

确认表示和类型，选择正确运算，用括号写清口径，再用边界样例验证。

- 认识整数、浮点数和布尔值
- 完成显式转换，区分 / // %
- 使用比较、逻辑和成员运算
- 完成时间间隔与 BMI 计算

外观相近的内容，类型和运算可能不同

```
10          # 整数 int
10.0        # 浮点数 float
"10"        # 字符串 str
```

`10 == 10.0` 为 **True**；`10 == "10"` 为 **False**。

显示内容只能提供线索，类型决定可用运算。

浮点数适合一般小数计算，但存在存储近似

```
result = 0.1 + 0.2
print(result)            # 0.30000000000000004
print(result == 0.3)     # False
```

- 展示金额：格式化或 round
- 比较近似结果：math.isclose
- 精确金额需求：整数“分”或 Decimal

布尔值：判断条件的结果只有真或假

bool

```
bool(0)         # False
bool("")        # False
bool("False") # True
```

比较表达式产生 True/False；**非空文本”False” 仍然是真值**。

显式转换之前，先确认文本是否符合数字格式

```
quantity = int("42")
price = float("19.9")

int(3.9)        # 3：截去小数，不是四舍五入
float("19.9") # 19.9：文本符合数字格式
```

`float("abc")` 会触发 `ValueError`；错误信息应指出哪个输入不合格。

算术运算中，最容易混淆的是三种“除”

表达式	含义	17 与 5
`17 / 5`	普通除法	3.4
`17 // 5`	向下取整除法	3
`17 % 5`	余数	2
`17 ** 5`	幂	1419857

`125 // 60 = 2` 小时；`125 % 60 = 5` 分钟。

复合赋值表达“基于当前状态继续更新”

```
stock = 100
stock -= 8
stock += 20

a, b = "A", "B"
a, b = b, a
```

- `stock -= 8` 等价于 `stock = stock - 8`
- 复合赋值前，左侧变量必须已存在
- `/=` 通常会得到浮点结果

比较运算的关键在边界口径

表达式	100 是否入选	适用口径
`quantity > 100`	否	严格大于 100
`quantity >= 100`	是	至少 100、含 100
`quantity == 100`	是	恰好等于 100
`quantity != 100`	否	不等于 100

用 **99**、**100**、**101** 三个样例测试阈值。

逻辑运算把多个清楚条件组合成业务规则

```
is_target = (
    quantity >= 100
    and unit_price < 200
    and not is_discontinued
)
```

- and：所有条件成立
- or：至少一个条件成立
- not：对条件结果取反

成员运算回答“它是否属于这组值”

```
allowed_channels = {" 直播", " 短视频", " 搜索"}
channel = " 直播"
channel in allowed_channels

order = {"product": " 双肩包", "quantity": 3}
"product" in order           # 检查键
" 双肩包" in order.values()  # 检查值
```

优先级可以记，但业务公式更应该主动加括号

```
net_sales = (
    sales_quantity - return_quantity
) * unit_price
```

常用顺序：括号 → 幂 → 乘除 → 加减 → 比较 → not → and → or。

括号既控制运算，也向读者声明业务口径。

实训一：时间先统一成分钟，再拆回小时与分钟

```
def interval(sh, sm, eh, em):
    start = sh * 60 + sm
    end = eh * 60 + em
    minutes = (end - start) % (24 * 60)
    return minutes // 60, minutes % 60

interval(17, 10, 18, 15)  # (1, 5)
```

取模让 23:50 → 00:20 也能得到 30 分钟。

实训二：BMI 公式同时练习转换、幂与除法

```
def calculate_bmi(weight_kg, height_m):
    if height_m <= 0:
        raise ValueError(" 身高必须大于 0")
    return weight_kg / (height_m ** 2)

print(f"{calculate_bmi(60, 1.70):.2f}")
```

本练习只演示公式计算，不构成医学诊断。

综合微案例：把运算符连成业务口径

```
net_quantity = sales_quantity - return_quantity
net_sales = net_quantity * unit_price
low_return = return_quantity / sales_quantity <= 0.10

is_priority = (
    net_sales >= 20_000
    and low_return
    and channel in allowed_channels
)
```

先打印每个中间结果，再检查最终布尔条件。

离场检查：八个高频坑，至少能解释六个

- `/` `//` `%` 混淆
- 数字文本未转换
- `int(3.9)` 不是四舍五入
- `bool("False")` 为真
- `=` 与 `==` 混用
- `>` 与 `>=` 边界不清
- 多步公式缺括号
- 分母可能为 0

任务 解释 `125 // 60` 与 `125 % 60` 如何组成“2 小时 5 分钟”。

可靠计算来自四步：确认类型、选择运算、写清口径、测试边界。

下一步：打开 课后练习.py，只修改 TODO。