第二次课

知识点一：注释

学习目标：

注释的作用

单行注释（行注释）

多行注释（块注释）

1.1 注释作用

使用用自己熟悉的语言，在程序中对某些代码进行**标注说明**，增强程序的可读性。

1.2 单行注释

以 # 开头，# 右边的所有东西都被当做说明文字，而不是真正要执行的程序，只起到辅助说明作用。注意：为了保证代码的可读性，# 后面建议**先添加一个空格**，然后再编写相应的说明文。

示例代码如下：

```
# 这是第一个单行注释
print("hello python")
```

在代码后面增加的单行注释在程序开发时，同样可以使用 # 在代码的后面（旁边）增加说明性的文字，但是，需要注意的是，为了保证代码的可读性，**注释和代码**之间 至少要**有 两个空格**

示例代码如下：

```
print("hello python")   # 输出 `hello python`
```

1.3 多行注释（块注释）

如果希望编写的注释信息很多，一行无法显示，就可以使用多行注释，要在 Python 程序中使用多行注释，可以用**一对连续**的 **三个 引号**(单引号和双引号都可以)。

示例代码如下：

```
"""

这是一个多行注释

在多行注释之间，可以写很多很多的内容……

"""

print("hello python")
```

1.4 什么时候需要使用注释？

1. 注释不是越多越好，对于一目了然的代码，不需要添加注释。

2. 对于 复杂的操作，应该在操作开始前写上若干行注释。

3. 对于 不是一目了然的代码，应在其行尾添加注释（为了提高可读性，注释应该至少离开代码 2 个空格）

4. 绝不要描述代码，假设阅读代码的人比你更懂 Python，他只是不知道你的代码要做什么。

知识点二：算术运算符

目标：

算术运算符的基本使用。

2.1 算术运算符

算数运算符是 运算符的一种，是完成基本的算术运算使用的符号，用来处理四则运算。

运算符	描述	实例
+	加	10 + 20 = 30
-	减	10 - 20 = -10
*	乘	10 * 20 = 200
/	除	10 / 20 = 0.5
//	取整除	返回除法的整数部分（商） 9 // 2 输出结果 4
%	取余数	返回除法的余数 9 % 2 = 1

运算符	描述	实例
**	幂	又称次方、乘方，2 ** 3 = 8

注意：在 Python 中 * 运算符还可以用于字符串，计算结果是字符串重复指定次数的结果

```
In [1]: "-" * 50

Out[1]: '--------------------------------------------------'
```

2.2 算术运算符优先级

和数学中的运算符的优先级一致，在 Python 中进行数学计算时，同样也是：

⊙先乘除后加减

⊙同级运算符是 从左至右 计算

⊙可以使用 () 调整计算的优先级

以表格的算数优先级由高到最低顺序排列

运算符	描述
**	幂 (最高优先级)
* / % //	乘、除、取余数、取整除
+ -	加法、减法

知识点三：变量

程序就是用来处理数据的，而变量就是用来存储数据的。

目标

* 变量定义
* 变量的类型
* 变量的命名

3.1 变量的定义

在 Python 中，每个变量在使用前都**必须赋值**，变量赋值以后该变量才会被创建。等号（=）用来给变量赋值。

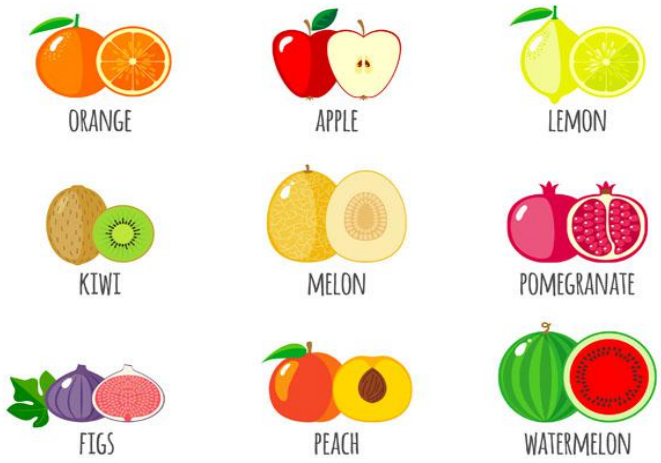
= 左边是一个变量名

= 右边是存储在变量中的值

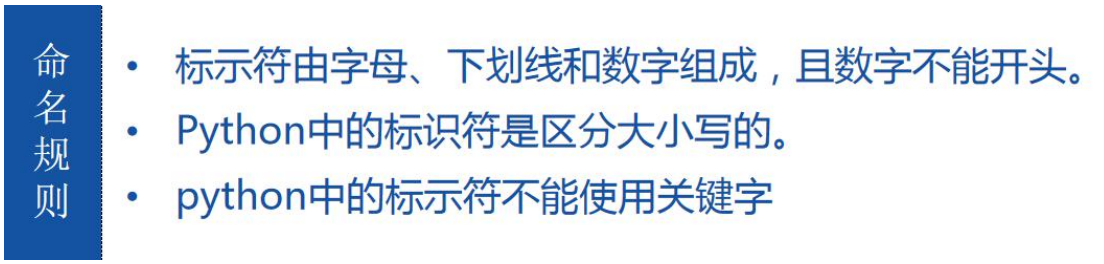
```
变量名 = 值

# 定义 qq 号码变量

qq_number = "1234567"

# 定义 qq 密码变量

qq_password = "123"

# 在程序中，如果要输出变量的内容，需要使用 print 函数

print(qq_number)

print(qq_password)
```

补充知识点：标示符和关键字

现实生活中，人们常用一些名称来标记事物，例如，现实生活中每种水果都有一个名称来标识。



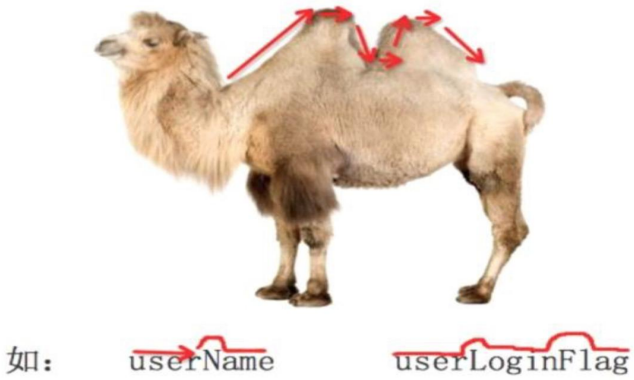
程序中表示一些事物，需要开发人员自定义一些符号和名称，这些符号和名称叫做**标识符**。



为了规范命名标识符，关于标识符的命名提以下建议。

1. 见名之意

2. 不建议使用驼峰式



关键字指的是具有**特殊功能**的**标识符**

如 **False True、def、if、 return、and、not、or、with、while、for** 等等。已经被占用了的 不可以再被定义成标示符。

3.2 变量演练

目的：可以用 其他变量的计算结果 来定义变量

变量定义之后，后续就可以直接使用了。

案例：

1 苹果的价格是 8.5 元/斤

2 买了 7.5 斤 苹果

3 计算付款金额

```
# 定义苹果价格变量

price = 8.5

# 定义购买重量

weight = 7.5

# 计算金额

money = price * weight

print(money)
```

思考题

如果 **只要买苹果，就返 5 块钱**

请重新计算购买金额

```
# 定义苹果价格变量

price = 8.5

# 定义购买重量

weight = 7.5

# 计算金额

money = price * weight

# 只要买苹果就返 5 元

money = money - 5

print(money)
```

提问

1. **上述代码中，一共定义有几个变量**？

   三个：price / weight / money

2. **money = money - 5 是在定义新的变量还是在使用变量**？

   直接使用之前已经定义的变量

   变量名 只有在 第一次出现 才是 定义变量

   变量名 再次出现，不是定义变量，而是直接使用之前定义过的变量

3. **在程序开发中，可以修改之前定义变量中保存的值吗**？

   可以

   变量中存储的值，就是可以 **变** 的

3.3 变量的类型

在内存中创建一个变量，会包括：

i. 变量的名称
ii. 变量保存的数据
iii. 变量存储数据的类型
iv. 变量的地址（标示）

案例：个人信息

- 定义变量保存小明的个人信息
- 姓名：**小明**
- 年龄：**18** 岁
- 性别：**是**男生
- 身高：**1.75** 米
- 体重：**75.0** 公斤

提问

1. 在演练中，一共有几种数据类型？
    1. 4 种
    2. `str`——— 字符串
    3. `bool`——— 布尔（真假）
    4. `int`——— 整数
    5. `float`——— 浮点数（小数）
2. 在 `Python` 中定义变量时需要指定类型吗？

    1. 不需要
    2. `Python` 可以根据=等号右侧的值，自动推导出变量中存储数据的类型在

Python 中定义变量是 **不需要指定类型**（在其他很多高级语言中都需要），数据类型可以分为 **数字型** 和 **非数字型**

数字型

- 整型 (`int`)
- 浮点型（`float`）
- 布尔型（`bool`）
    - 真 `True` 非 `0` 数——**非零即真**
    - 假 `False0`
- 复数型 (`complex`)
    - 主要用于科学计算，例如：平面场问题、波动问题、电感电容等问题

非数字型

- 字符串
- 列表
- 元组
- 字典

使用 type 函数可以查看一个变量的类型。

```
In [1]: type(name)
```