C++ 变量与常量 讲义

一、 为什么我们需要变量?

例 1-8：存钱问题（参考书本 P9 1.3 节）
Uim 的银行账户里原有 100 元。经过了以下操作：

1. 往里面存了 10 元；
2. 购物花掉了 20 元；
3. 把里面的钱全部取出。
   需求：在每次操作后输出账户余额。

分析： 这个问题无法用一个简单的算式一次性解决。我们需要在每一步操作后，把算出来的**中间结果**保存下来，留给下一步继续计算。在编程中，用来**存储计算中间结果**的工具，就叫做**变量** 。

二、 计算机里的容器

在生活中，我们用**容器**（如碗、盘子）来存放物品。在计算机中，我们用**变量**来存放数据。

做菜的比喻（参考书本 P15 第一自然段）：

- **原材料** = 我们要处理的数据（数字、字符等）。
- **容器** = 变量（用来装数据）。
- **操作台** = 计算机的内存。

内存的物理形态：
计算机的数据存放于内存中。内存就像一排排的小抽屉，这样的小抽屉就是一个**字节 (Byte)**，它是计算机中最小的存储单位。不同的数据类型（比如一张照片和一个单纯的数字），占用的抽屉数量是不一样的。

三、 声明变量：挑个柜子写上自己的名字

如果你想占用内存里的“抽屉”，必须先向计算机声明，否则别人（其他程序）可能也会去用这个抽屉，导致数据乱套。

声明变量的格式： `数据类型 + 变量名;`

1. 数据类型（选多大的柜子）

参考书本 P16 表 2-1
做菜时，装高汤不能用菜篓子。编程时，不同的数据也要选不同类型的柜子：

分类	类型名称	关键字	说明 / 存储范围
整型 (整数)	短整型	`short`	-32768 ~ 32767
	基本整型	`int`	**-2147483648 ~ 2147483647 (最常用)**
	长整型	`long long`	存非常大的整数
浮点型 (小数)	单精度型	`float`	精确到小数点后 6~7 位

分类	类型名称	关键字	说明 / 存储范围
	双精度型	`double`	**精确到小数点后 15~16 位 (最常用)**

2. 变量名命名规则（贴什么名字）

参考书本 P11 第三自然段
变量名必须做到**见名知意**（具有自解释性，比如用 `balance` 表示余额，用 `age` 表示年龄）。

- **规则：** 只能由字母、数字和下划线（`_`）组成。并且数字**绝对不能**出现在第一个位置上！（`num1` 正确，`1num` 错误）。
- **避开关键字：** 不能使用 C++ 自带的指令词（如 `include`，`return`，`int`）。
- **不可重复：** 同一作用域（范围）内，名字不能重复，你不能命名两个柜子都叫做`pingguo`。
- **语言限制：** 英文不好可以用拼音（如 `pingguo`），但**严禁使用中文**！

四、 赋值与初始化：往柜子里装东西

声明了柜子后，我们如何把数据放进去呢？这就需要用到**赋值符 (`=`)**。
注意： 在 C++ 中，`=` 不叫“等于”，它叫“赋值”。它的动作是：**将等号右边的数值，粘贴（覆盖）给左边的变量。**

1. 变量的初始化

初始化指的是**在定义变量的同时就给它赋值**，避免柜子里残留以前的垃圾数据。比方说之前有人往这个柜子里放了化学试剂，如果你直接往里放食材的话，肯定就会有危险！

```
int num; // 未初始化，如果直接去读取里面的数据，会读取到随机的垃圾数字
int a = 10; // 安全：定义整数 a，并初始化为 10
double b = 3.123; // 安全：定义小数 b，并初始化为 3.123
```

2. 连续定义与赋值（补充知识，请记在书本 P15 页，下节课讲解）

如果我们需要同时申请多个同类型的变量，可以用逗号 `,` 连写：

```
int a, b; // 同时申请两个空柜子 a 和 b
int x = 10, y = 20; // 同时申请两个柜子 x 和 y 并装入初始值
```

3. 赋值的覆盖特性与累加

对变量的赋值是一个**覆盖**过程。新数据装进去，旧数据就彻底消失了。

变量的自我更新：

```
int num1 = 1;
num1 = num1 + 7; // 此时 num1 变成了 8
```