C++ 格式化输入与输出（`printf` 与 `scanf`）讲义

课前引入：为什么要学格式化输入输出？

【书本 P21 页 例 2-7】数字反转问题

> **题目描述：** 输入一个不小于 100 且小于 1000，同时包括小数点后一位的浮点数（例如 `123.4`），要求把这个数字翻转过来，变成 `4.321` 并输出。

如果只用我们之前学过的 `cin` 和 `cout`，处理带有小数点的数字反转会非常麻烦。但如果我们掌握了 C++ 里的另外两件工具——`printf` 和 `scanf`，这道题只需两行代码即可轻松破解！

今天，我们就来学习这两个工具。但在使用它们之前，请务必在代码最上方添加头文件：`#include <cstdio>`。

第一部分：`printf`

1. 核心概念与语法

`printf` 的全称是 **Print Formatted（格式化打印）**。
它的核心思想是：**先写好一段带有“空位”的模板句子，然后再把变量按顺序填入这些空位中。**

语法格式：

> `printf("包含格式说明符（占位符）的字符串", 变量1, 变量2, ...);`

2. 基础用法：从打印文本到打印变量

第一步：简单打印文本
最基本的形式是在括号里直接放入你想输出的字符串：

```
#include <cstdio>
#include <iostream>
using namespace std;

int main() {
    // 打印字符串并换行 (\n)
    printf("Hello, World!\n");
    return 0;
}
```

第二步：打印变量（使用格式说明符）
`printf` 的强大之处在于可以将变量的值插入到字符串中。这需要用到**格式说明符（占位符）**，它们以 `%` 开头。

```
int age = 30;
// %d 是用于打印整数的格式说明符
printf("我的年龄是%d岁。\n", age);
```

逻辑解释：

1. `"我的年龄是%d岁。\n"` 是输出的模板字符串。
2. `%d` 是一个占位符。
3. `printf` 会在输出时，自动按照顺序用后面 `age` 变量的值（30）去替换 `%d` 的位置。

第三步：混合输出多个变量

`printf` 也可以极其方便地在一句话中同时输出多个、多种类型的变量，而不需要像 `cin` 或 `cout` 那样反复使用 `<<` 拼接。

```
int age = 15;
double height = 168.5;
char rank = 'S';

// 依次使用 %d, %lf, %c 作为占位符
printf("我今年%d岁，身高是%lf厘米，综合评级：%c\n", age, height, rank);
```

逻辑解释：

1. **一一对应：** 字符串里有 3 个坑（`%d`, `%lf`, `%c`），后面就必须跟着 3 个变量（`age`, `height`, `rank`）。
2. **按顺序入座：** `printf` 会严格按照从左到右的顺序，把第一个变量填入第一个占位符，第二个变量填入第二个占位符，以此类推。
3. **类型必须匹配：** 第一个是整数 `age`，所以必须用 `%d`；第二个是小数 `height`，必须用 `%lf`；第三个是字符 `rank`，必须用 `%c`。

3. 常用格式说明符（书本 P24 页）

> **核心规则：** 占位符的顺序必须与后面变量的顺序一一对应，且数据类型必须兼容！

占位符	对应数据类型	说明
`%d`	`int`	读取或打印整数
`%f`	`float`	读取或打印单精度小数
`%lf`	`double`	读取或打印双精度小数
`%c`	`char`	读取或打印单个字符

4. `cout` 与 `printf` 写法对比

操作目标	`cout` 写法	`printf` 写法
打印纯文本	`cout << "打印的内容";`	`printf("打印的内容");`
打印单个变量	`cout << num1;`	`printf("%d", num1);`
混合打印多个变量	`cout << num1 << "和" << num2;`	`printf("%d和%d", num1, num2);`

第二部分：`scanf`

1. 核心概念与语法

`scanf` 的全称是 **Scan Formatted（格式化扫描/输入）**。
它的逻辑和 `printf` 非常类似：根据提供的格式模板，去键盘输入流中精准抓取对应的数据。

语法格式：

> `scanf("格式控制字符串", &变量1, &变量2, ...);`

2. 为什么必须加 `&`（取地址符）？

请注意，`scanf` 括号里的变量名前面，多了一个符号 `&`（读作"取地址符"）。

原理解析：
还记得我们说"变量是在内存里申请的抽屉"吗？
以前的 `cin` 很聪明，你告诉它抽屉的名字，它就能自己找到。但是 `scanf` 是一位老派且严格的员工，你光告诉它抽屉的名字是不够的，你必须把这个抽屉在内存中的**具体物理坐标（地址）**交给它！
在变量名前加上 `&`，就相当于提取了这个抽屉的坐标。

> **警告：** 如果忘记写 `&`，程序在运行到输入阶段时会直接崩溃！

基础示例：

```
int age;
printf("请输入您的年龄：");
scanf("%d", &age); // 获取一个整数，存入 age 所在的内存地址
```

3. 一次性读取多个数据

当需要接收多个数据时，可以连着写占位符：

```
int num;
char letter;
// 接收一个整数和一个字符
scanf("%d%c", &num, &letter);
```

注：在输入多个数字时，中间可以用空格、Tab 键或回车键隔开，`scanf` 会自动跳过空白符抓取数据。

4. `scanf` 使用的易错点：不要乱加标点符号

错误示例： `scanf("%d,%d", &a, &b);`
如果这样写，用户在键盘输入时，**必须严格输入逗号**（如 `5,10`）。如果用户输入了空格（如 `5 10`），`scanf` 发现没有逗号，会立刻停止工作，导致变量 `b` 无法获取数据。

正确做法： 保持双引号里干净，除了 `%d` 等占位符，**不要加任何多余的标点或文字**（除非题目明确要求使用特殊符号作为分割符）。写成 `scanf("%d%d", &a, &b);` 是最安全的。

5. `cin` 与 `scanf` 写法对比

操作目标	`cin` 写法	`scanf` 写法
读取单个变量	`cin >> a;`	`scanf("%d", &a);`
连续读取多个变量	`cin >> a >> b;`	`scanf("%d%d", &a, &b);`

第三部分：实战演练

实战 1：数字反转（书本 P21 页 例 2-7）

解题思路： 对于 `123.4`，我们不把它当成数字，而是当成“3 个字符 + 1 个小数点 + 1 个字符”。利用 `scanf` 去分别获取四个变量！

```
#include <cstdio>

int main() {
    char a, b, c, d;
    // 输入格式为 abc.d，精准匹配分离出 4 个字符
    scanf("%c%c%c.%c", &a, &b, &c, &d);

    // 按照 d.cba 的倒序格式输出
    printf("%c.%c%c%c\n", d, c, b, a);
    return 0;
}
```

实战 2：三角形面积（保留两位小数）

解题思路： 使用 `printf` 的格式控制。`%.2f` 表示将浮点数精确保留 2 位小数输出。

```
#include <iostream>
#include <cstdio>
using namespace std;

int main() {
    float h, b, s;
    cin >> h >> b; // 输入高和底
    s = h * b / 2; // 计算面积

    // %.2f 限制输出格式保留两位小数
    printf("%.2f\n", s);
    return 0;
}
```

实战 3：再分肥宅水（书本 P23 页 例 2-8）

题目要求： t 毫升水均分给 n 名同学。每人需 2 个杯子。输出每人获得多少毫升（严格保留 3 位小数）以及一共需要多少个杯子。

```
#include <iostream>
#include <cstdio>
using namespace std;

int main() {
    // 1. 准备好变量
    double t;
    int n;

    // 2. 通过 scanf 输入数据（注意必须加 & 符号！）
    scanf("%lf%d", &t, &n);

    // 3. 计算并输出
    // t/n 对应 %.3f (保留三位小数)，n*2 对应 %d，中间用 \n 换行
    printf("%.3f\n%d\n", t / n, n * 2);

    return 0;
}
```