# **第一章：认识C++**

## 引言

**同学们，想象一下，如果我们面对一张白纸，需要构建一座复杂的建筑，却不知道从何开始。在编程的世界里，我们经常需要从基础开始，一步步构建起复杂的程序。基础语法就是编程世界的砖石，它们是构建每一个程序的起点。**

**在编程中，我们经常需要执行一些基础但至关重要的任务，比如定义数据的存储方式、控制程序的流程、以及执行基本的数学运算。这些任务如果手动执行，将非常耗时且容易出错。幸运的是，编程语言提供了一套基础语法，它们是构建程序的规则和工具，允许我们以一种清晰、高效的方式表达我们的意图。**

**在本章节中，我们将一起探索编程基础语法的奥秘。我们将学习如何使用变量来存储数据，如何通过运算符进行计算，以及如何利用控制结构来决定程序的执行流程。通过丰富的示例和练习，你将掌握编程的基本概念和应用技巧，为你的编程技能打下坚实的基础。**

**准备好了吗？让我们一起开启基础语法的学习之旅，让这些基础工具成为我们构建程序的坚实基石，让我们的程序更加清晰和高效！**

## 课程目标

* **理解机器学习的概念和重要性。**
* **掌握编程基础知识，包括变量、内存、流程图和主函数。**
* **通过实例教学，学会编写简单算术运算程序解决实际问题。**
* **了解 C ++中 cin 和 cout 的使用方法和 C语言中 printf 和 scanf理解标准输入输出的概念。**

### 第1小节-基础语法

#### 【★】引入自动化概念

**同学们，大家想过吗？我们每天在网上购物时，点击下单后，商品会自动送到我们家。这一切的背后，实际上都是通过编程来完成的，计算机根据我们编写的代码自动执行任务。今天，我们就要学习如何通过编程让计算机完成任务！**

**编程就像是给计算机写下一份任务清单，告诉它该做什么。我们需要使用一些基本的编程概念来实现这些任务，比如变量，它们就像是计算机的‘记事本’，用来存储我们需要的数据。**

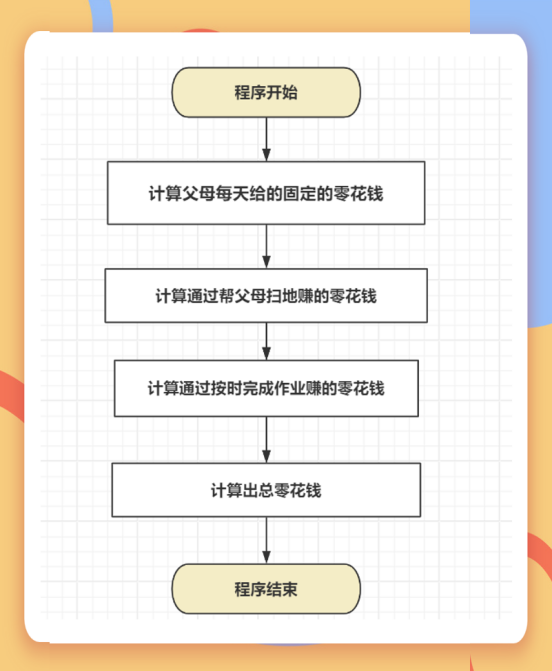
#### 【★】流程图

**在学习C++语言之前，让我们通过一些简单的例题， 来熟悉程序的基本组成和基本语句的用法**

**题目：计算每天的零花钱**

**你每天都会从家里领取固定的零花钱，同时还能通过完成不同的任务，如帮父母扫地或按时完成作业等，来赚取额外的零花钱。假设让你编写一个C++程序，用于计算你每天的零花钱总额，那么该如何实现呢？**

**不妨，我们先用流程图来理一下计算机处理流程，模拟一下计算机该怎么计算每天的零花钱总额：**

****

**在编写代码之前，我们需要确保任务的步骤清晰明了。流程图就像是一张地图，它帮助我们规划程序的执行顺序。我们先画出一个简单的流程图，看看如何用它来指导程序的执行。**

#### 【★】主函数

**程序的起点就是‘主函数’。就像一本书的封面，它定义了程序的开始和结束。接下来，我们将学习如何写这个‘主函数’，并在其中编写我们的代码。**

**流程图设计完了，大家是不是有了思路？在编程之前，我们还可以进一步用伪代码的形式来描述C++程序，该如何计算每天的零花钱总额：**

**主函数:**

**程序开始**

**1.计算父母每天给的固定的零花钱**

**2.计算通过帮父母扫地赚的零花钱**

**3.计算通过每天按时完成作业赚的零花钱**

**4.总零花钱 = 固定零花钱 + 扫地赚的零花钱 + 按时完成作业赚的零花钱**

**程序结束**

**流程图帮我简化了程序的处理流程，伪代码帮我们描述了程序的处理步骤，接下去我们开始尝试编写第一个C++程序。**

**int main() //主函数**

**{**

**1.计算父母每天给的固定的零花钱 //程序开始**

**2.计算通过帮父母扫地赚的零花钱**

**3.计算通过每天按时完成作业赚的零花钱**

**4.总零花钱 = 固定零花钱 + 扫地赚的零花钱 + 按时完成作业赚的零花钱**

**return 0; //程序结束**

**}**

### 第2小节-变量和内存

**逻辑处理部分总共是四个步骤，我们继续上一小节的例子进行讲解，分别计算不同方式获得的零花钱以及总零花钱，那么计算出来的结果存放在哪里呢？我们人脑是存在脑细胞里，那计算机存储在哪里呢？**

**计算机的核心组成部分之一是内存，用于存放计算结果。相当于我们家里的储物柜，可以储存各种物品。而我们逻辑处理部分四个步骤会产生四个计算结果，需要在内存中划出四个空间来存放，为了便于查找，对四个空间分别用不同名字来命名。**

**根据这个思路，假设每天固定零花钱计算得到10元，扫地赚的零花钱得到5元，按时完成作业赚的零花钱得到8元，我们可以这样编程实现：**

**int num1 = 10; // 固定的零花钱**

**int num2 = 5; // 扫地赚的零花钱**

**int num3 = 8; // 按时完成作业赚的零花钱**

**int sum = num1 + num2 + num3; // 总零花钱**

**其中**

**num1表示内存中的一个空间，用于存储每天固定的零花钱。**

**num2表示内存中的一个空间，用于存储每天扫地赚的零花钱。**

**num3表示内存中的一个空间， 用于存储每天按时完成作业赚的零花钱。**

**sum表示内存中的一个空间，用于存储每天总零花钱。**

**内存存储如下图所示**

****

**【上机练习】**

**int main()**

**{**

**int num1 = 10; // 固定的零花钱，也是程序开始**

**int num2 = 5; // 扫地赚的零花钱**

**int num3 = 8; // 按时完成作业赚的零花钱**

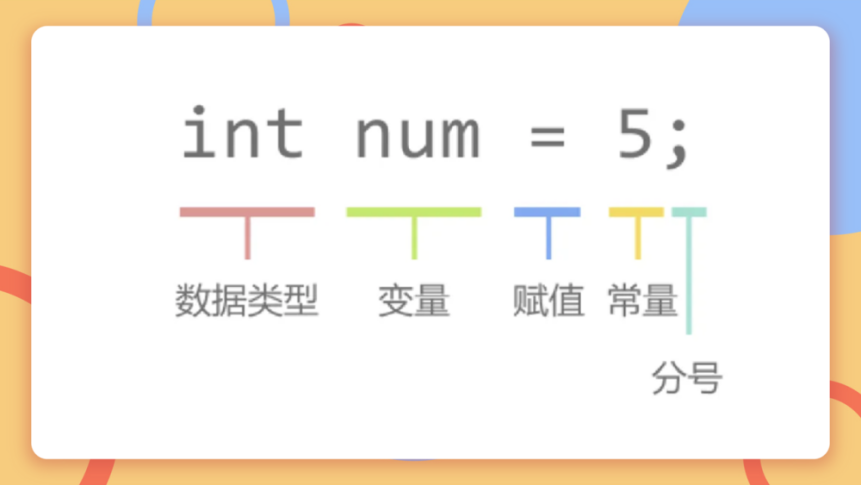
**int sum = num1 + num2 + num3; // 总零花钱**

**return 0; //程序结束**

**}**

**现在，大家动手试试编写这个计算零花钱总额的小程序。记得从定义变量开始，逐步进行计算。编程就是通过这样的步骤，一步步把想法变成实际的程序 。在编程的过程中，错误是很常见的。当你运行代码时，可能会看到报错信息。不要担心，报错是学习编程的一部分。我们需要仔细检查代码，尤其是符号、括号和分号的位置。**

**【章节总结】**

****

**今天的课程我们学习了编程的基础知识，如何通过编写代码让计算机自动完成任务。大家通过写代码计算零花钱总额的例子，掌握了变量、内存、流程图和主函数。希望大家通过今天的学习，对编程有了一个初步的认识。记住，编程的关键在于不断的练习和思考，遇到问题时，要学会调试和解决错误。下次我们会继续深入学习，探索更多的编程技巧。**

#### **【★】变量名**

**在编程中，变量可以被看作是一个存储单元，用来保存数据。在计算机的内存中，变量占据一定的空间，存放具体的值。**

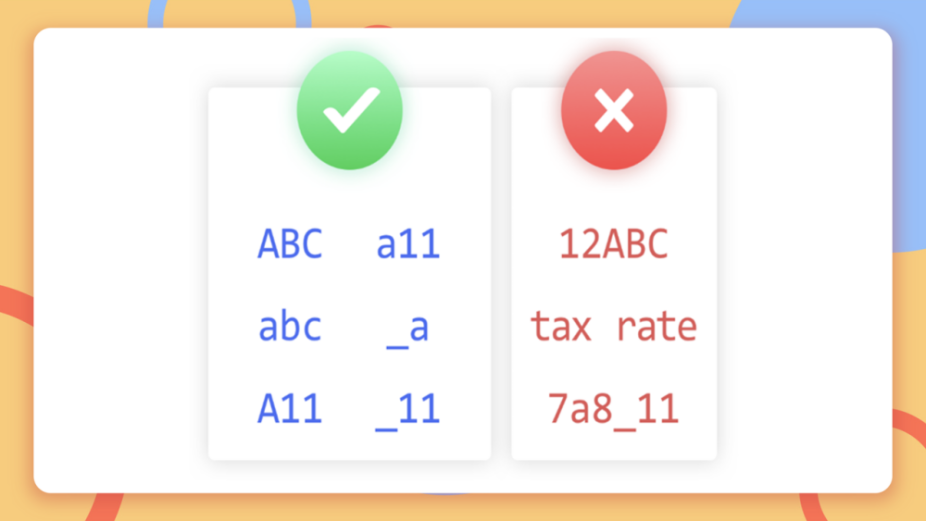
**注意事项**

**变量名也叫用户标识符，只能由字母、数字、下划线组成。**

**1.不能以数字开头，例，12ABC;**

**2.不能包含空格，例，tax rate**

**3.大小写敏感，变量A和变量a是不同的，例， A 和 a**

****

**以下哪一个是错误的C++变量名?  
 A. taxRate  
 B. tax Rate  
 C. tax\_Rate  
  
以下哪个变量名在C++中是允许的？  
 A. %variable  
 B. variable%  
 C. variableName  
  
以下哪一个变量名在C++中是合法的?  
 A. int main  
 B. \_privateVar  
 C. variable name**

### 第3小节-算术运算

**在超市购物后，我们经常会遇到打折促销活动。本小节，我们将学习如何使用算术运算符来进行加赋值操作，并通过编写程序来计算打折后的小票总价后打印，从而更好地理解和掌握算术运算在实际生活中的应用。**

**#include <iostream>  
using namespace std;  
  
int main() {  
 // 定义物品的数量和单价  
 int num1 = 5; // 苹果的数量  
 int price1 = 3; // 苹果的单价  
 int num2 = 3; // 香蕉的数量  
 int price2 = 2; // 香蕉的单价  
 int num3 = 10; // 胡萝卜的数量  
 int price3 = 1; // 胡萝卜的单价  
  
 // 计算总价  
 int sum = (num1 \* price1) + (num2 \* price2) + (num3 \* price3);  
  
 // 输出打折后的小票总价  
 cout << "小票总价: " << sum << "元" << endl;  
 return 0;  
}**

**程序将计算三种物品（苹果、香蕉和胡萝卜）的总价，并输出这个总价。这里是具体的输出内容：**

**小票总价: 23元**

**那么在原来计算并打印的基础上加上打折该如何实现，我们将代码继续用C++语言实现**

**#include <iostream>  
using namespace std;  
  
int main() {  
 int num1 = 5; // 苹果的数量  
 int price1 = 3; // 苹果的单价  
  
 int num2 = 3; // 香蕉的数量  
 int price2 = 2; // 香蕉的单价  
  
 int num3 = 10; // 胡萝卜的数量  
 int price3 = 1; // 胡萝卜的单价  
  
 // 计算未打折的总价  
 int sum1 = (num1 \* price1) + (num2 \* price2) + (num3 \* price3);  
   
 // 计算打折后的总价  
 double sum2 = sum1 \* 0.9;// 90% 的折扣  
  
 // 输出打折后的小票总价  
 cout << "原始小票总价: " << sum1 << "元" << endl;  
 cout << "打折后的小票总价: " << sum2 << "元" << endl;  
  
 return 0;  
}**

**现在，这段代码将正确地计算并输出原始小票总价和打折后的小票总价。输出内容如下：**

**原始小票总价: 31元  
打折后的小票总价: 27.9元**

**前面讲解了C++，现在我们介绍一下C语言的实现。**

**在C语言中，我们同样可以计算小票的总价并应用折扣。与C++不同，C语言使用printf函数来输出信息，并且需要使用格式化占位符来指定输出的格式。以下是C语言的实现代码：**

**#include <stdio.h>  
int main() {  
 int num1 = 5; // 苹果的数量  
 int price1 = 3; // 苹果的单价  
 int num2 = 3; // 香蕉的数量  
 int price2 = 2; // 香蕉的单价  
 int num3 = 10; // 胡萝卜的数量  
 int price3 = 1; // 胡萝卜的单价**

**// 计算未打折的总价  
 int sum1 = (num1 \* price1) + (num2 \* price2) + (num3 \* price3);  
  
 // 计算打折后的总价  
 double sum2 = sum1 \* 0.9; // 90% 的折扣  
  
 // 输出打折后的小票总价  
 // 使用%d作为整数的占位符，%f作为浮点数的占位符  
 printf("原始小票总价: %d元\n", sum1);  
 printf("打折后的小票总价: %.1f元\n", sum2); // %.1f表示保留一位小数  
 return 0;  
}**

**在这段C语言程序中，我们使用了%d作为整数的占位符，用于输出sum1的值。对于浮点数sum2，我们使用了%.1f作为占位符，这表示输出的浮点数将保留一位小数。这是C语言中处理变量输出的一种方式，它允许我们控制输出的格式和精度。**

**在下一小节中，我们将进一步探讨printf函数中变量输出的具体细节，包括各种占位符的使用和它们如何帮助我们定制输出格式。**

### 第4小节-输入和输出

#### 【★】cout 函数

**cout 就是帮我们输出信息到屏幕的‘工具’，使用它可以将信息或数据结果显示给用户，无需自己编写复杂的代码。**

**功能概述： cout 是由技术专家预先编写的功能，专用于将信息输出到屏幕。**

**【头文件：#include <iostream>】**

**在使用cout之前，你需要包含iostream头文件。这个头文件是C++标准库的一部分，它定义了输入输出流，包括cin（用于输入）和cout（用于输出）。你可以在程序的顶部添加以下代码来包含这个头文件：**

**#include <iostream>**

**这行代码告诉编译器在编译你的程序之前，先去包含（插入）iostream这个头文件的内容。**

**一旦你包含了iostream头文件并使用了using namespace std;，你就可以使用cout来向屏幕输出信息了。cout使用插入操作符<<来发送数据到标准输出流。例如：**

**cout << "Hello, World!" << endl;**

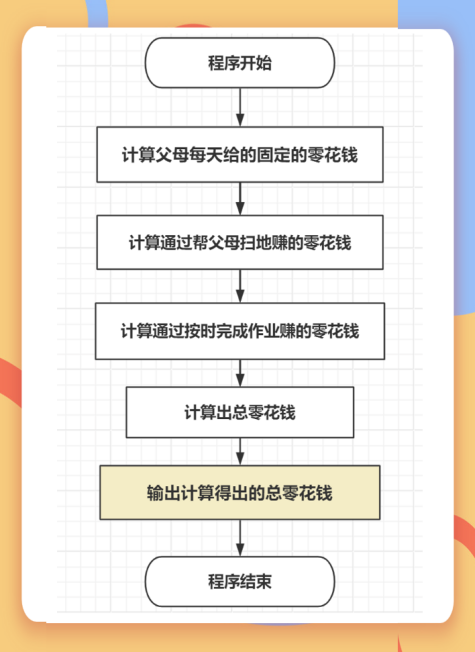
**这行代码会将字符串"Hello, World!"输出到屏幕上，并且endl是一个操纵符，用于在输出后添加一个换行符。**

**继上一章题目：计算每天的零花钱**

**你每天都会从家里领取固定的零花钱，同时还能通过完成不同的任务，如帮父母扫地或按时完成作业等，来赚取额外的零花钱。假设让你编写一个C++程序，用于计算你每天的零花钱总额，那么该如何实现呢？**

**在编写完现有程序后，为了清楚了解每天可以使用的总零花钱，我们还需要进一步让计算机输出总零花钱。**

**不妨，我们先用流程图来理一下计算机处理流程，大家想一想，输出总零花钱放在哪一步最合适？**

****

**根据流程图的思路，我们将其转化伪代码的形式来描述C语言程序：**

**引入头文件  
主函数:  
程序开始  
 1.计算父母每天给的固定的零花钱  
 2.计算通过帮父母扫地赚的零花钱  
 3.计算通过每天按时完成作业赚的零花钱  
 4.总零花钱 = 固定零花钱 + 扫地赚的零花钱 + 按时完成作业赚的零花钱  
 输出总零花钱  
 程序结束**

**【上机练习】**

**紧接着将伪代码转化为C++：**

**#include <iostream> //引入标准输入输出头文件  
using namespace std;  
  
int main()   
{   
 int num1 = 10; // 固定的零花钱，也是程序开始  
 int num2 = 5; // 扫地赚的零花钱  
 int num3 = 8; // 按时完成作业赚的零花钱  
 int sum = num1 + num2 + num3; // 总零花钱  
 cout << sum << endl; // 输出总零花钱  
 return 0; //程序结束   
}**

#### **【★】cin 函数**

**与cout相对应，cin是另一个C++标准库中的工具，它专门用于从标准输入设备（通常是键盘）读取用户输入的数据。使用cin，我们可以轻松地收集用户的信息，而无需编写复杂的代码来处理输入。正如cout一样，cin也定义在iostream头文件中。这意味着，在使用cin之前，我们同样需要在程序的顶部包含这个头文件：**

**#include <iostream>**

**一旦包含了iostream头文件并使用了using namespace std;，我们就可以利用cin来从用户那里获取输入了。cin使用插入操作符>>来从标准输入流接收数据。例如：**

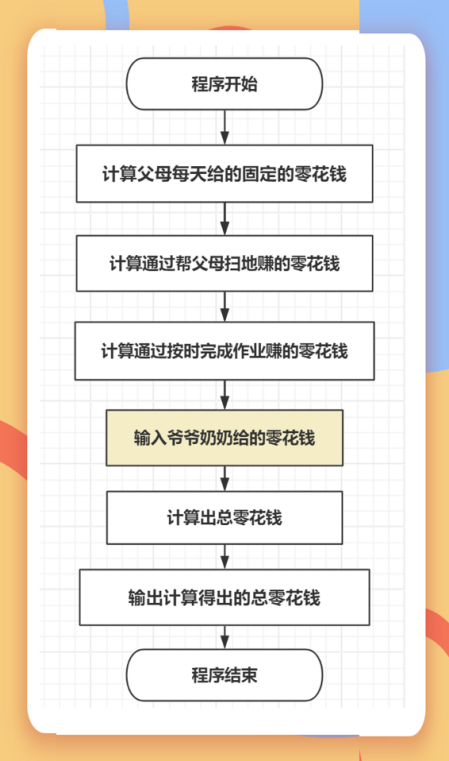
**int number;  
cout << "请输入一个整数：";  
cin >> number;  
cout << "你输入的整数是：" << number << endl;**

**在这个例子中，程序首先提示用户输入一个整数，然后使用cin读取用户输入的整数并存储在变量number中。之后，程序使用cout输出用户输入的整数。**

**cin：这是一个预定义的输入流对象，用于从标准输入设备（通常是键盘）读取用户输入的数据。它是由技术专家预先编写的功能，专用于将用户输入的数据读取到程序中。**

**爷爷奶奶可能还会给你一些随机数额的零花钱，计算机无法直接得出这些数额，所以我们需要手动输入。**

**不妨，我们先用流程图来理一下计算机处理流程，大家想一想，输入这些随机数额放在哪一步最合适？**

****

**根据流程图的思路，我们将其转化伪代码的形式来描述C语言程序：**

**引入头文件  
主函数:  
程序开始  
 1.计算父母每天给的固定的零花钱  
 2.计算通过帮父母扫地赚的零花钱  
 3.计算通过每天按时完成作业赚的零花钱  
 4.输入爷爷奶奶给的零花钱   
 5.总零花钱 = 固定零花钱 + 扫地赚的零花钱 + 按时完成作业赚的零花钱 + 爷爷奶奶给的零花钱  
 输出总零花钱  
 程序结束**

**【上机练习】**

**紧接着将伪代码转化为C++：**

**#include <iostream> //引入标准输入输出头文件  
using namespace std;  
  
int main()   
{   
 int num1 = 10; // 固定的零花钱，也是程序开始  
 int num2 = 5; // 扫地赚的零花钱  
 int num3 = 8; // 按时完成作业赚的零花钱  
 int num4; // 声明变量，用于存储用户输入的零花钱   
 cin >> num4; //从用户输入中读取整数并存储在num4变量中  
 int sum = num1 + num2 + num3 + num4; // 总零花钱  
 cout << sum << endl; // 输出总零花钱  
 return 0; //程序结束   
}**

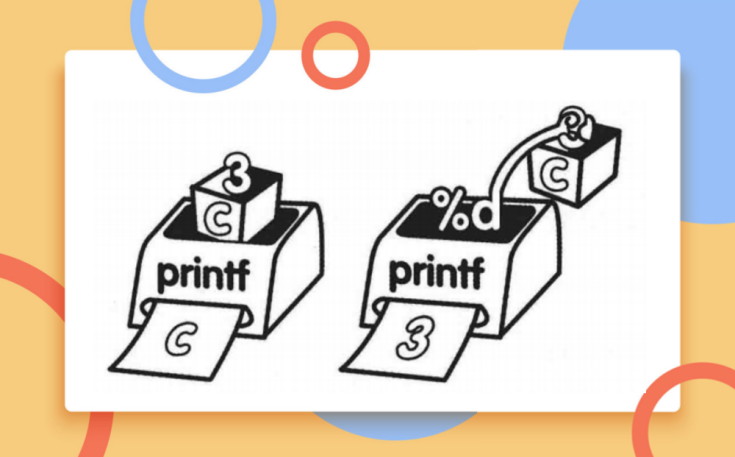
#### **【★】printf函数**

**根据上文学到的C++输入输出知识，现在我们来学习C语言的输入和输出。同样，只要把要输出的内容放在双引号里面就可以了，代码如下:**

**printf(“Hello, world!\n”)**

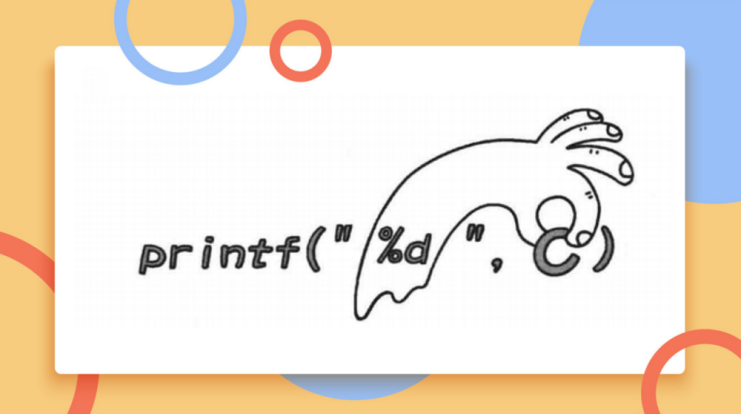
**这行代码会将字符串"Hello, World!"输出到屏幕上，并且\n是一个操纵符，用于在输出后添加一个换行符。**

**当我们想输出变量的值，该如何做呢？**

****

**“%d”其实是一个“讨债的”，或者也可以说是“要饭的”。它的专职工作就是向别人“要钱”!那我们应该怎么使用它呢?**

**printf(“%d”, c);**

****

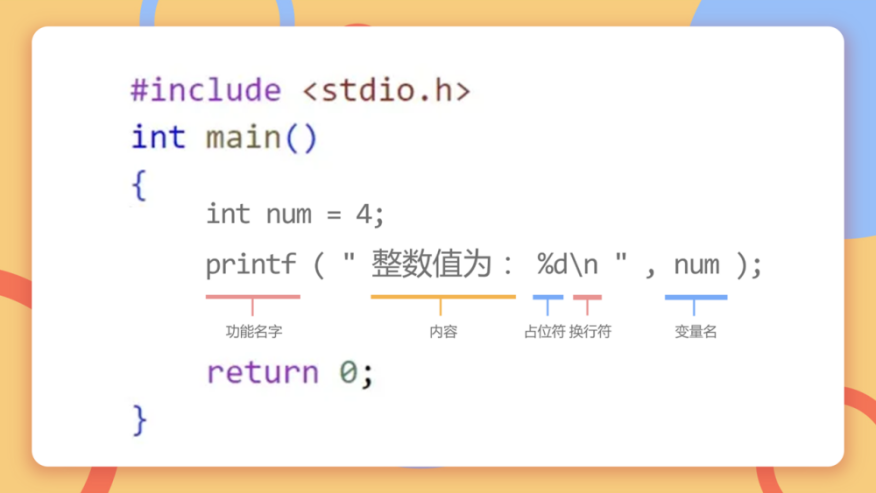
**将“%d”放在双引号之间，把小房子c放在双引号后面，并且用逗号隔开。这时 printf发现双引号里面是个“讨债的”，就知道此时需要输出一个具体的数值，而不是符号，就会向双引号后面的小房子c索取具体的数值了。**

**【真题练习】**

**C++语句 printf("6%2={%d}", 6%2) 执⾏后的输出是 \_\_\_\_ 。  
A.  
 "6%2={6%2}"  
B.  
 6%2={6%2}  
C.  
 0=0  
D.  
 6%2={0}  
解析：  
本题考察printf格式化输出语句；printf语句中将会输出双引号中的格式化字符串，  
其中（%）为占位符，如本题的%d表示使用整数格式输出参量表中对应参量的值。  
参量表为格式化字符串右侧内容，与格式化字符串中的占位符一一对应。  
本题执行语句后会输出双引号中的内容，其中%d对应参量表中的6 % 2，  
6%2是取模运算，表示6除以2的余数，结果是0。其值为0。  
最终输出为6%2={0}。所以本题正确答案为D。**

**输出变量**

**变量的值为整数**

****

**1）输出变量的格式是（"%类型\n"，变量），需要填写两个部分，(数据的类型，变量名称)，中间用逗号隔开。**

**2）%d 是一个占位符，作用是指输出 num 值的位置。%告诉程序有一个变量在这个位置输出。d告诉程序将输出整数变量，输出浮点数使用 f，输出字符使用c。**

**如下图所示。**

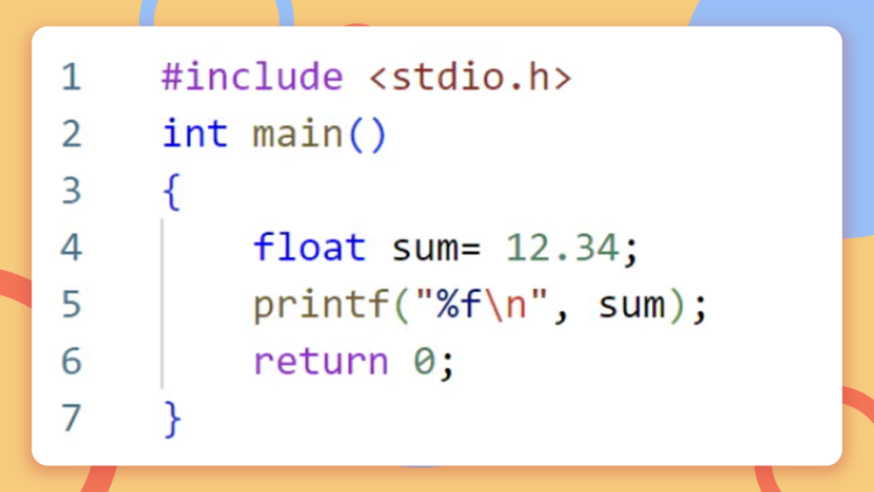
****

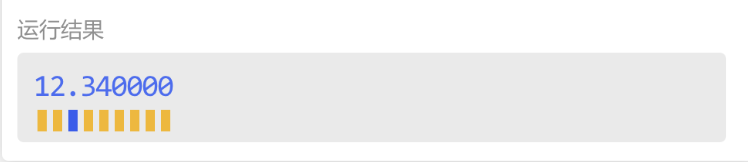
**3）换行符，与前面一致，输出的内容后面一定要加\n ，表示换行。**

**4）变量，num是指要输出变量名。**

**变量的值为浮点数**

**假设我们定义一个浮点数 sum，赋值为 12.34，则输出如下如所示**

****

****

**运行结果解释：浮点数默认小数点后6位数字，没有6位会自动补0，所以输出结果小数点后有6位。由于小数点也占1个字段，所以输出的内容共占9个字段。**

#### **【★】scanf函数**

**scanf(”%d”, &a);**

**你瞧，与输出小房子a的语句 printf("%d",a);是差不多的，只有以下两处不同。**

**第一处是:读入是使用 scanf这个词，而输出是使用printf这个词。**

**第二处是:读入比输出在a前面多了一个“&”符号。**

**“&”符号我们称为“取地址符”，简称“取址符”。它的作用是得到小房子a的地址。**

**scanf("%d",&a);这句话可以理解为:我们要从外界向计算机的内部传送一个数值，并需要将这个值存放到指定的“编号为a”的盒子中。这样我们就需要知道小盒子a在计算机内部的地址。就像邮递员送信一样，要传送的数据就是信件，小盒子a就是信箱编号。**

****

**那你可能要问，为什么在读入的时候要得到小房子a的地址，而输出的时候却不需要呢?在读入数据的时候，计算机需要把读入的值存放在小房子a(也就是变量a)中，此时需要知道你指定的这个小房子a的地址，才能把值准确地放进其中;但是在输出的时候，值已经在小房子a中了，因此可以直接输出到屏幕上。**

**打一个比方:假如你要去一个教室上课，那么在上课之前你需要知道这个教室的地址，这样你才能去;但是如果下课了，你需要走出这个教室，因为此时你已经在教室中，所以就不再需要这个教室的地址。**

**如果要从键盘读入两个数,分别给小房子a和小房子b呢?这里有以下两种写法。**

**第一种:**

**scanf(”%d”, &a);**

**scanf(”%d”, &b);**

**第二种:**

**scanf("%d%d",&a,&b);**

**第二种的写法较为简便，两个“%d”之间不需要空格,“&a”和“&b”之间用逗号隔开。**

**【章节总结】**

**在本章节中，我们探讨了C++中两个核心的输入输出流对象：**

**cout和cin，它们是实现程序与用户之间交互的关键工具.**