**与字符串相关的三个常用库和相应函数的介绍**

**前言：**

把 C 语言想成一个工厂

stdio.h、string.h、ctype.h 等等就是工厂里的不同功能车间

它们都是“库（Library）”，并列存在，互不从属。

**标准输入输出函数<stdio.h>**

老朋友了，从我们一接触 C 语言开始就离不开它。  
学习编程时，我们需要让程序把内容输出到终端（屏幕）上供我们检查，也需要从终端输入信息与程序交互。  
所以对于初学者，甚至所有面向用户交互的程序来说，stdio.h 都是不可或缺的标准库。  
就像程序的“眼睛”和“嘴巴”一样，它负责读入信息、表达结果。

**成员介绍：**

**Getchar() 和Putchar()**—— **单字符输入输出的一对轻量函数**



需要注意的是，`getchar()` 会原样读取包括换行符（`\n`）在内的所有字符，且不像 `scanf("%c")` 那样可以在前面加空格跳过空白字符。也因此，它常被用于清理输入缓冲区中的残留内容（例如 `while(getchar() != '\n');`），确保后续输入不会出错。

而 `putchar()` 虽然功能简单，仅用于输出单个字符，但在循环中常被用于逐字符输出，甚至可以用来\*\*倒序输出字符数组\*\*等场景，是构建字符处理逻辑的一种高效工具。

**Null** 是“空指针”标志，用来判断指针是否有效，是安全编程中不可或缺的常量。

**Eof** 是输入读取结束或出错的“信号值”，多用于循环中断条件；

**printf() 和 scanf() —— 格式化输入输出函数的“双子星”**

很多格式指定子在 printf() 和 scanf() 中都可以使用，但有些行为上**略有不同**

|  |  |  |  |
| --- | --- | --- | --- |
| **格式** | **printf 用法示例** | **scanf 用法示例** | **说明** |
| %d | printf("%d", i); | scanf("%d", &i); | 有符号十进制整数（int） |
| %x | printf("%x", i); | scanf("%x", &i); | 十六进制整数（输入不带0x） |
| %o | printf("%o", i); | scanf("%o", &i); | 八进制整数 |
| %u | printf("%u", i); | scanf("%u", &i); | 无符号整数 |
| %f | printf("%f", f); | scanf("%f", &f); | 浮点数（float） |
| %lf | printf("%lf", d); | scanf("%lf", &d); | 双精度浮点（double） |
| %c | printf("%c", ch); | scanf(" %c", &ch); | 单个字符（注意 scanf 要加空格跳过换行） |
| %s | printf("%s", str); | scanf("%s", str); | 字符串，scanf 遇空白符结束 |

注意点①：

单精度小数%f 和 双精度小数%lf 在 printf 中表现相同、但在 scanf 中严格区分

float num = 3.14f;

printf("%f", num); //会将 float 提升为 double 后输出

float num;

scanf("%lf", &num); //错误：%lf 期望的是 double

注意点②：

scanf 使用 %c 读取字符时，可能受到“缓冲区残留”的影响。

Char moji;

scanf("%d", &moji); // 之前因为种种操作，比如输入5回车写进，5被读取回车还残留在缓冲区

scanf("%c", &moji); // 第二次假设你用的是%c,他就会把缓冲区残留的回车读进去

解决办法1: scanf(" 空格 %c", &moji); // 这个空格的作用是让 scanf() 跳过输入缓冲区中的空白字符（包括换行）

解决办法2：while (getchar() != '\n'); // 通过getchar一个个拿出回车直到有效字符

注意点**③**

使用%s时，scanf 无法读入带空格的字符串，遇空格或回车就结束，而 printf 可以完整输出。

**字符处理函数<ctype.h>**

字符分类与转换库 <ctype.h>   
这个库负责“识字”，它能判断一个字符是不是字母、数字、空格等，还能做大小写转换。  
你如果想写一个判断用户输入是否合法的功能，比如“必须全是字母”“不能有空格”，那 ctype.h 的各种 isxxx() 和 toxxx() 函数就特别有用了。  
简单来说，它是 C 语言中处理“单个字符”性质判断和转换的专家。

文本

AI 生成的内容可能不正确。

**成员介绍**

**Isalnum()** 是英数字（A-Z；a-z；0-9）吗？

**Isalpha()** 是字母（A-Z；a-z）字符吗？

**Isdigit()** 是数字（'0'~'9'）字符吗？

**Isgraph()** 是图形字符吗？（可见非空格）

**Iscntrl()** 是控制字符吗？（如\n,\t）

**Islower()** 是小写字母吗？

**Isupper()** 是大写字母吗？

**Ispunct()** 是标点符号吗？

**Isspace()** 是空白字符吗？（空格、换行等）

**Isprint()** 是可打印字符吗？（含空格）

**Isxdigit()** 是十六进制字符吗？（0f）

（转换）**Tolower()** 将字母转为小写（若非字母原样返回）

（转换）**Toupper()** 将字母转为大写（若非字母原样返回）

**字符串处理函数<string.h>**

一提到字符串操作，比如拷贝、拼接、比较、计算长度，就一定绕不开这个库。  
string.h 就像一个专门处理文本的工具箱，它负责帮你管理字符数组，让你不用手动一个字符一个字符地搬运。  
当我们在处理名字、密码、输入文本等“文字类数据”时，string.h 就是你的好帮手。

**Strcpy(1，2)** 拷贝字符串：2复制到1

Strcpy\_s() 安全版本

**Strcat(1，2)** 拼接字符串：：2拼接到1

Stract\_s() 安全版本

**Strlen()** 返回字符串长度（不含\0）

**Strcmp(str1 , str2 )** 比较两个字符串是否完全相等 ==0为相等， str1>2为正，反之为负