深入Linux C/C++

目录

[环境篇 3](#_Toc3058)

[1. Linux C/C++开发环境 3](#_Toc204)

[C语言篇 3](#_Toc26595)

[2. 基本数据类型 3](#_Toc26459)

[2.1. sizeof 3](#_Toc31143)

[2.2. 基本数据类型 3](#_Toc17369)

[2.3. 符号位于补码 4](#_Toc2158)

[2.4. 浮点数 4](#_Toc7645)

[3. 复合数据类型 5](#_Toc3576)

[3.1. 结构体 5](#_Toc24226)

[3.2. 共用体 5](#_Toc13091)

[3.3. 字节对齐 5](#_Toc9687)

[4. 数组 5](#_Toc29185)

[5. 指针 5](#_Toc481)

[6. 函数 5](#_Toc4507)

[6.1. 函数传参 5](#_Toc3853)

[6.2. 函数指针 5](#_Toc4297)

[6.3. 调用栈 5](#_Toc23949)

[7. const与static 5](#_Toc199)

[8. 类型识读 5](#_Toc15418)

[9. 宏与预编译 5](#_Toc5485)

[C++语言篇 6](#_Toc13926)

[10. 面向对象回顾 6](#_Toc8245)

[11. 继承 6](#_Toc17737)

[12. 内存管理 6](#_Toc5242)

[12.1. 堆内存 6](#_Toc31182)

[12.2. 内存布局 6](#_Toc26254)

[13. 异常和系统调用 6](#_Toc18887)

[13.1. 异常 6](#_Toc25754)

[13.2. 系统调用 6](#_Toc17370)

[应用篇 7](#_Toc24420)

[14. 文件操作 7](#_Toc18329)

[15. 网络编程 7](#_Toc27251)

[附加篇 7](#_Toc2985)

[16. 编程规范 7](#_Toc30250)

# 环境篇

# Linux C/C++开发环境

# C语言篇

# 基本数据类型

## sizeof

C语言中提供了sizeof运算符，用于计算数据类型或者变量的数据长度，sizeof为单目运算符，写作：sizeof(*数据类型*) 或者 sizeof(*变量名*)，比如以下代码。

|  |
| --- |
| printf(“%d\n”, sizeof(char));  printf(“%d\n”, sizeof(long));  char var1 = 5;  long var2 = 6;  printf(“%d\n”, sizeof(var1));  printf(“%d\n”, sizeof(var2)); |

表格 1 sizeof 使用举例

虽然采用sizeof()的形式，但sizeof不是一个被调用的函数，在编译器编译期间可以获知数据类型或者变量的长度，并将sizeof()部分替换为数据类型的长度。sizeof()运算符不仅能获取基本数据类型的长度，也能获取结构体、共用体数据的长度。

## 基本数据类型

C语言支持多种不同的基本数据类型，比如 char、int等，而这些基本的数据类型在不同的编译环境下的长度并不相同(主要受目标计算机字长的影响，通常为16位，32位和64位)。具体的数据类型占用字节数如下：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 类型 | DOS  Turbo C  16bit | Linux  GCC  32bit | Linux  GCC  64bit | Windows  VC  32bit | Windows  VC  64bit |
| char | 1 | 1 | 1 | 1 | 1 |
| unsigned char | 1 | 1 | 1 | 1 | 1 |
| short | 2 | 2 | 2 | 2 | 2 |
| unsigned short | 2 | 2 | 2 | 2 | 2 |
| int | 2 | 4 | 4 | 4 | 4 |
| unsigned int | 2 | 4 | 4 | 4 | 4 |
| long | 4 | 4 | 8 | 4 | 4 |
| unsigned long | 4 | 4 | 8 | 4 | 4 |
| long long |  | 8 | 8 | 8 | 8 |
| unsigned long long |  | 8 | 8 | 8 | 8 |
| float | 4 | 4 | 4 | 4 | 4 |
| double | 8 | 8 | 8 | 8 | 8 |
| void \* | 2 | 4 | 8 | 4 | 8 |

上表中，比如long和unsigned long 数据类型，在32bit和64bit的gcc编译环境下，数据的长度并不相同，在编程的时候需要多加注意。void \*表示C语言中指针数据类型的长度，指针类型的长度全都与计算机的字长相同，这样才能使指针指向任意的地址位置。

值得注意的是，决定的数据类型长度的是编译器而不是运行时的操作系统，比如在32bit gcc编译的程序拷贝到64bit linux下运行，此时long型依然占4个字节。因为64bit linux提供了兼容32bit的运行环境，将64bit的计算机当做32bit计算机来使用。程序的数据长度在编译完成时就已经确定了，并且不会再改变。如果希望long型占用8个字节，则需要使用64bit gcc重新编译。

## 符号位于补码

## 浮点数

# 复合数据类型

## 结构体

## 共用体

## 字节对齐

# 数组

# 指针

# 函数

## 函数传参

## 函数指针

## 调用栈

# const与static

# 类型识读

# 宏与预编译

# C++语言篇

# 面向对象回顾

# 继承

# 内存管理

## 堆内存

## 内存布局

# 异常和系统调用

## 异常

## 系统调用

# 应用篇

# 文件操作

# 网络编程

# 附加篇

# 编程规范