C_C 跨平台I_O操作技巧 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/133/2021_2022_C_C___E8_B7_A8_E5_B9_c97_133847.htm 如果你正在写从文件或网络读写数据的跨平台C/C 代码，那么你必须明白有些问题是因语言，编译器，平台而不同的。 主要的问题是数据对齐，填充，类型大小，字节顺序和默认状态char是否有符号。 对齐 特定机器上，特定的数据被对齐于特定的边界。如果数据没有正确对齐，结果可能是效率降低甚至崩溃。 当你从I/O源读取数据的时候，确保对齐是正确的。 填充 "填充" 是数据集合中不同元素之间的间隔， 一般是为了对齐而存在。不同编译器和平台下，填充的数量可能会不同。?不要假设结构的大小和成员的位置在任何编译器和平台下都是相同的。 不要一次性读取或者写入一整个结构体，因为写入的程序可能会使用和读取的程序不同的填充方式。对于域也同样适用。 类型大小 不同数据类型的大小随编译器和平台而不同。 在C/C 中， 内置类型的大小完全取决于编译器(在特定范围内). 不要读写大小不明确的数据类型。也就是说，不要读写bool, enum, long, int, short, float, 或者double类型.(译者注：事实似乎不是这样，我记得C/C 标准规定了一些数据类型的长度，例如short 2字节，long 4字节等等，在符合标准规定的编译器上，使用这些类型可以保证跨平台的正确性) 用这些 替代这些... int8, uint8 char, signed char, unsigned char, enum, bool int16, uint16 short, signed short, unsigned short, enum int32, uint32 int, signed int, unsigned int, long, signed long, unsigned long, enum int64, uint64 long, signed long, unsigned long int128, uint128 long long, signed

long long, unsigned long long float32 float float64 double 字节顺序 字节顺序，就是字节在内存中存储的顺序。 不同的处理器存储多字节数据的顺序是不同的。 小端处理器由低到高存储(换句话说，和书写的顺序相反).。大端处理器由高到低存储(和书写顺序相同)。如果数值的字节顺序和读写它的处理器不同，它必须被事先转化。同时，为了标准化网络传输的字节顺序，定义了网络字节顺序。 char - 有符号还是无符号? 一个鲜为人知的事实，char默认可以是有符号的也可以是无符号的-完全取决于编译器。结果导致你从char转化为其他类型的时候(比如int)，结果会因编译器而不同。 例如： char x. int y. read(fd, &amp.x, 1). // 读取一个byte值为0xff y = x. // y 是 255 或者 -1, 依赖编译器 不要把数据读入一般的char。明确指定是有符号或者无符号的 100Test 下载频道开通，各类考试题目直接下载。详细请访问 www.100test.com