计算机二级C_C 误区四:charc=getchar().计算机二级考试 PDF 转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/584/2021_2022__E8_AE_A1_E7_AE_97_E6_9C_BA_E4_c97_584446.htm 2009年下半年全国计算机等级考试你准备好了没?考计算机等级考试的朋友,2009年下半年全国计算机等级考试时间是2009年9月19日至23日。更多优质资料尽在百考试题论坛 百考试题在线题库 许多初学者都习惯用 char 型变量接收 getchar、getc，fgetc 等函数的返回值，其实这么做是不对的，并且隐含着足以致命的错误。getchar 等函数的返回值类型都是 int 型，当这些函数读取出错或者读完文件后，会返回 EOF.EOF 是一个宏，标准规定它的值必须是一个 int 型的负数常量。通常编译器都会把 EOF 定义为 -1.问题就出在这里，使用 char 型变量接收 getchar 等函数的返回值会导致对 EOF 的辨认出错，或者错把好的数据误认为是 EOF，或者把 EOF 误认为是好的数据。例如： int c. /* 正确。应该使用 int 型变量接收 fgetc 的返回值 */ while ((c = fgetc(fp)) != EOF) { putchar(c). } 如上例所示，我们很多时候都需要先用一个变量接收 fgetc 等函数的返回值，然后再用这个变量和 EOF 比较，判断是否已经读完文件。上面这个例子是正确的，把 c 定义为 int 型保证了它能正确接收 fgetc 返回的 EOF，从而保证了这个比较的正确性。但是，如果把 c 定义为 char 型，则会导致意想不到的后果。 首先，因为 fgetc 等函数的返回值是 int 型的，当赋值给 char 型变量时，会发生降级，从而导致数据截断。例如： ---------- | 十进制 | int | char | |--------|-------------|-------| | 10 | 00 00 00 0A | 0A | | -1 | FF FF FF FF | FF | | -2 | FF FF FF FE | FE |

---------- 在此，我们假设 int 和 char 分别是 32 位和 8 位的。由上表可得，从 int 型到 char 型，损失了 3 个字节的数据。而当我们要拿 char 型和 int 型比较的时候，char 型会自动升级为 int 型。char 型升级为 int 型后的值会因为它到底是 signed char 还是 unsigned char 而有所不同。不幸的是，如果我们没有使用 signed 或者 unsigned 来修饰 char，那么我们无从知晓 char 到底是指 unsigned char 还是指 signed char，因为这是由编译器决定的。不过，无论 char 是 signed 的也好，unsigned 的也罢，都不能改变使用 char 型变量接收 fgetc 等函数的返回值是错误的这个事实。唯一能改变的是该错误导致的后果。前面我们说了，char 型和 int 型比较时，char 会自动升级为 int，下面我们来看看 signed char 和 unsigned char 在转换成 int 后，它们的值有什么不同：
---------- | char | unsigned | signed | |-------|--------------|------------| | 10 | 00 00 00 0A | 00 00 00 0A | | FF | 00 00 00 FF | FF FF FF FF | | FE | 00 00 00 FE | FF FF FF FE | ---------- 由上表可知，当 char 是 unsigned 的时候，其转换为 int 后的值是正数。也就是说，假如我们把 c 定义为 char 型变量，而编译器默认 char 为 unsigned char，那么以下表达式将永远成立。 （c = fgetc（fp））！= EOF /* c 的值永远为正数，而标准规定 EOF 为负数 */ 也就是说以下循环是一个死循环。 while ((c = fgetc(fp)) != EOF) { putchar(c). } 读到这里，可能有些读者朋友会说：“ 那么我明确把 c 定义为 signed char 型的就没问题了吧！ ”很遗憾，就算把 c 定义为 signed char，仍然是错误的。假设 fgetc 等函数读到一个字节的值为 FF，那么返回值就是 00 00 00 FF.

把这个值赋值给 c 后， c 的值变成 FF.然后 c 的值为了和 EOF 比较，会自动升级为 int 型的值，也就是 FF FF FF FF.从而导致以下表达式不成立。 （c = fgetc（fp）） ！= EOF /* 读到值为 FF 的字符，误认为 EOF */ 也就是说以下循环在没有读完文件的情况下提前退出。 while ((c = fgetc(fp)) != EOF) { putchar(c). } 综上所述，使用 char 型变量接收 fgetc 等函数的返回值是错误的，我们必须使用 int 型变量接收这些函数的返回值，然后判断接收到的值是否 EOF.只有判断发现该返回值并非 EOF，我们才可以把该值赋值给 char 型变量。 同理，C 中，用 char 型变量接收 cin.get（） 的返回值也是错误的。不过，把 char 型变量当作参数传递给 cin.get 则是正确的。例如： char c = cin.get(). // 错误，理由同上 char c. cin.get(c). // 正确 特别推荐：2009年9月全国计算机等级考试时间及科目预告 2009年上半年全国计算机等级考试参考答案请进入计算机考试论坛 2009年全国计算机等级考试报名信息汇总 2009年NCRE考试有新变化 2009年全国计算机等级考试大纲 2009年上半年全国计算机二级考试试题及答案 2009年上半年全国计算机等级考试试题答案汇总 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