计算机二级C 辅导:GUNC正则表达式计算机等级考试 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/645/2021_2022__E8_AE_A1_E7_AE_97_E6_9C_BA_E4_c97_645643.htm 最近项目中要处理文本,因此就用了gun的正则表达式,它是posix风格的..我们一般使用的都是perl风格的,所以一开始使用可能会有一些不习惯.详细的区别可以在wiki上看到: http://en.wikipedia.org/wiki/Regular_expression 头文件是regex.h可以在里面看到他所提供的接口.这里主要是3个函数和一个结构体: 引用 int regcomp (regex_t *compiled, const char *pattern, int cflags) int regexec (regex_t *compiled, char *string, size_t nmatch, regmatch_t matchptr [], int eflags) void regfree (regex_t *compiled) typedef struct { regoff_t rm_so. regoff_t rm_eo. } regmatch_t. regcomp会将参数pattern编译为compiled,也就是讲字符串编译为正则表达式. 而参数cflags可以是下面1种或者几种的组合: REG_EXTENDED 使用 扩展的 posix Regular Expressions. REG_ICASE 忽略大小写 REG_NOSUB 不存储匹配结果,只返回是否匹配成功. REG_NEWLINE 可以匹配换行. regexec执行匹配.compiled为刚才编译好的正则表达式,string为将要匹配的字符串,nmatch为后面的结构体数组的长度(regmatch_t).matchptr为regmatch_t的一个数组(也就是存储着像perl里面的$0,$1这些的位置,也就是).而 eflag参数则可以是下面中的1个或多个. REG_NOTBOL 会讲^作为一个一般字符来匹配,而不是一行的开始 REG_NOTEOL 会讲$作为一个一般字符来匹配,而不是一行的结束 regfree每次用完之后需要释放这个正则表达式.compiled为需要释放的正则表达式. regmatch_t

中的rm_so为匹配字符的开始位置,rm_eo为结束位置. 说了这么多,其实使用很简单的: 引用 POSIX Regexp Compilation: Using regcomp to prepare to match. Flags for POSIX Regexps: Syntax variations for regcomp. Matching POSIX Regexps: Using regexec to match the compiled pattern that you get from regcomp. Regexp Subexpressions: Finding which parts of the string were matched. Subexpression Complications: Find points of which parts were matched. Regexp Cleanup: Freeing storage. reporting errors. 然后看个例子吧: C代码 #include 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