新手看招：调试工具GDB基本知识全接触 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/237/2021_2022__E6_96_B0_E6_89_8B_E7_9C_8B_E6_c103_237547.htm 1、GDB 是什么？ GDB（GNU symbolic debugger）简单地说就是一个调试工具。它是一个受通用公共许可证即GPL保护的自由软件。 2、GDB特性 象所有的调试器一样，GDB可以让你调试一个程序，包括让程序在你希望的地方停下，此时你可以查看变量，寄存器，内存及堆栈。更进一步你可以修改变量及内存值。GDB是一个功能很强大的调试器，它可以调试多种语言。在此我们仅涉及C和C 的调试，而不包括其它语言。还有一点要说明的是，GDB是一个调试器，而不象VC一样是一个集成环境。你可以使用一些前端工具如XXGDB，DDD等。他们都有图形化界面，因此使用更方便，但它们仅是GDB的一层外壳。因此，你仍应熟悉GDB命令。事实上，当你使用这些图形化界面时间较长时，你才会发现熟悉GDB命令的重要性。下面我们将结合简单的例子，来介绍GDB的一些重要的常用命令。在你调试你的程序之前，当你编译你的源程序时，不要忘了-g选项或其它相应的选项，才能将调试信息加到你要调试的程序中。例如：gcc -g -o hello hello.c 。 3、GDB常用命令简介 GDB的命令很多，本文不会全部介绍，仅会介绍一些最常用的。在介绍之前，先介绍GDB中的一个非常有用的功能：补齐功能。它就如同Linux下SHELL中的命令补齐一样。当你输入一个命令的前几个字符，然后输入TAB键，如果没有其它命令的前几个字符与此相同，SHELL将补齐此命令。如果有其它命令的前几个字符与此相同，你会听到一声警告

声，再输入TAB键，SHELL将所有前几个字符与此相同的命令全部列出。而GDB中的补齐功能不仅能补齐GDB命令，而且能补齐参数。 本文将先介绍常用的命令，然后结合一个具体的例子来演示如何实际使用这些命令。下面的所有命令除了第一条启动GDB命令是在SHELL下输入的，其余都是GDB内的命令。大部分GDB内的命令都可以仅输入前几个字符，只要不与其它指令冲突。如quit可以简写为q，因为以q打头的命令只有quit。List可以简写为l，等等 3．1 启动GDB 你可以输入GDB来启动GDB程序。GDB程序有许多参数，在此没有必要详细介绍，但一个最为常用的还是要介绍的：如果你已经编译好一个程序，我们假设文件名为hello，你想用GDB调试它，可以输入gdb hello来启动GDB并载入你的程序。如果你仅仅启动了GDB，你必须在启动后，在GDB中再载入你的程序。 3．2 载入程序 === file 在GDB内，载入程序很简单，使用file命令。如file hello。当然，程序的路径名要正确。 退出GDB === quit 在GDB的命令方式下，输入quit，你就可以退出GDB。你也可以输入C-d来退出GDB。 3．3运行程序 === run 当你在GDB中已将要调试的程序载入后，你可以用run命令来执行。如果你的程序需要参数，你可以在run指令后接着输入参数，就象你在SHELL下执行一个需要参数的命令一样。 3．4查看程序信息 === info info指令用来查看程序的信息，当你用help info查看帮助的话，info指令的参数足足占了两个屏幕，它的参数非常多，但大部分不常用。我用info指令最多的是用它来查看断点信息。 3．4．1 查看断点信息 info br br是断点break的缩写，记得GDB的补齐功能吧。用这条指令，你可以得到你所设置的所有断点的详细信息。包括断点号

，类型，状态，内存地址，断点在源程序中的位置等。 3．4．2 查看当前源程序 info source 3．4．3 查看堆栈信息 info stack 用这条指令你可以看清楚程序的调用层次关系。 3．4．4 查看当前的参数 info args 3．5 列出源一段源程序 === list 3．5．1 列出某个函数 list FUNCTION 3．5．2 以当前源文件的某行为中间显示一段源程序 list LINENUM 3．5．3 接着前一次继续显示 list 3．5．4 显示前一次之前的源程序 list - 3．5．5 显示另一个文件的一段程序 list FILENAME:FUNCTION 或 list FILENAME:LINENUM 3．6 设置断点 === break 现在我们将要介绍的也许是最常用和最重要的命令：设置断点。无论何时，只要你的程序已被载入，并且当前没有正在运行，你就能设置，修改，删除断点。设置断点的命令是break。有许多种设置断点的方法。如下： 3．6．1 在函数入口设置断点 break FUNCTION 3．6．2 在当前源文件的某一行上设置断点 break LINENUM 3．6．3 在另一个源文件的某一行上设置断点 break FILENAME:LINENUM 3．6．4 在某个地址上设置断点，当你调试的程序没有源程序是，这很有用 break *ADDRESS 除此之外，设置一个断点，让它只有在某些特定的条件成立时程序才会停下，我们可以称其为条件断点。这个功能很有用，尤其是当你要在一个程序会很多次执行到的地方设置断点时。如果没有这个功能，你必须有极大的耐心，加上大量的时间，一次一次让程序断下，检查一些值，接着再让程序继续执行。事实上，大部分的断下并不是我们所希望的，我们只希望在某些条件下让程序断下。这时，条件断点就可以大大提高你的效率，节省你的时间。条件断点的命令如下，在后面的例子中会有示例。 3．6．5 条件断点 break ...if

COND COND是一个布尔条件表达式，语法与C语言中的一样。条件断点与一般的断点不同之处是每当程序执行到断点处，都要计算条件表达式，如果为真，程序才会断下，否则程序会一直执行下去。 3．7 其它断点操作 GDB给每个断点赋上一个整数数字，这个数字在操作断点时起到重要作用，它实际上就代表相应的断点。GDB中的断点有四种状态： 有效(Enabled) 禁止(Disabled) 一次有效(Enabled once) 有效后删除(Enabled for deletion) 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