0x01 gdb命令参考

这边作为备忘，列出常用的命令，方便查阅

主要参考：https://github.com/hellogcc/100-gdb-tips/blob/master/refcard.pdf

若有不会的，可在gdb命令行交互模式查看帮助：`help command`

重要命令

gdb命令	作用
`gdb program [core]`	调试程序
`b [file:]function`	在函数下断点，此时参数也可以是地址(*0x00124365)、文件行号
run [arglist]	以arglist为参数，运行该程序
bt	backtrace，打印程序堆栈信息
p expression	print，计算表达式的值
c	continue，继续运行程序（当遇到断点时）
n	next，单步跟踪程序，当遇到函数调用时，也不进入此函数。函数当一行命令使用
s	step，单步调试，如果有函数调用，则进入函数内
until	运行程序直至循环结束，后跟参数行号`i`，则运行至第`i`行
finish	运行程序，直到当前函数完成返回，并打印返回时的堆栈地址和返回值及参数值等信息
call function	调用程序内可见的函数，并传递参数，就是把一个函数拿来单独使用
q	quit，退出gdb

第一个命令的`core`是指一般错误是有个核心已转储，此时会产生一个coredump文件，这里可以利用这个coredump文件进行调试

具体参考：https://www.cnblogs.com/luhouxiang/p/6830316.html

查看当前栈前16个元素：`> x/16x $esp`

设置断点

gdb命令	作用
`b [file:]n`	break，在file文件下第n行设置断点
`b [file:]function`	break，在file文件下的function函数下断点，（断点在函数开始的位置）
b +offset	在现在停止的地方偏移offset行设置断点
b *addr	在addr地址设置断点
b ... if expression	在表达式条件下，设置断点
delete n	删除n号断点
clear n	清除第n行的断点
disable n	暂停第n个断点
enable n	开启第n个断点
info b	显示目前从程序断点情况
delete breakpoints	清除所有断点

查看源代码

gdb命令	作用
list	列出程序源代码
list n	显示以当前为行号前后10行代码
list func	显示func所在函数的源代码
list	接着上次list命令输出下边的内容

打印表达式

gdb命令	作用
print expr	expr可以是任何当前正在被测试程序的有效表达式，包括数字、变量、函数调用
print a	打印a的值
print ++a	把a的值+1，并显示出来
print test(2)	把2作为test函数的参数，调用test()函数
display expr	在单步运行时将非常有用，使用display命令设置一个表达式后，它将在每次单步进行指令后，紧接着输出被设置的表达式及值
watch expr	设置一个监视点，一旦被监视的“表达式”的值改变，gdb将强行终止正在被调试的程序。如： watch a
whatis func&var	查询变量或函数
info function	查询函数
info locals	显示当前堆栈页的所有变量

查询运行信息

gdb命令	作用
where/bt	当前运行的堆栈列表
bt	backtrace，显示当前调用堆栈
up/down	改变堆栈显示的深度
set args	参数:指定运行时的参数
show args	查看设置好的参数
info program	来查看程序的是否在运行，进程号，被暂停的原因

分割窗口

- layout：用于分割窗口，可以一边查看代码，一边测试：
- layout src：显示源代码窗口
- layout asm：显示反汇编窗口
- layout regs：显示源代码/反汇编和CPU寄存器窗口
- layout split：显示源代码和反汇编窗口
- Ctrl + L：刷新窗口