1 简介

GDB（GNU Debugger）是GCC的调试工具。其功能强大，现描述如下：
GDB主要帮忙你完成下面四个方面的功能：
1.启动你的程序，可以按照你的自定义的要求随心所欲的运行程序。
2.可让被调试的程序在你所指定的调置的断点处停住。（断点可以是条件表达式）
3.当程序被停住时，可以检查此时你的程序中所发生的事。
4.动态的改变你程序的执行环境。

2 生成调试信息

一般来说GDB主要调试的是C/C++的程序。要调试C/C++的程序，首先在编译时，我们必须要把调试信息加到可执行文件中。使用编译器（cc/gcc/g++）的 -g 参数可以做到这一点。如：

gcc -g hello.c -o hello

g++ -g hello.cpp -o hello

如果没有-g，你将看不见程序的函数名、变量名，所代替的全是运行时的内存地址。当你用-g把调试信息加入之后，并成功编译目标代码以后，让我们来看看如何用gdb来调试他。

3 启动GDB 的方法

gdb program

program 也就是你的执行文件，一般在当前目录下。

4 程序运行上下文

4.1 程序运行参数

set args 可指定运行时参数。（如：set args 10 20 30 40 50 ）

show args 命令可以查看设置好的运行参数。

run 启动程序

4.2 工作目录

cd 相当于shell的cd命令。

pwd 显示当前的所在目录。

5 设置断点

5.1 简单断点

break 设置断点，可以简写为b

b 10 设置断点，在源程序第10行

b func 设置断点，在func函数入口处

5.2 多文件设置断点

在进入指定函数时停住:

C++中可以使用class::function或function(type,type)格式来指定函数名。如果有名称空间，可以使用namespace::class::function或者function(type,type)格式来指定函数名。

break filename:linenum -- 在源文件filename的linenum行处停住
break filename:function -- 在源文件filename的function函数的入口处停住

break class::function或function(type,type) -- 在类class的function函数的入口处停住

break namespace::class::function -- 在名称空间为namespace的类class的function函数的入口处停住

5.3 查询所有断点

info b

6条件断点

一般来说，为断点设置一个条件，我们使用if关键词，后面跟其断点条件。

设置一个条件断点

b test.c:8 if intValue == 5

7 维护停止点

delete [range...] 删除指定的断点，如果不指定断点号，则表示删除所有的断点。range 表示断点号的范围（如：3-7）。其简写命令为d。

比删除更好的一种方法是disable停止点，disable了的停止点，GDB不会删除，当你还需要时，enable即可，就好像回收站一样。

disable [range...]

disable所指定的停止点，如果什么都不指定，表示disable所有的停止点。简写命令是dis.

enable [range...]

enable所指定的停止点，如果什么都不指定，表示enable所有的停止点。简写命令是ena.

8 调试代码

run 运行程序，可简写为r

next 单步跟踪，函数调用当作一条简单语句执行，可简写为n

step 单步跟踪，函数调进入被调用函数体内，可简写为s

finish 退出进入的函数

until 在一个循环体内单步跟踪时，这个命令可以运行程序直到退出循环体,可简写为u。

continue 继续运行程序，可简写为c

9 查看运行时数据

print 打印变量、字符串、表达式等的值，可简写为p
p count 打印count的值

10 自动显示

你可以设置一些自动显示的变量，当程序停住时，或是在你单步跟踪时，这些变量会自动显示。相关的GDB命令是display。

display 变量名

info display -- 查看display设置的自动显示的信息。

undisplay num（info display时显示的编号）

delete display dnums... -- 删除自动显示，dnums意为所设置好了的自动显式的编号。如果要同时删除几个，编号可以用空格分隔，如果要删除一个范围内的编号，可以用减号表示（如：2-5）

disable display dnums...

enable display dnums...

disable和enalbe不删除自动显示的设置，而只是让其失效和恢复。

11. 查看修改变量的值

(gdb) ptype width -- 查看变量width的类型

type = double

(gdb) p width -- 打印变量width 的值

$4 = 13

你可以使用set var命令来告诉GDB，width不是你GDB的参数，而是程序的变量名，如：

(gdb) set var width=47

在你改变程序变量取值时，最好都使用set var格式的GDB命令。

12 显示源代码

GDB 可以打印出所调试程序的源代码，当然，在程序编译时一定要加上 –g 的参数，把源程序信息编译到执行文件中。不然就看不到源程序了。当程序停下来以后，
GDB会报告程序停在了那个文件的第几行上。你可以用list命令来打印程序的源代码。默认打印10行，还是来看一看查看源代码的GDB命令吧。

list linenum

Print lines centered around line number linenum in the current source file.

list function

显示函数名为function的函数的源程序。

list

显示当前行后面的源程序。

list -

显示当前行前面的源程序。

一般是打印当前行的上5行和下5行，如果显示函数是是上2行下8行，默认是10行，当然，你也可以定制显示的范围，使用下面命令可以设置一次显示源程序的行数。

set listsize count

设置一次显示源代码的行数。(unless the list argument explicitly specifies some other number)

show listsize

查看当前listsize的设置。