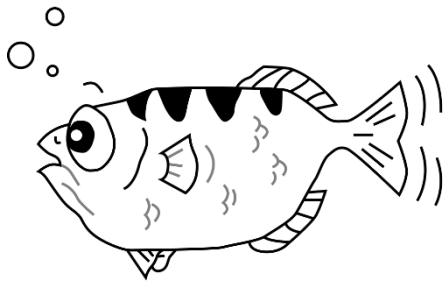
- GDB 是由 GNU 软件系统社区提供的调试工具，同 GCC 配套组成了一套完整的开发环境，GDB 是 Linux 和许多类 Unix 系统中的标准开发环境。
- 一般来说，GDB 主要帮助你完成下面四个方面的功能：
  1. 启动程序，可以按照自定义的要求随心所欲的运行程序
  2. 可让被调试的程序在所指定的调置的断点处停住（断点可以是条件表达式）
  3. 当程序被停住时，可以检查此时程序中所发生的事
  4. 可以改变程序，将一个 BUG 产生的影响修正从而测试其他 BUG



- 通常，在为调试而编译时，我们会（）关掉编译器的优化选项（`-O`），并打开调试选项（`-g`）。另外，`-Wall`在尽量不影响程序行为的情况下选项打开所有warning，也可以发现许多问题，避免一些不必要的 BUG。
- gcc -g -Wall program.c -o program
- `-g` 选项的作用是在可执行文件中加入源代码的信息，比如可执行文件中第几条机器指令对应源代码的第几行，但并不是把整个源文件嵌入到可执行文件中，所以在调试时必须保证 gdb 能找到源文件。

- 启动和退出

  gdb 可执行程序

  quit

- 给程序设置参数/获取设置参数

  set args 10 20

  show args

- GDB 使用帮助

  help

- 查看当前文件代码

  list/l （从默认位置显示）

  list/l 行号 （从指定的行显示）

  list/l 函数名（从指定的函数显示）

- 查看非当前文件代码

  list/l 文件名:行号

  list/l 文件名:函数名

- 设置显示的行数

  show list/listsize

  set list/listsize 行数

- 设置断点

  b/break 行号

  b/break 函数名

  b/break 文件名:行号

  b/break 文件名:函数

- 查看断点

  i/info b/break

- 删除断点

  d/del/delete 断点编号

- 设置断点无效

  dis/disable 断点编号

- 设置断点生效

  ena/enable 断点编号

- 设置条件断点（一般用在循环的位置）

  b/break 10 if i==5

- 运行GDB程序

  start（程序停在第一行）

  run（遇到断点才停）

- 继续运行，到下一个断点停

  c/continue

- 向下执行一行代码（不会进入函数体）

  n/next

- 变量操作

  p/print 变量名（打印变量值）

  ptype 变量名（打印变量类型）

- 向下单步调试（遇到函数进入函数体）

  s/step

  finish（跳出函数体）

- 自动变量操作

  display 变量名（自动打印指定变量的值）

  i/info display

  undisplay 编号

- 其它操作

  set var 变量名=变量值 （循环中用的较多）

  until （跳出循环）