## 进入调试

编译程序时添加-g，使程序文件具有可调试信息。

gdb [程序名] 或 gdb之后file [程序名] 对指定的程序载入调试信息。

run或r开始执行程序，run [启动参数] 可以将参数传递给程序。

## 常用命令简略

|  |  |  |
| --- | --- | --- |
| 命令 | 命令缩写 | 命令说明 |
| list | l | 显示多行代码 |
| break | b | 设置断点，当程序运行到断点处会暂停 |
| info | i | 描述程序的状态 |
| run | r | 开始运行程序 |
| display | disp | 跟踪查看某个变量，每次程序暂停时都显示其值 |
| step | s | 执行下一条语句，会进入函数调用 |
| next | n | 执行下一条语句，不会进入函数调用 |
| print | p | 打印内部变量值 |
| continue | c | 继续程序的运行，直到遇到下一个断点 |
| set vat name=val |  | 设置变量的值 |
| start | st | 开始运行程序，在main函数的第一条语句暂停 |
| file |  | 装入需要调试的程序 |
| kill | k | 终止正在调试的程序 |
| watch |  | 监视变量值的变化 |
| backtrace | bt | 产看函数调用堆栈信息 |
| frame | f | 查看栈帧 |
| quit | q | 退出gdb环境 |

## 具体命令

1. **list**

默认情况下，一次显示10行，从代码起始位置开始显示

显示以第n行为中心的10行代码:list n

显示已函数名为中心的10行代码:list funcName

显示刚才打印过的源代码之前的代码:list -

1. **break**

设置断点，设置对象可以为行数、函数名、结构地址: break location

删除指定编号的断点:delete 断点编号

删除指定行的断点:clear 断点所在的行数

启用断点:enable 断点编号

失效断点:disable 断点编号

查看断点信息:info breakpoints

1. **watch**

设置观察点，当观察对象值发生变化时，程序暂停执行。

使用观察点时，必须保证程序已经运行。

1. info

info args :打印出当前函数的参数及其值

info locals:打印出当前函数中的所有局部变量及其值

info catch:打印出当前函数中的异常处理信息