Investigating the State of Your Program

/run/media/lijunjie/资料/Learning Video/writing-running-fixing-code/03_testing-and-debugging/03_debugging-tools/04_investigating-the-state-of-your-program_instructions.html

One of the most useful feature of a debugger is the ability to investigate the state of your program. The *print* and *display* commands that we have discussed so far provide a start, as they allow you to see what value an expression evaluates to, however, there is much more that you can do.

One useful features is the ability to inspect the current set of stack frames, and move up and down within them. The *backtrace* command lists all of the stack frames (with the current one on top, and *main* on the bottom), so that you can see what function calls got you to the current place. The backtrace also lists the line where each call was made (or where the execution arrow is, for the current frame).

When you print expressions, *gdb* uses the variables in the current scope. However, sometimes, you might want to inspect variables in other frames further up the stack. You can instruct *gdb* to select different frames with *up* and *down* , which move up and down the stack specifically.

One particularly common use of *up* is when your program stops in a failed *assert* . When this happens, *gdb* will stop deep inside the C library, in the code that handles *assert* . However, you will want to get back to your own code, which is a few stack frames up. You can use up a few times until *gdb* returns to a frame corresponding to your code.

You can also get information about various aspects of the program with the *info* command, which has various subcommands. For example, *info frame* will describe the memory layout of the current frame, *info types* will describe the types that are in the current program. There are a variety of *info* commands, but most of them are for more advanced uses—you can use help *info* to see them all.