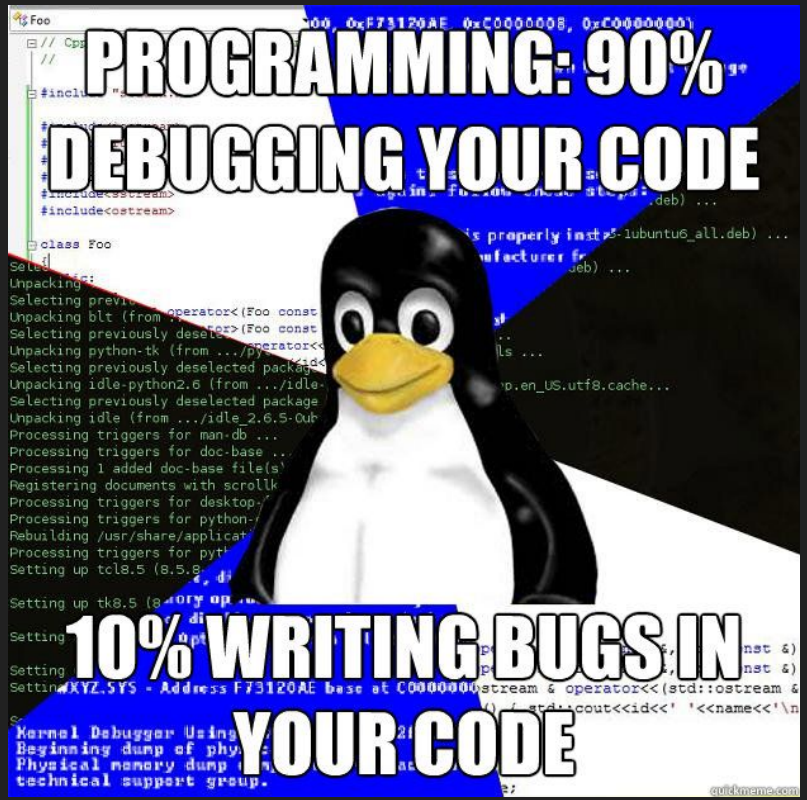
GDB (GNU debugger)

Introduction


PROGRAMMING: 90%
DEBUGGING YOUR CODE
10% WRITING BUGS IN
YOUR CODE

Nobody can write programs without bugs

Except this guy

Testing and debugging are different

Testing

Testing is the process of finding bugs. You should have to have a functioning program to test it

Debugging

Debugging is the process of solving bugs.

What can you do if a particular line is causing you troubles?

Use printf and print variable values

`printf` is a very useful command in debugging. But sometimes it is not enough

GDB
The GNU Project Debugger

GDB allows you to,

see what is going on `inside' a program while it executes or what the program was doing at the moment it crashed.

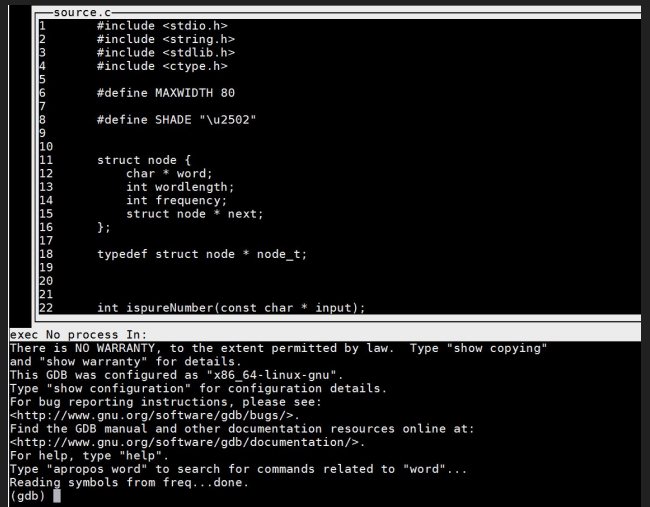
Procedure

1. Compile the program with -g option.
2. gdb <program> or gdbtui <program>
3. run <arg1> <arg2> …

If the program finished running, it would print “exited normally” message.

Otherwise, it will print when it is broken.

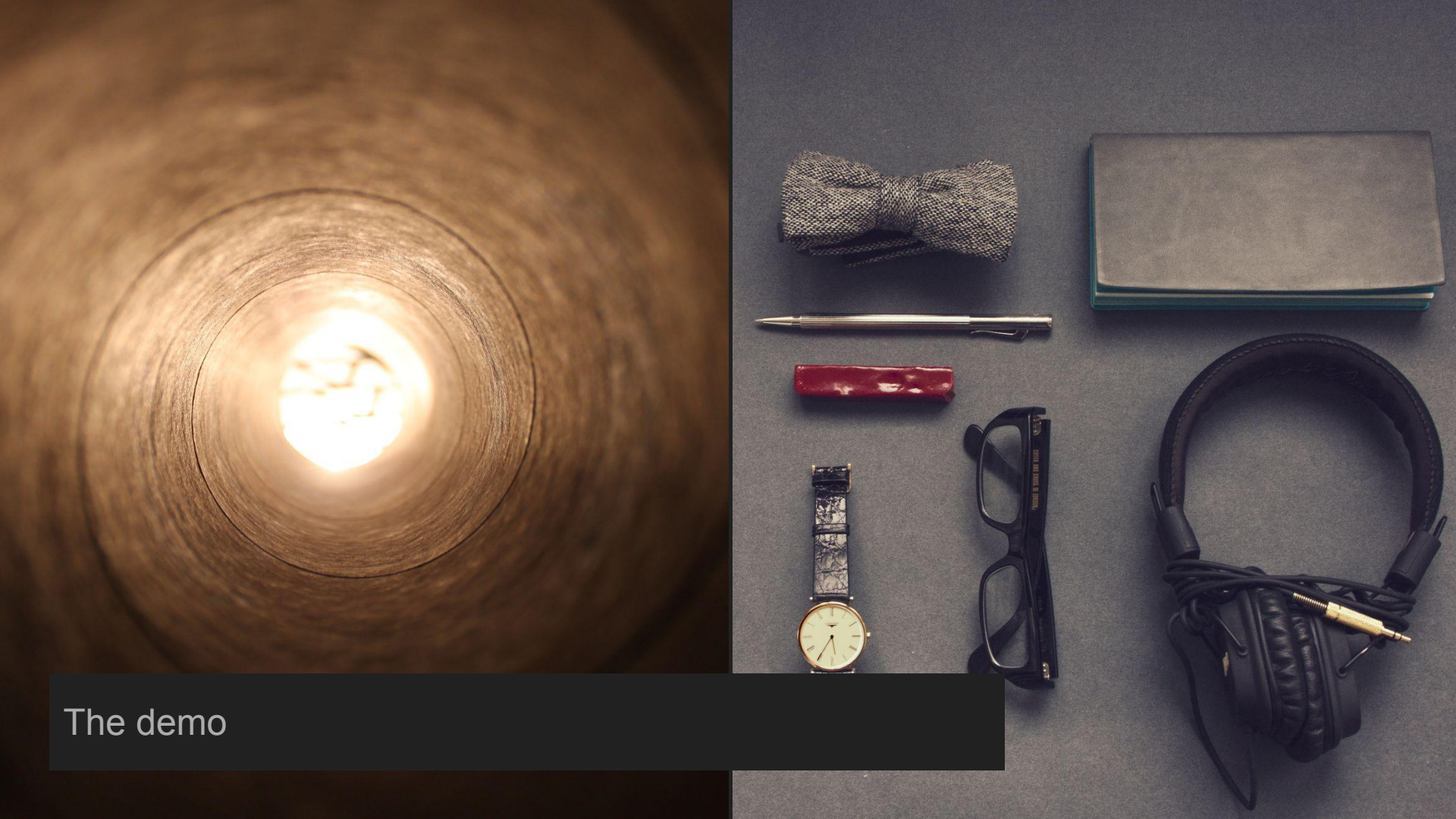
```
GNU gdb (Ubuntu 7.7.1-0ubuntu5~14.04.2) 7.7.1
Copyright (C) 2014 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from freq...done.
(gdb)
```



GDB commands

- **list**
- **run <arguments>**
- **backtrace**
- **break**
- **watch**
- **step/next/finish/continue**
- **print/display**
- **clear/delete**
- **info**

Each command has its own arguments



The demo

list

- You can see the source code of the compiled file!
- I didn’t compile the answer with -g option
- This is one reason you should never use -g option in the production environment. Debugging options should only be enabled in the development environment.
- You can still use gdb to see the assembly code even the program not compiled with -g option

Use: layout asm with gdbtui

run

- Run your program with given arguments
- It will run until it exits or break in the middle

Short version is “r”

backtrace

- Backtrace will show you which function called the current function
- It will also show what are the arguments used

break

- Break the execution of the program at a certain point
  - break <line number>
  - break <function name>
  - break <where> if <condition>

watch

- You can watch a variable when the value changes

step/next/finish/continue

- step
  - Step to the next line. If there is a function call, it will step into the function also
- next
  - step to the next line. This command will not go inside functions
- finish
  - Finish the current function and return
- continue
  - Continue the normal execution

All command use the first letter as the short format

print/display

- print
  - Can be used to print a variable value once
- display
  - Keep printing the variable value at each step
  - To remove, use undisplay command

clear / delete

- Clear a breakpoint
  - clear <breakpoint number>
- Delete
  - Delete all breakpoints

info

- Info b will show all the breakpoints
- And many other informations ..

gdbtui (gdb text user interface)

- **Has four windows**
  - **src**
  - **cmd**
  - **asm**
  - **regs**
- **layout <window>**
- **focus <window>**