Sasta ICP - GDB Handout

Archit Checker

June 29, 2020

Introduction

This document contains several important GDB commands that I covered in the tutorial. GDB is short for the GNU Project Debugger. This is a program that helps us debug several languages such as C, C++, go, and more.

The key idea behind GDB is to provide you access to the program state at any given point in time while the program is running. You can utilize this knowledge to find unexpected behaviour in order to find out what is going wrong, and eventually fix it.

Note that this document is by no means exhaustive. It is only meant to give you an introduction to GDB, but will help you debug a significant chunk of your programs for now.

Compilation and Usage

Remember that you compile C programs using the gcc compiler with the following line -

```
gcc −o <name> <program>.c
```

In order to compile with debugging symbols enabled, you need to use the -g flag in the compilation line. If you wish to debug your program, you should compile with -

```
gcc −o −g <name> <program>.c
```

In order to use gdb with the compiled program, just type -

```
gdb <name>
```

Commands

Some important commands are listed below. Full names of commands are in square brackets, and short forms are outside.

r[un]

The run command is used to run a given program. If the program does not have any errors, this would enable normal execution of the program provided you have not set any breakpoints.

If however, there are any errors, this command will execute until the error and exit unless you set a breakpoint before the error occurs. The run command should tell you the line on which the program crashed. Example usage -

```
r
```

b[reak]

A breakpoint is something that stops execution ideally before or around the area you want to inspect.

The b command sets a breakpoint at the specified point. You can specify line numbers, function names, and even memory addresses as argument to the b command to set a breakpoint. Multiple breakpoints can be set at once. Example usage -

```
b <Line Number/Function Name/etc.>
```

d[elete]

The d command is used to delete a given breakpoint. Example usage -

```
d <Breakpoint number>
```

s[tep]

The s command is used to step through one line of code. It steps into function calls. Example usage -

```
s <Optional: Number of Lines to Step>
```

n[ext]

The n command is used to step through one line of code. It does not step into function calls. Example usage -

```
n <Optional: Number of Lines to Step>
```

p[rint]

The p command is used to print the value of any variable or memory address. Example usage -

```
p <Variable Name/Address/Pointer Chain>
```

c[ontinue]

The c command is used to continue execution of the program until the next breakpoint/error is reached. Example usage -

```
c
```

q[uit]

The q command is used to stop the gdn session. Example usage -

```
q
```