gdb Cheatsheet

Fall 2019

1 Introduction

This document contains a short list of **gdb** commands to help you debug your cs33 programs. The commands contained within this document are by no means exhaustive. Consult the GDB guide, the man pages **(man gdb)** or the internet if you require further information.

How to run gdb: **gdb ./executable [arguments]**
Be sure to recompile your program every time you make changes.

2 GDB Commands

For each of the following commands, bolded text is required (commands and arguments), square brackets are shortcuts, and angle brackets are arguments (non bold ones are optional).

layout <window>	Opens a terminal interface that displays the source file while debugging. The **<window>** can either be **src** to display C code, **asm** for assembly, or **regs** for registers.
focus <window>	Switches focus between windows. The **<window>** parameter can be those supplied to **layout** or **cmd** to change focuns to the command window.
[b]reak <location>	Sets a breakpoint on either a function, a line given by a line number, or the instruction located at a particular address. The **<location>** can be a function name or filename:line# or *memory address.
[d]elete <breakpoint #>	Removes the indicated breakpoint. To see breakpoint numbers, run **info break**, or **i b.**
[cond]ition <breakpoint #> <condition>	Updates the breakpoint indicated by the given number so that it's only hit if `condition` is true. `condition` is expressed in C syntax, and can use variables and functions that are in the scope of the breakpoint.
[i]nfo <about>	Lists information about the argument (`about`), or lists what possible arguments are if none are provided.

[r]un <arg1 arg2 ... argn>	Runs the loaded executable program with program arguments **arg1 ... argn**.
[c]ontinue	Resumes execution of a stopped program, stopping again at the next breakpoint.
[s]tep[i] or **[n]ext[i]**	Steps through a single line of code. **step** steps **into** function calls while **next** skips over them. If **i** is provided, steps over a single instruction as opposed to a line.
[b]ack[t]race	Prints a stack trace, listing each function and its arguments. This does the same thing as the commands **info stack** and **where**.
[f]rame <number> or **up** or **down**	**frame** switches context to a previous frame indexed by **<number>**. To see a list of the current stack frames, use **backtrace**. **up** goes up one frame, and **down** goes down one frame. (especially helpful when using layout)
[q]uit	Quits gdb.
[p]rint <expression>	Prints the value which the indicated **expression** evaluates to. There are various formatting arguments to change how print outputs things.
[x]/<number><format><unit_size> **<address>**	Examines the data located in memory at address. ● **number** optionally indicates that several contiguous elements, beginning at **address**, should be examined. This is very useful for examining the contents of an array. By default, this argument is 1. ● **format** indicates how data should be printed. In most cases, this is the same character that you would use in a call to **printf()**. One exception is the format i, which prints an instruction rather than a decimal integer. ● **unit_size** indicates the size of the data to examine. It can be **[b]ytes**, **[h]alfwords** (2 bytes), **[w]ords**, or **[g]iant** words. By default, this is bytes, which is perfect for examining instructions.