CSC 101 Gdb Basics

1. Introduction

The handout entitled "**Twelve Basic Gdb Debugging Commands**" summarizes the most useful gdb commands. There are two ways to run gdb on CSL machines:

1. directly from the UNIX shell, in a terminal window
2. under Emacs.

2. Running Emacs on a Standard Terminal

Use the UNIX command

```
gdb program
```

where *`program`* is the name of the executable program to be run under gdb control. Gdb will issue a brief message followed by the prompt "`(gdb)`". Then issue any gdb command at the prompt.

When using gdb, you need to refer to line numbers for breakpoint setting and viewing files at the lines output by the "bt" command. To determine line numbers, run your normal standard text editor in a separate window from where gdb is running and use the line-number query and positioning functions of the editor.

3. Running Gdb Under Emacs

The advantage of Emacs-gdb is that debugging commands are run in the context of the source files of the program being debugged. I.e., you do not need to run a another editor in a separate window. You can set a breakpoint by moving to a line in the source program, without having to use the line number explicitly. Also, when gdb breakpoints are reached, Emacs automatically splits the screen into two pains, displays the file in which the breakpoint occurred, and highlights the breakpoint line.

You do not need to use Emacs for any editing to use Emacs-gdb. All you'll use of Emacs are a few commands that integrate it with gdb.

To run emacs from UNIX, type "`emacs`" at the UNIX prompt. To start gdb from Emacs, type the command "`escape-x gdb`". This "`escape-x`" command is a two-key sequence -- type the `escape` key once, then type "`x`". If you inadvertently type the escape key more than once, you'll see a message from Emacs which you can ignore. You can get rid of the message by typing "`control-g`". And in general, if Emacs does something or asks your for something you don't like or understand, type "`control-g`".

Immediately after invoking gdb from within Emacs, it will prompt for the name of the executable file to debug. The prompt appears in the bottommost line of the screen, in the same place that command input appears in vim and similar UNIX text editors. If you have recently run `gcc` to compile your program, `gdb` will assume that the most recently compiled program is what you want to debug. It will in the prompt with the name of this program. If `gdb` guesses the program wrong, erase its guess, and fill in the name of the program you want to debug. Emacs provides the same form of filename completion as the shell, so you can use the `tab` key to complete long filenames in the Emacs prompt line.

After you've entered the name of the program to debug, Emacs will open an edit window where gdb executes. You can type gdb commands in this window just as you would if running gdb in a UNIX terminal window. See the gdb-commands handout for details of some useful command `gdb` commands.

There are *many* Emacs editing commands. In the Linux version of Emacs, you can use the menubar and mouse to perform all basic editing, as well as invoking some of the debugging commands. As noted above, you do not need to perform any text editing to use Emacs gdb -- just type gdb commands and move around with the arrow keys and mouse. If you normally used Emacs as your editor, you can make changes to you program as you normally do, by moving the edit cursor into the program edit window.

There are only two Emacs keyboard commands that you need to know to stay out of trouble, and get your debugging work done:

1. The `control-g` command is universal abort; if you enter any state that you don't like or don't understand, type `control-g`.
2. The `escape-x` command is used to execute any Emacs command by name; for example, we used `escape-x gdb` above to execute gdb under Emacs control;

When you're in the gdb execution window, you can use the normal gdb text commands for controlling execution, such as "r" for run, "bt" for backtrace, etc. The most frequently used gdb commands are described in the "Twelve Debugging Commands" handout.

Finally, there is one Emacs keyboard shortcut that will make using Emacs-gdb easier. It's quite handy but not required, since you can do the same thing by typing in the gdb execution window. The command is `control-x space`, which sets a breakpoint from inside a text file window. Use `control-x space`, as follows:

1. Use the '`Split Window`' command in the '`Files`' menu to divide the window into two segments.
2. Use the '`Open File`' command in the '`Files`' menu to open a .c file in which you wan to set a breakpoint; the prompt for the file name appears in the normal bottommost command line, and again filename completion is available there.
3. When the .c file opens, click the left mouse button on the line at which you wish to set a breakpoint.
4. Type the `control-x space` command.

Gdb will confirm that the breakpoint has been set by echoing a message in the gdb execution window. To continue with gdb commands, click the left mouse button at the bottom of the gdb execution window, and proceed.