GDB in a nutshell

15-213 Spring 2000

Getting in and out of gdb

```
gdb         (starts gdb, files to be debugged can be later loaded)
gdb <file>  (starts gdb with <file> to debug)
gdb -h      (lists command options)

quit        (exits from gdb)
Ctrl-d      (same effect as quit)
Note: Ctrl-c does not exit from gdb, but halts the current command
```

Running programs

```
run         (start a program)
set args <args list>
            (specify the arguments to be used to run the program)
kill        (kill the program)
```

Stopping and Continuing

Breakpoints

```
break <function>  (set a breakpoint at entry to <function>)
break <line num>  (set a breakpoint at specified line number)
break *address    (set a breakpoint at specified address)

clear <function>  (delete the breakpoint set at entry to <function>)
clear <line num>  (delete the breakpoint set at that line number)

delete <num>      (delete the breakpoint with <num>)
delete            (delete all breakpoints)
```

Stepping and Continuing

```
step        (continue running program until contrl reaches a different source line)
stepi       (execute one machine instruction)
stepi <num> (execute <num> machine instructions)

next        (continue to the next source line in the current stack frame)
nexti       (execute one machine instruction, stepping over function)
nexti <num> (execute <num> machine instructions, stepping over function)

continue    (resume execution)
continue <num>  (continue, ignoring this breakpoint <num> of times)

until       (continue running until a source line past the current line,
             in the current stack frame, is reached)
until <loc> (continue running until the specified location is reached
             or the current stack frame returns)

finish      (run until the current function returns)
```

Examining the stack

```
frame <args>          (print out a stack frame)
select-frame <args>   (move from one stack frame to another)

backtrace             (print a backtrace of the entire stack)
where                 (print a backtrace of the entire stack)
```

Examining source files

```
list                  (print out source file lines)

disas                 (dumps a range of memory as machine instructions)
disas <addr>          (the range is around the address)
disas <addr1> <addr2> (the range is between the two addresses)
```

Examining data

```
print/f <exp>         (print out the value of <exp> with format /f)
x/nfu <addr>          (print the contents of <addr> in memory)
                       n: repeat count;
                       f: display format(i, instructions);
                       u: unit size (b, bytes, h, two bytes, w, 4 bytes)
display/f <exp>       (display the value of <exp> every time program stops)
display               (show the auto-displayed items)
delete display <num>  (stop displaying item <num>)
```

Examples:

```
print/a $pc           (print the program counter)
print $esp            (print the stack pointer)
print $eax            (print the contents of %eax)
print/x $eax          (print the contents of %eax as hex)
print/a $eax          (print the contents of %eax as an address)
print/d $eax          (print the contents of %eax as decimal)
print/t $eax          (print the contents of %eax as binary)
print/c $eax          (print the contents of %eax as a character)

print 0x100           (print the decimal representation of a hex value)
print/x 555           (print the hex. representation of a decimal value)

x/6xw 0x12345678      (print 6 words in hex from address 0x12345678)
x/10i <addr>          (print next 10 instructions)

display $eax          (print contents of %eax whenever program stops)
display/i $pc         (print the next machine instruction to be executed)
```

Information commands

```
help
info breakpoints      info display
info registers        info frame
info program          info functions
info variables        info address <symbol>
```