GDB

For : COP 3330.

Object oriented Programming (Using C++)

http://www.compgeom.com/~piyush/teach/3330

Piyush Kumar

GDB

- Lot of tutorials on google.
- Debugging tool, see what happens as your program runs
- Needs the –g flag while compiling
  - This adds additional information (i.e. line numbers) in the binary executable
- Execution:
  - Invoke by typing “gdb <executable name>”
  - E.g.: gdb a.out

Basic Commands

- **quit**
  - Quits gdb. Note that it is NOT exit…
- **run**
  - Run the program; it stops at each break point set by break command
- **x (address)**
  - Examine the CONTENTS of the memory at (address)
- **print (expression)**
  - Print the value of the expression - can be registers or simple equations
- **break (function name or *memory address)**
  - Set breakpoints
  - E.g. : break main
  - break x.cpp:15

Basic Commands

- **continue**
  - Resume execution
- **step (s)**
  - Step to the next line in CODE.
  - *Warning:* If you use the step command while control is within a function that was compiled without debugging information, execution proceeds until control reaches a function that does have debugging information.
- **next (n)**
  - Similar to step
  - This is similar to step, but function calls that appear within the line of code are executed without stopping.
- **disas**
  - Show assembly instructions for the current function

Basic Commands

- **where/bt**
  - Shows the current line and stack backtrace. Very useful for segmentation faults.
- **info registers**
  - Shows the current state of the registers
- **display <var/register>**
  - display the contents of the register/var.
  - E.g. : display count
  - display $eax
- More Commands:
  - http://sources.redhat.com/gdb/onlinedocs/gdb_toc.html
- Graphical User interface: ddd
- Screen shots/movies : http://undo-software.com/undodb_screens.html