The gdb Debugger for IA-32 Assembly Language Programs

gcc -g -o *program* ...	Assemble and link with debugging information
gdb [-d *sourcefiledir*] [-d *sourcefiledir*] ... *program* [*corefile*]	Run gdb from a shell
ESC x gdb [-d *sourcefiledir*] [-d *sourcefiledir*] ... *program* [*corefile*]	Run gdb from xemacs

Miscellaneous	
quit	Exit gdb.
directory [*dir1*] [*dir2*] ...	Add directories *dir1*, *dir2*, ... to the list of directories searched for source files, or clear the directory list.
help [*cmd*]	Print a description command *cmd*

Running the Program	
run [*arg1*],[*arg2*] …	Run the program with command-line arguments *arg1*, *arg2*, ...
set args *arg1 arg2 ...*	Set program's the command-line arguments to *arg1*, *arg2*, ...
show args	Print the program's command-line arguments.

Using Breakpoints	
info breakpoints	Print a list of all breakpoints.
break *label*	Set a breakpoint at the memory address denoted by *label*.
break *fn*	Set a breakpoint at the third instruction of function *fn*.
condition *bpnum expr*	Break at breakpoint *bpnum* only if expression *expr* is non-zero (TRUE).
commands [*bpnum*] *cmd1 cmd2 ...*	Execute commands *cmd1*, *cmd2*, ... whenever breakpoint *bpnum* (or the current breakpoint) is hit.
continue	Continue executing the program.
kill	Stop executing the program.
delete [*bpnum1*][,*bpnum2*]...	Delete breakpoints *bpnum1*, *bpnum2*, ..., or all breakpoints.
clear [**addr*]	Clear the breakpoint at memory address *addr*, or the current breakpoint.
clear [*fn*]	Clear the breakpoint at function *fn*, or the current breakpoint.
disable [*bpnum1*][,*bpnum2*]...	Disable breakpoints *bpnum1*, *bpnum2*, ..., or all breakpoints.
enable [*bpnum1*][,*bpnum2*]...	Enable breakpoints *bpnum1*, *bpnum2*, ..., or all breakpoints.

Stepping through the Program	
next	"Step over" the next instruction.
step	"Step into" the next instruction.
finish	"Step out" of the current function.

Examining Registers and Memory	
info registers	Print the contents of all registers.
print/*f* $*reg*	Print the contents of register *reg* using format *f*. The format can be x (hexadecimal), d (decimal), u (unsigned decimal), o (octal), a (address), c (character), or f (floating point).
print/*f label*	Print the contents of memory at the address denoted by *label* using format *f*.
x/*rsf addr*	Examine the contents of memory at address *addr* using repeat count *r*, size *s*, and format *f*. The repeat count is optional; it defaults to 1. The size is optional; it can be b (byte), h (halfword), w (word), or g (double word). The format can be x (hexadecimal), d (decimal), u (unsigned decimal), o (octal), a (address), c (character), f (floating point), s (string), or i (instruction).
x/*rsf* $*reg*	Examine the contents of memory at the address contained in register *reg*.
info display	Print the display list.
display/*f* $*reg*	At each break, print the contents of register *reg* using format *f* (as with a print command).
display/si *addr*	At each break, print the contents of memory at address *addr* using size *s* (as with an x command).
display/ss *addr*	At each break, print the string of size *s* that begins in memory at address *addr* (as with an x command).
undisplay *displaynum*	Remove *displaynum* from the display list

Examining the Call Stack	
where	Print the call stack.
backtrace	Print the call stack.
frame	Print the top of the call stack.
up	Move the context toward the bottom of the call stack.
down	Move the context toward the top of the call stack.