GDB commands

When gdb starts, your program is not actually running. It won't run until you tell gdb how to run it. Whenever the prompt appears, you have all the commands on the quick reference sheet available to you.

- **run** *command-line-arguments*

  Starts your program as if you had typed

  ```
  a.out command-line arguments
  ```

  or you can do the following

  ```
  a.out < somefile
  ```

  to pipe a file as standard input to your program

- **break** *place*

  Creates a breakpoint; the program will halt when it gets there. The most common breakpoints are at the beginnings of functions, as in

  ```
  (gdb) break Traverse
  Breakpoint 2 at 0x2290: file main.c, line 20
  ```

  The command **break main** stops at the beginning of execution. You can also set breakpoints at a particular line in a source file:

  ```
  (gdb) break 20
  Breakpoint 2 at 0x2290: file main.c, line 20
  ```

  When you run your program and it hits a breakpoint, you'll get a message and prompt like this.

  ```
  Breakpoint 1, Traverse(head=0x6110, NumNodes=4)
    at main.c:16
  (gdb)
  ```

- **delete** *N*

Removes breakpoint number *N*. Leave off *N* to remove all breakpoints. **info break** gives info about each breakpoint

- **help** *command*

  Provides a brief description of a GDB command or topic. Plain **help** lists the possible topics

- **step**

  Executes the current line of the program and stops on the next statement to be executed

- **next**

  Like **step**, however, if the current line of the program contains a function call, it executes the function and stops at the next line.

- **step** would put you at the beginning of the function
- **finish**

  Keeps doing **nexts**, without stepping, until reaching the end of the current function

- **Continue**

  Continues regular execution of the program until a breakpoint is hit or the program stops

- **file** *filename*

  Reloads the debugging info. You need to do this if you are debugging under emacs, and you recompile in a different executable. You **MUST** tell gdb to load the new file, or else you will keep trying to debug the old program, and this will drive you crazy

- **where**

  Produces a backtrace - the chain of function calls that brought the program to its current place. The command **backtrace** is equivalent

- **print** *E*

  prints the value of *E* in the current framein the program, where *E* is a C expression (usually just a variable). **display** is similar, except every time you

execute a next or step, it will print out the expression based on the new variable values

- **quit**

  Leave GDB. If you are running gdb under emacs,

  `C-x 0`

  will get you just your code back

The goal of gdb is to give you enough info to pinpoint where your program crashes, and find the bad pointer that is the cause of the problem. Although the actual error probably occurred much earlier in the program, figuring out which variable is causing trouble is a big step in the right direction. Before you seek help from a TA or preceptor, you should try to figure out *where*your error is occurring

Tham khảo tại trang web:
http://www.cs.princeton.edu/~benjasik/gdb/gdbtut.html

Ví dụ:
Để xem giá trị biến **type** trong thủ tục **ExceptionHandler** trong file **exception.cc** khi mà chương trình đang thực thi thì chúng ta làm như sau.

Tại thư mục nachos-3.4/code
% gdb ./userprog/nachos
(gdb) break ExceptionHandler(ExceptionType) //*tạo break point tại hàm ExceptionHandler*
(gdb) run –rs 1234 –x ./test/halt
(gdb) next *//thực thi lệnh kế tiếp, nhảy qua hàm, nếu gọi lệnh* ***step*** *thì gdb sẽ vào bên //trong hàm*
(gdb) display type *//hoặc print type*
(gdb) continue *// thực thi tới break point kế tiếp hoặc kết thúc chương trình*
(gdb) quit

Ví dụ: Nếu chương trình thực thi mà gặp lỗi (run-time error) phải kết thúc giữa chừng. Chúng ta muốn debug xem chương trình đã kết thúc ở tại thủ tục nào thì chúng ta có thể làm như sau:
% gdb ./userprog/nachos
(gdb) run –rs 1234 –x ./test/<yourprogram>
... chương trình bị kết thúc bất ngờ
(gdb)bt *//=* ***where*** *=* ***backtrace*** *hiển thị ra vị trí thứ tự các thủ tục đã được gọi cho đến lúc chương trình bị kết thúc*