Some useful commands

Basic Linux Commands

- pwd – present working directory
- ls – to list all files and directories of current directory(or specified directory)
- cd – for change directory in linux file system
- mkdir – to create a directory
- rmdir – to remove a directory
- touch – to create a file
- rm – to remove files and directories
- cat – to see the content of a file
- cp – to copy a file
- mv – to rename or move a file
- grep – to search a text
- man – used to get help on commands and C functions
  syntax : man <command/function_name>
- chmod - is used to change the access permission of files and directories)
  * to add the permission use '+' followed by permission
    example:
        chmod +w <file_name>
  * to remove the permission use '-' followed by permission

If you are not very familiar with Linux commands, read Chapter 4 of Running Linux by Dalheimer et al.

Compiling and Running C program

- Open terminal
- Create <file_name>.c file and type the C program in that file, save it.
- In your Terminal, enter the following command in order to make an executable version of the program you have written:

```
$ gcc [programName].c -o programName
For example,  $ gcc firstProgram.c -o firstProgram
```

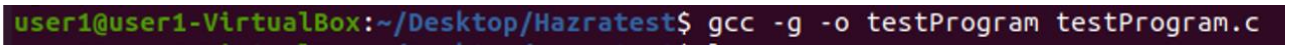
  -o flag writes the build output to an output file.

- If there is no error, then an executable file with name < programName > will be created in the current directory.
- To run the file type command './<programName>' (without quotation). For example,
- `$ ./firstProgram`

Debugging in C using gdb

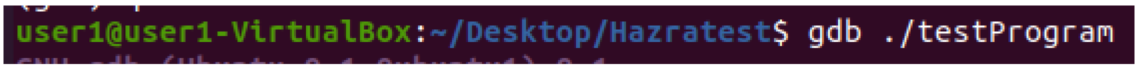
gdb stands for GNU Project Debugger and is a powerful debugging tool for C. You will need executable files which are binary files produced by compilation step.

To prepare your program for debugging with gdb, you must compile it with the -g flag. g flag means you can see the names of variables and functions in your stack frames, get line numbers and see the source as you check in the executable.



```
user1@user1-VirtualBox:~/Desktop/Hazratest$ gcc -g -o testProgram testProgram.c
```

Then, run gdb with the generated executable.

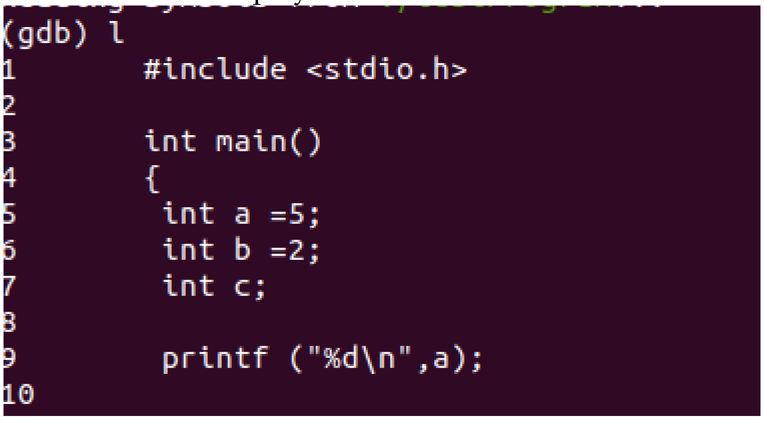


```
user1@user1-VirtualBox:~/Desktop/Hazratest$ gdb ./testProgram
```

If you want to exit then type quit or q.

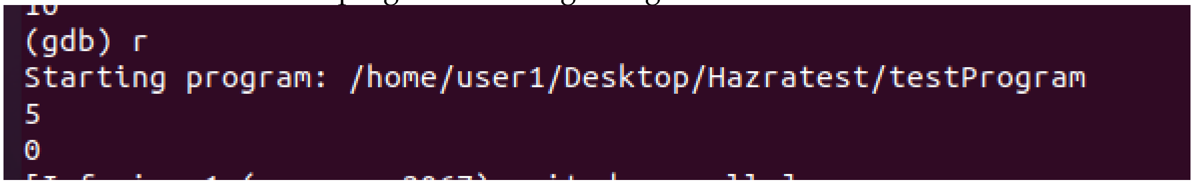
Below are few useful commands to get started with gdb

- List or l – to display the source code



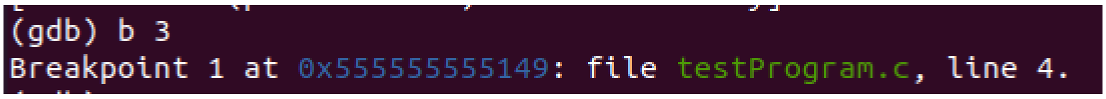
```
(gdb) l
1       #include <stdio.h>
2
3       int main()
4       {
5        int a =5;
6        int b =2;
7        int c;
8
9        printf ("%d\n",a);
10
```

2. run or r - executes the program from beginning to end.



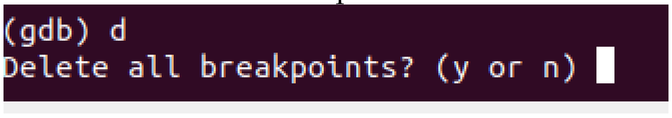
```
(gdb) r
Starting program: /home/user1/Desktop/Hazratest/testProgram
5
0
```

3. break or b - to sets a breakpoint on a particular line.



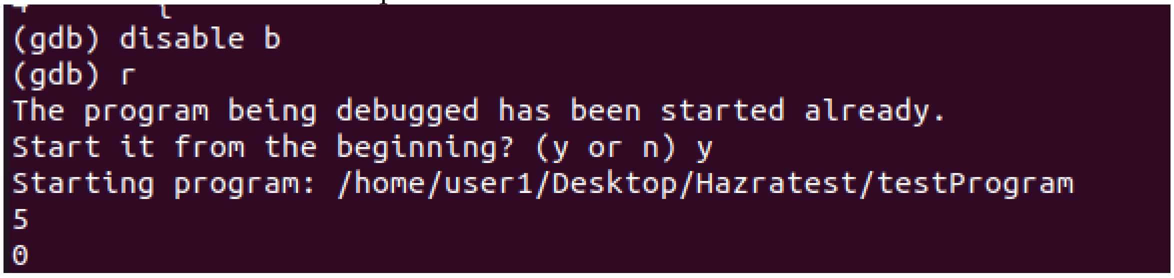
```
(gdb) b 3
Breakpoint 1 at 0x555555555149: file testProgram.c, line 4.
```

4. d - to delete all breakpoints



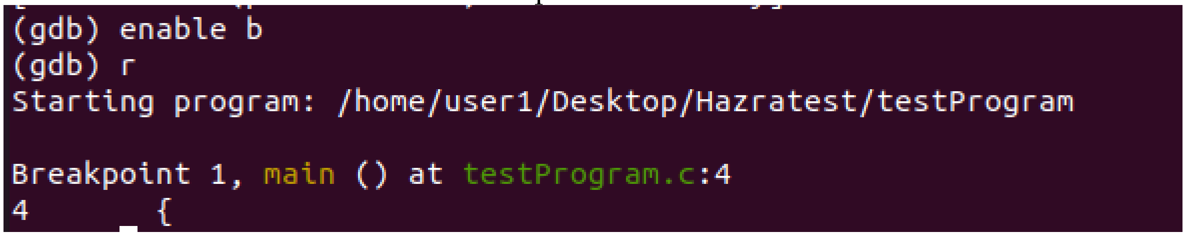
```
(gdb) d
Delete all breakpoints? (y or n)
```

5. disable b - to disable breakpoints



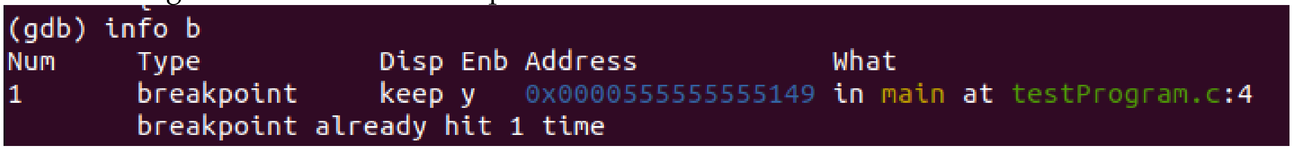
```
(gdb) disable b
(gdb) r
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/user1/Desktop/Hazratest/testProgram
5
0
```

6. enable b – to enable the disabled breakpoints



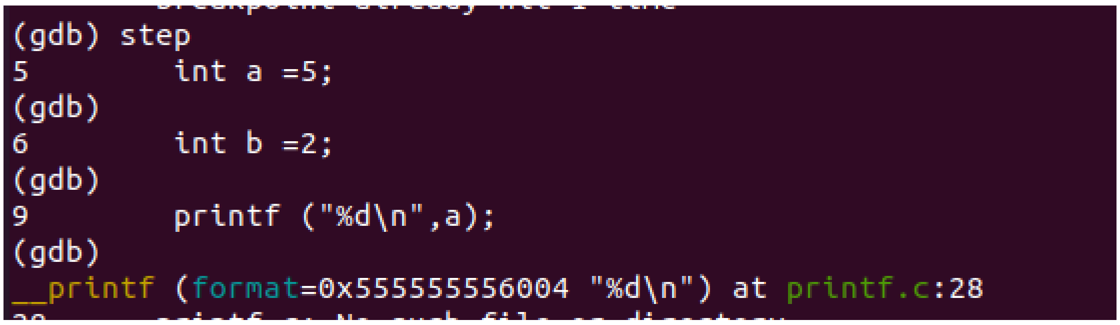
```
(gdb) enable b
(gdb) r
Starting program: /home/user1/Desktop/Hazratest/testProgram

Breakpoint 1, main () at testProgram.c:4
4       {
```

7. info b - to get information on breakpoints



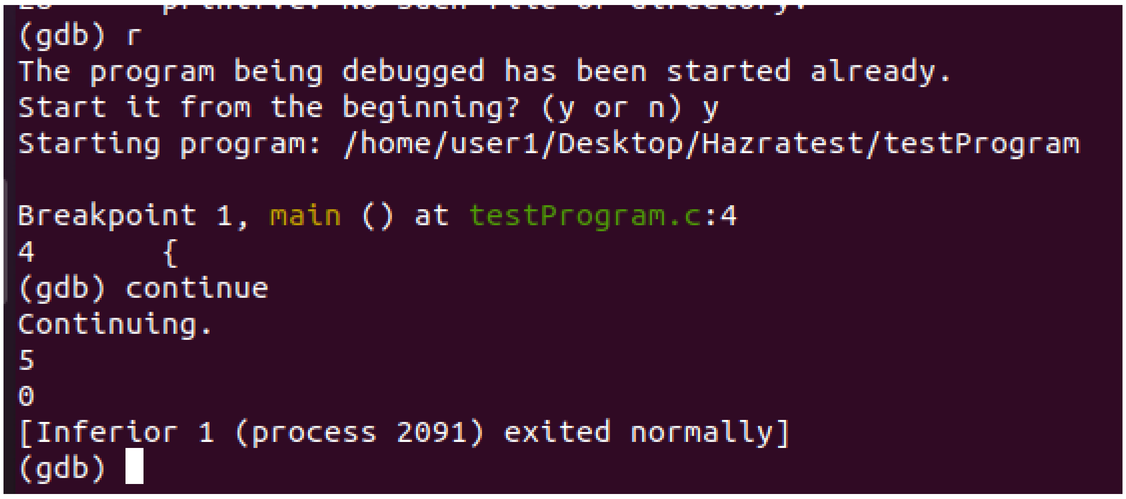
```
(gdb) info b
Num     Type           Disp Enb Address            What
1       breakpoint     keep y   0x0000555555555149 in main at testProgram.c:4
        breakpoint already hit 1 time
```

8. step - will execute the current source line, and then stop execution again before the next source line.



```
(gdb) step
5         int a =5;
(gdb)
6         int b =2;
(gdb)
9         printf ("%d\n",a);
(gdb)
__printf (format=0x555555556004 "%d\n") at printf.c:28
```

9. continue - will set the program running again, after you have stopped it at a breakpoint.



```
(gdb) r
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/user1/Desktop/Hazratest/testProgram

Breakpoint 1, main () at testProgram.c:4
4       {
(gdb) continue
Continuing.
5
0
[Inferior 1 (process 2091) exited normally]
(gdb) 
```

10. help – for online documentation.

Review the pointer in C