Week 12 Lecture 3

Debugging with GDB

Gnu Debugger (gdb)

gdb

- The Gnu Debugger terminal command (gdb)
  - It allows you to track the execution of a program
  - You can see which function a segmentation violation took place.
  - You can inspect the values of variables in the running code

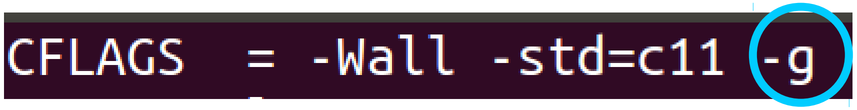
TDD minimizes debugging

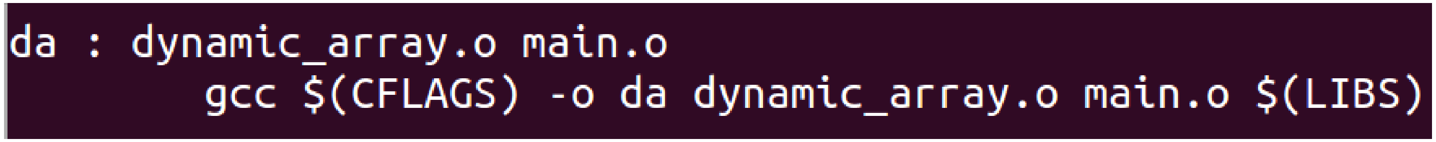
- By testing every few lines; you can minimize the need to debug
- It is usually clear where in the two or three lines you are testing, the error occurred.
- Sometimes, especially with pointers, bugs may hide until many lines are written

Debugger

- Debugger does not help with compile error
- It makes visible run time elements, such as:
  - The value of variables
    - Can use printf
  - The value of pointers
    - Following them will cause a segmentation fault
  - The flow of control
    - See which lines are really executed

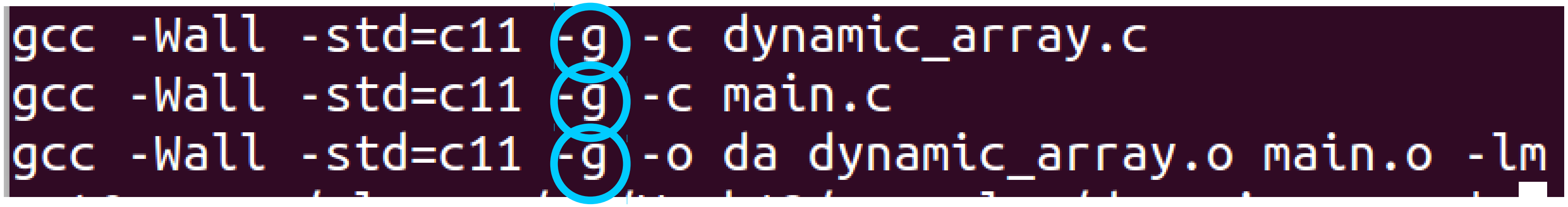
To use GDB compile with -g

- Makefile is set up to automatically compile with -g
  - 

    ```
    CFLAGS  = -Wall -std=c11 -g
    ```
  - 

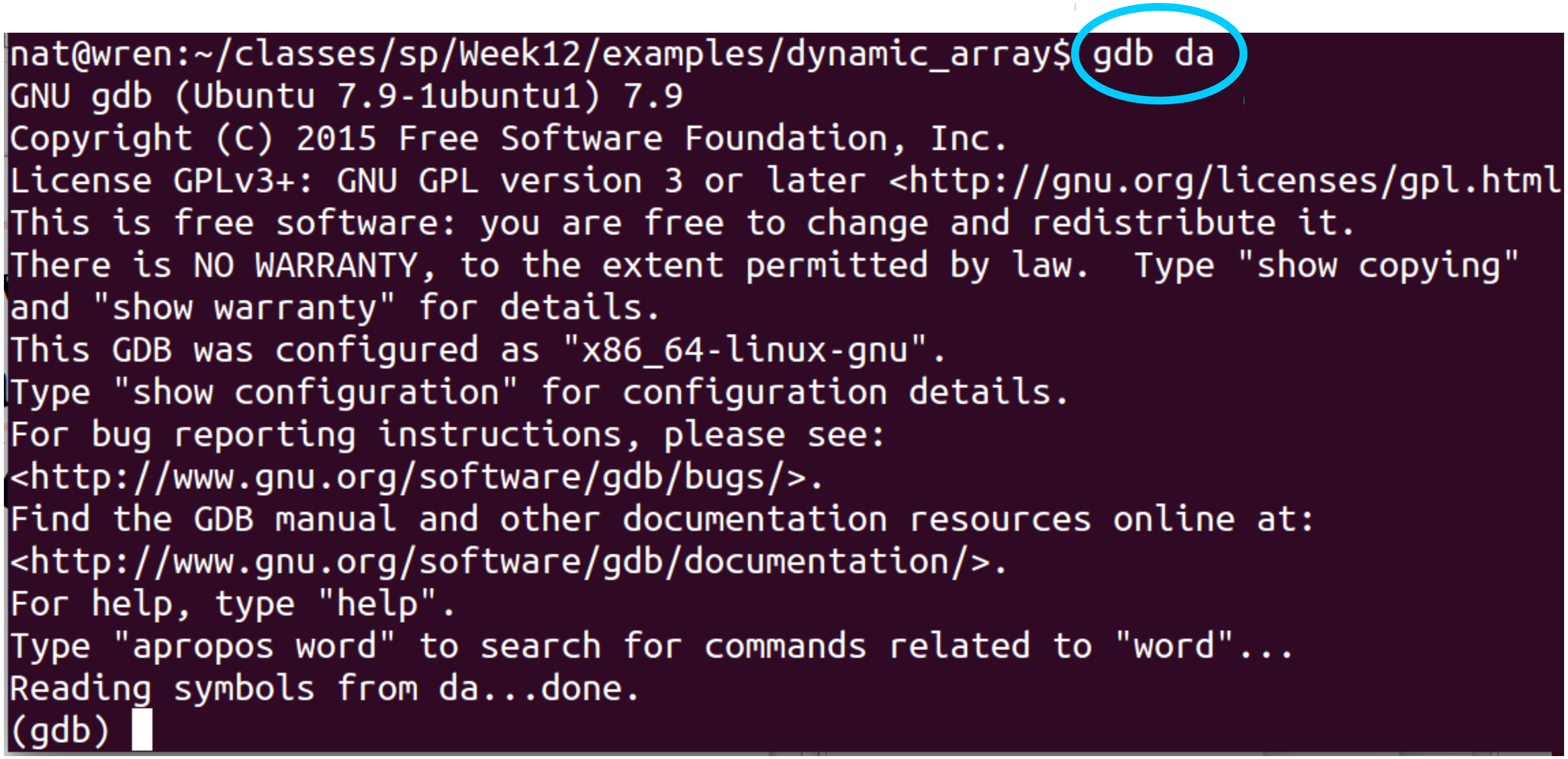
    ```
    da : dynamic_array.o main.o
            gcc $(CFLAGS) -o da dynamic_array.o main.o $(LIBS)
    ```

- Output when compiling



```
gcc -Wall -std=c11 -g -c dynamic_array.c
gcc -Wall -std=c11 -g -c main.c
gcc -Wall -std=c11 -g -o da dynamic_array.o main.o -lm
```

Run gdb with program as parameter

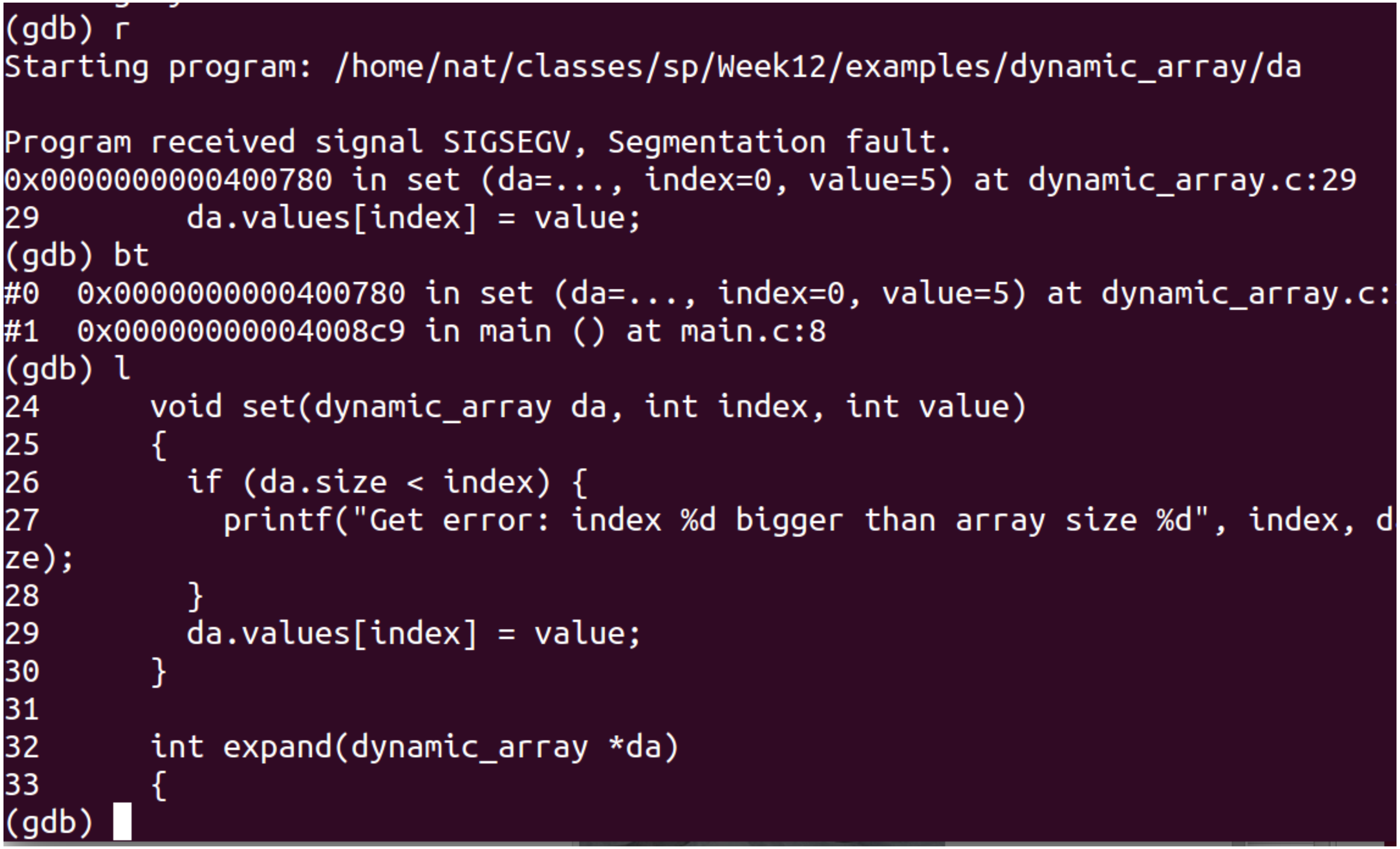


```
nat@wren:~/classes/sp/Week12/examples/dynamic_array$ gdb da
GNU gdb (Ubuntu 7.9-1ubuntu1) 7.9
Copyright (C) 2015 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from da...done.
(gdb) █
```

Commands

- Run the program: r
- Trace back through the function calls: bt
- List the most recent 10 line: l
  - List n lines: l n

Example (r, bt,l)



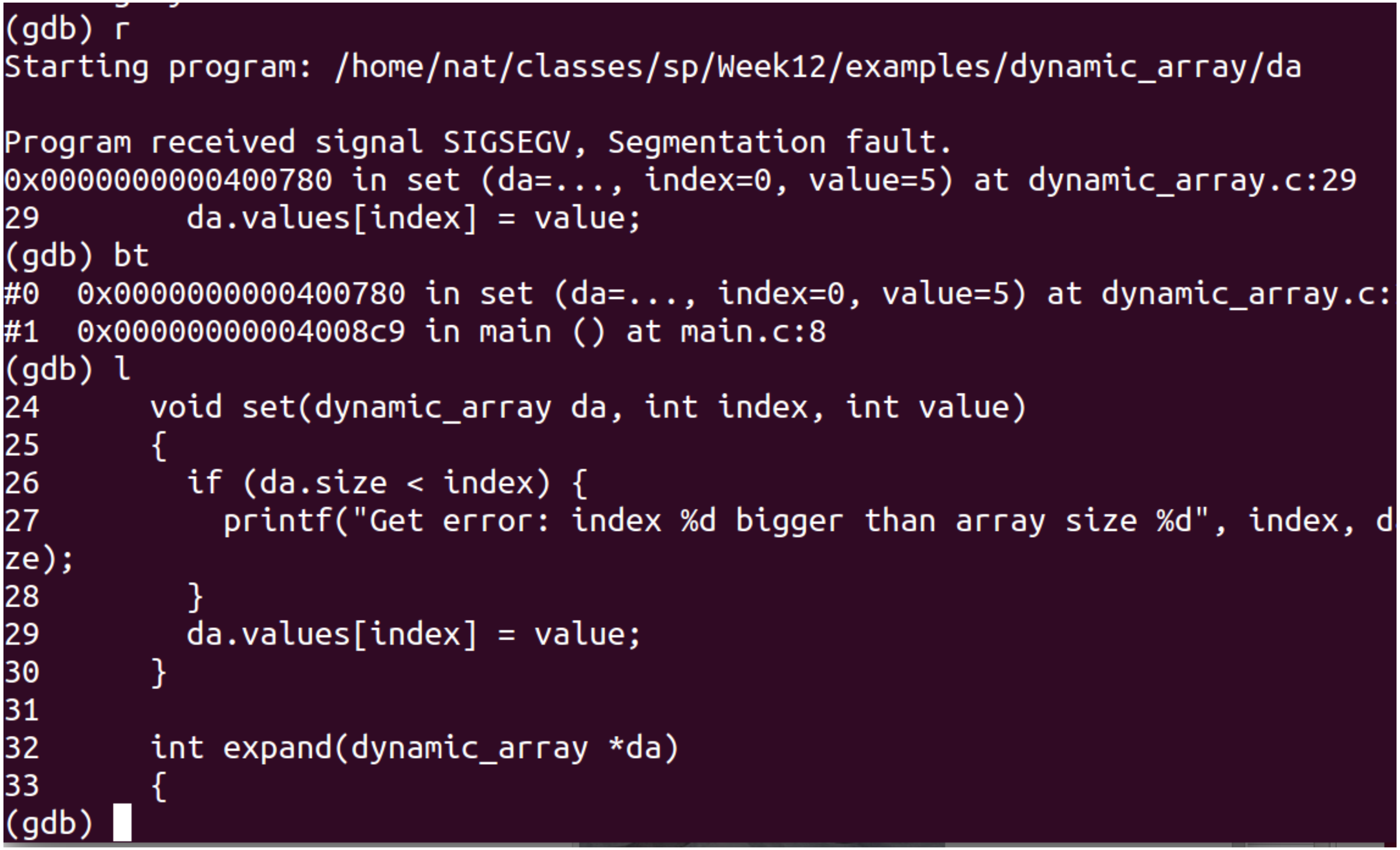
```
(gdb) r
Starting program: /home/nat/classes/sp/Week12/examples/dynamic_array/da

Program received signal SIGSEGV, Segmentation fault.
0x0000000000400780 in set (da=..., index=0, value=5) at dynamic_array.c:29
29          da.values[index] = value;
(gdb) bt
#0  0x0000000000400780 in set (da=..., index=0, value=5) at dynamic_array.c:
#1  0x00000000004008c9 in main () at main.c:8
(gdb) l
24        void set(dynamic_array da, int index, int value)
25        {
26          if (da.size < index) {
27            printf("Get error: index %d bigger than array size %d", index, d
ze);
28          }
29          da.values[index] = value;
30        }
31
32        int expand(dynamic_array *da)
33        {
(gdb) 
```

More Commands

- Display content of <var>: p <var>
- Break at current line: b
- Break in n more lines: b n
- Break at program start: b main
- Next line (including functions): n
- Next line (stepping over funcstions): s
- Quit gdb: q

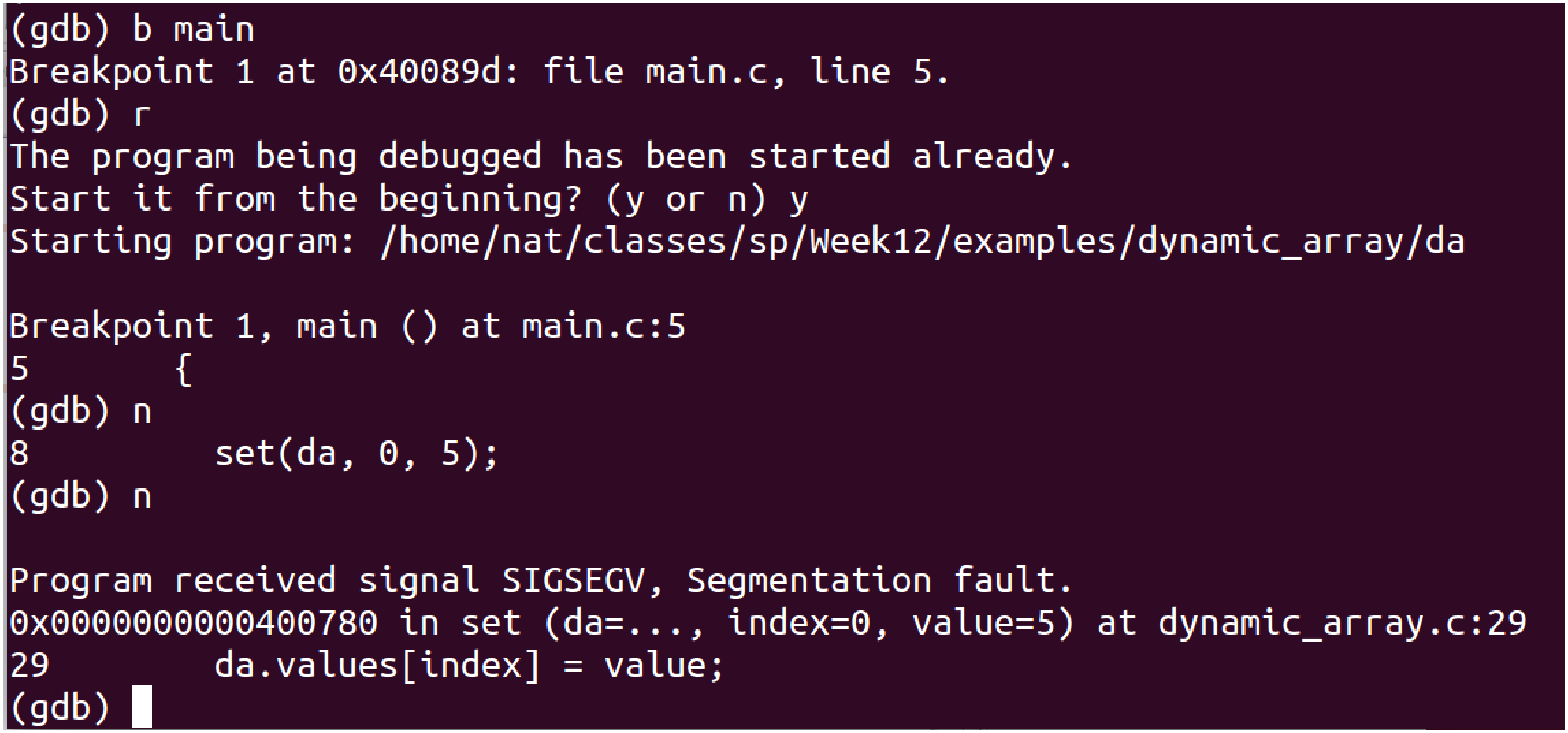
Example (p)



```
(gdb) r
Starting program: /home/nat/classes/sp/Week12/examples/dynamic_array/da

Program received signal SIGSEGV, Segmentation fault.
0x0000000000400780 in set (da=..., index=0, value=5) at dynamic_array.c:29
29        da.values[index] = value;
(gdb) bt
#0  0x0000000000400780 in set (da=..., index=0, value=5) at dynamic_array.c:
#1  0x00000000004008c9 in main () at main.c:8
(gdb) l
24      void set(dynamic_array da, int index, int value)
25      {
26        if (da.size < index) {
27          printf("Get error: index %d bigger than array size %d", index, d
ze);
28        }
29        da.values[index] = value;
30      }
31
32      int expand(dynamic_array *da)
33      {
(gdb) 
```

Example (b, n)



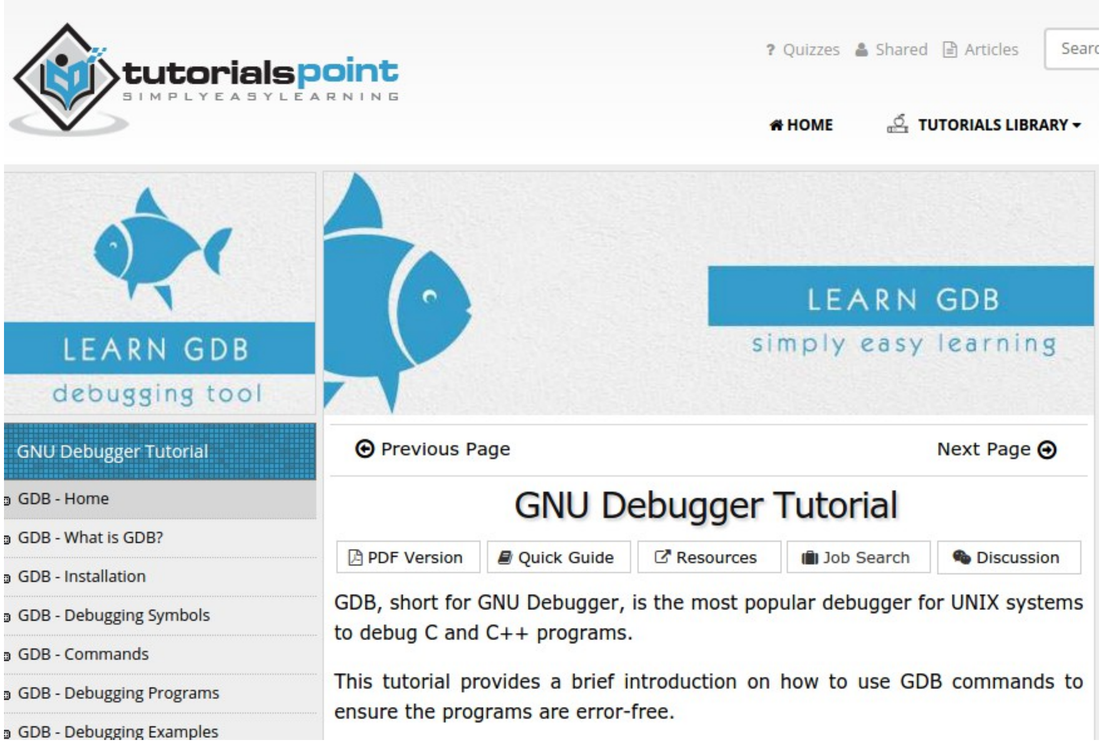
```
(gdb) b main
Breakpoint 1 at 0x40089d: file main.c, line 5.
(gdb) r
The program being debugged has been started already.
Start it from the beginning? (y or n) y
Starting program: /home/nat/classes/sp/Week12/examples/dynamic_array/da

Breakpoint 1, main () at main.c:5
5       {
(gdb) n
8           set(da, 0, 5);
(gdb) n

Program received signal SIGSEGV, Segmentation fault.
0x0000000000400780 in set (da=..., index=0, value=5) at dynamic_array.c:29
29          da.values[index] = value;
(gdb) 
```

GDB Tutorial

- http://www.tutorialspoint.com/gnu_debugger/index.htm



GNU Debugger Tutorial

GDB, short for GNU Debugger, is the most popular debugger for UNIX systems to debug C and C++ programs.

This tutorial provides a brief introduction on how to use GDB commands to ensure the programs are error-free.