Jeremy Scheuerman

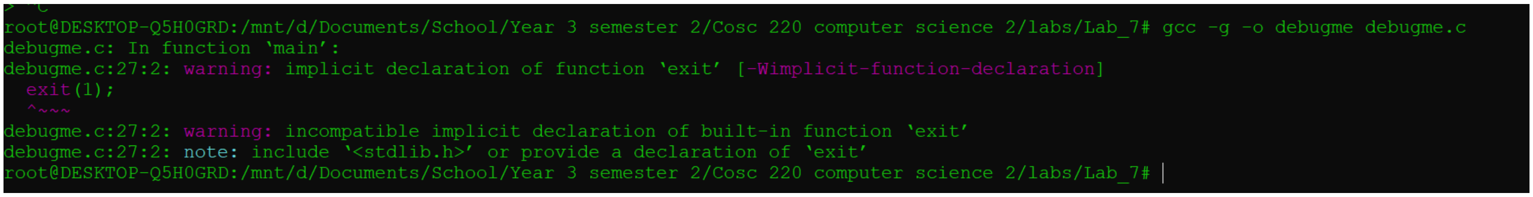
COSC 220

Dr. Wang

9 March 2021

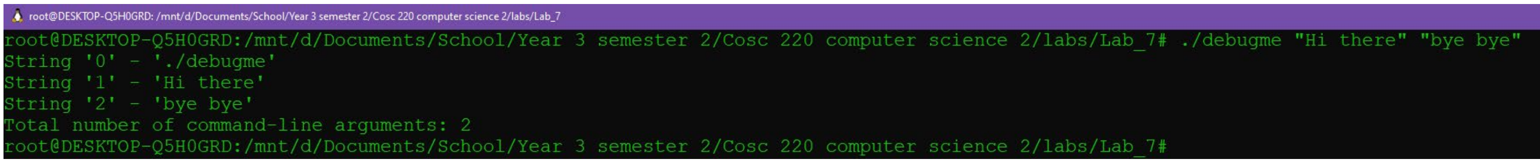
Lab 7 Writeup

Compile debugme



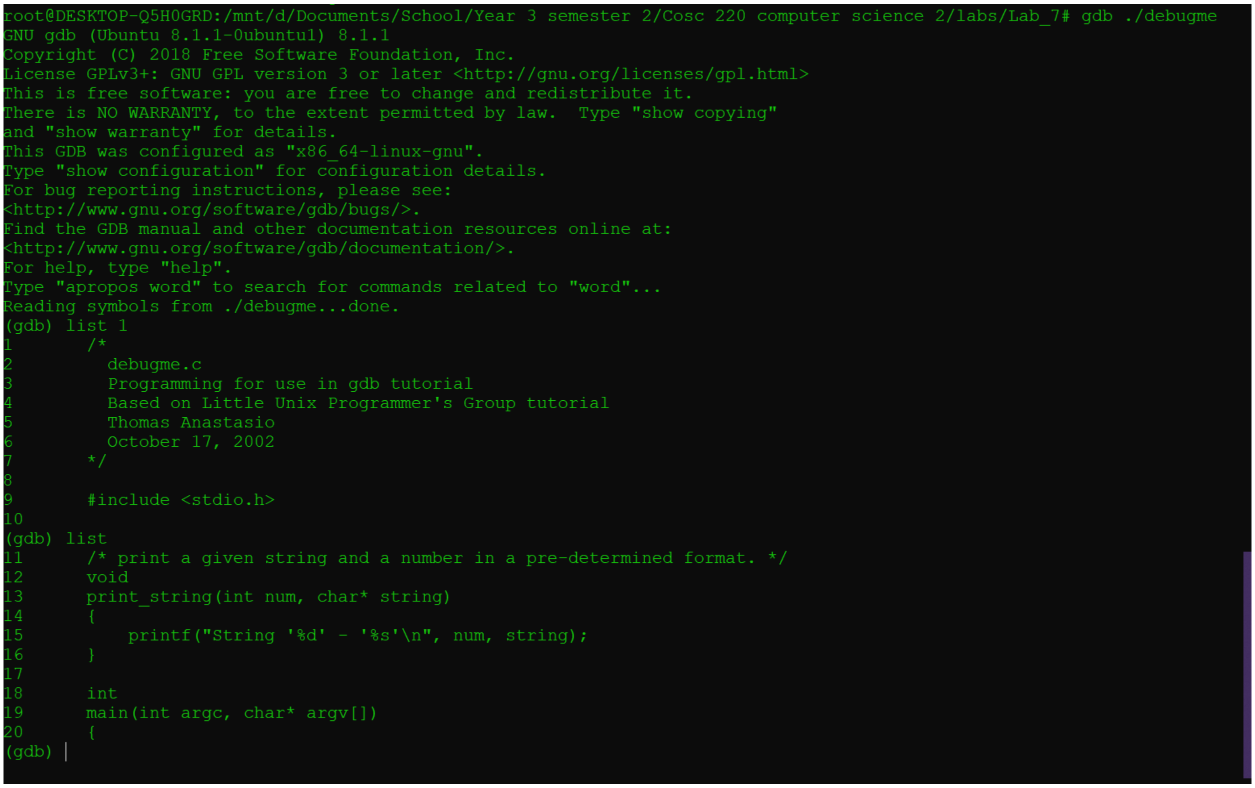
```
> ^C
root@DESKTOP-Q5H0GRD:/mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7# gcc -g -o debugme debugme.c
debugme.c: In function ‘main’:
debugme.c:27:2: warning: implicit declaration of function ‘exit’ [-Wimplicit-function-declaration]
  exit(1);
  ^~~~
debugme.c:27:2: warning: incompatible implicit declaration of built-in function ‘exit’
debugme.c:27:2: note: include ‘<stdlib.h>’ or provide a declaration of ‘exit’
root@DESKTOP-Q5H0GRD:/mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7# |
```

Hi there and Bye Bye



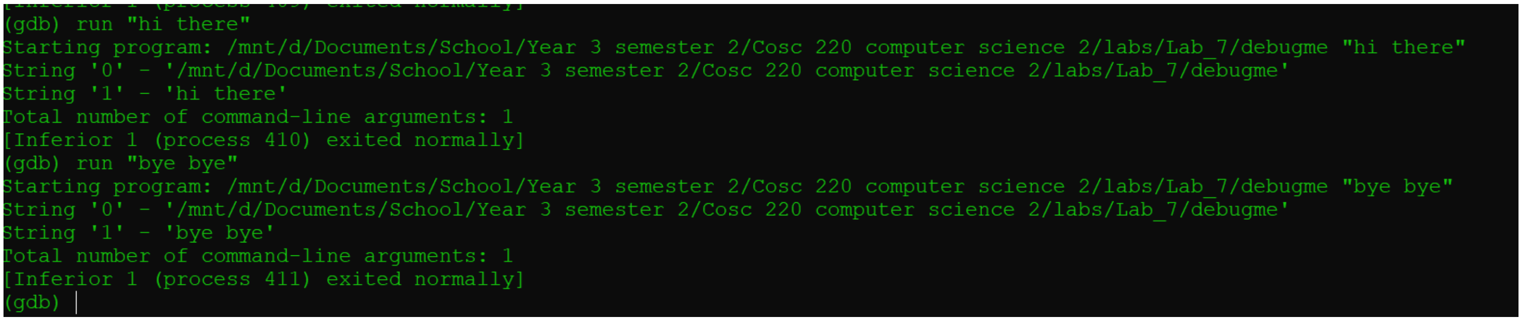
root@DESKTOP-Q5H0GRD: /mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7

```
root@DESKTOP-Q5H0GRD:/mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7# ./debugme "Hi there" "bye bye"
String '0' - './debugme'
String '1' - 'Hi there'
String '2' - 'bye bye'
Total number of command-line arguments: 2
root@DESKTOP-Q5H0GRD:/mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7#
```



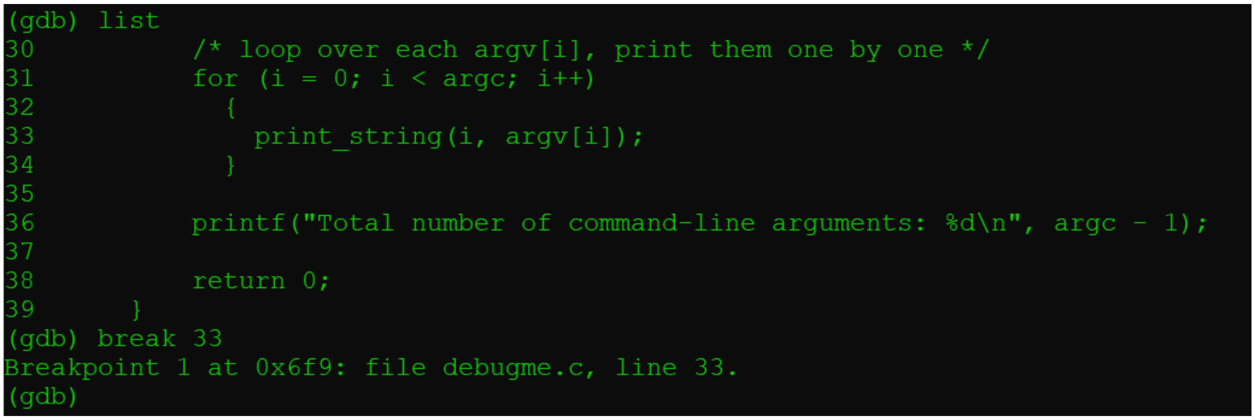
```
root@DESKTOP-Q5H0GRD:/mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7# gdb ./debugme
GNU gdb (Ubuntu 8.1.1-0ubuntu1) 8.1.1
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: 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 ./debugme...done.
(gdb) list 1
1       /*
2         debugme.c
3         Programming for use in gdb tutorial
4         Based on Little Unix Programmer's Group tutorial
5         Thomas Anastasio
6         October 17, 2002
7       */
8
9       #include <stdio.h>
10
(gdb) list
11      /* print a given string and a number in a pre-determined format. */
12      void
13      print_string(int num, char* string)
14      {
15          printf("String '%d' - '%s'\n", num, string);
16      }
17
18      int
19      main(int argc, char* argv[])
20      {
(gdb) |
```

Run “Hi there” and “Bye Bye”



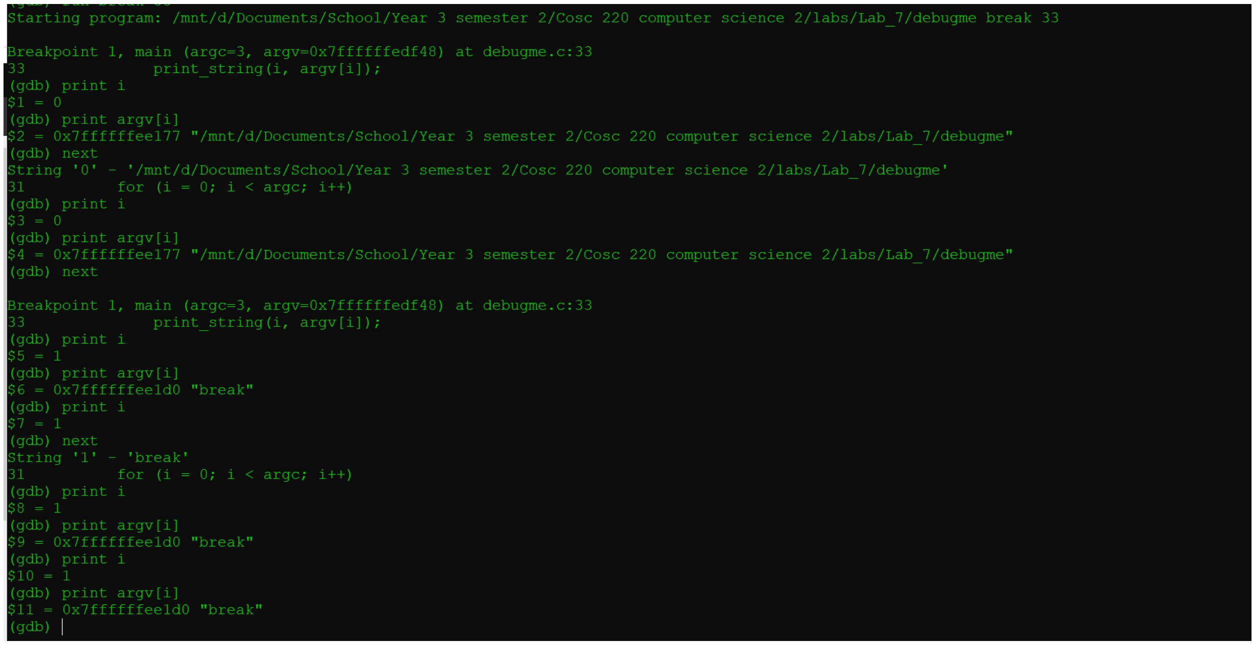
```
[Inferior 1 (process 409) exited normally]
(gdb) run "hi there"
Starting program: /mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7/debugme "hi there"
String '0' - '/mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7/debugme'
String '1' - 'hi there'
Total number of command-line arguments: 1
[Inferior 1 (process 410) exited normally]
(gdb) run "bye bye"
Starting program: /mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7/debugme "bye bye"
String '0' - '/mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7/debugme'
String '1' - 'bye bye'
Total number of command-line arguments: 1
[Inferior 1 (process 411) exited normally]
(gdb) |
```

Breakpoint at print string

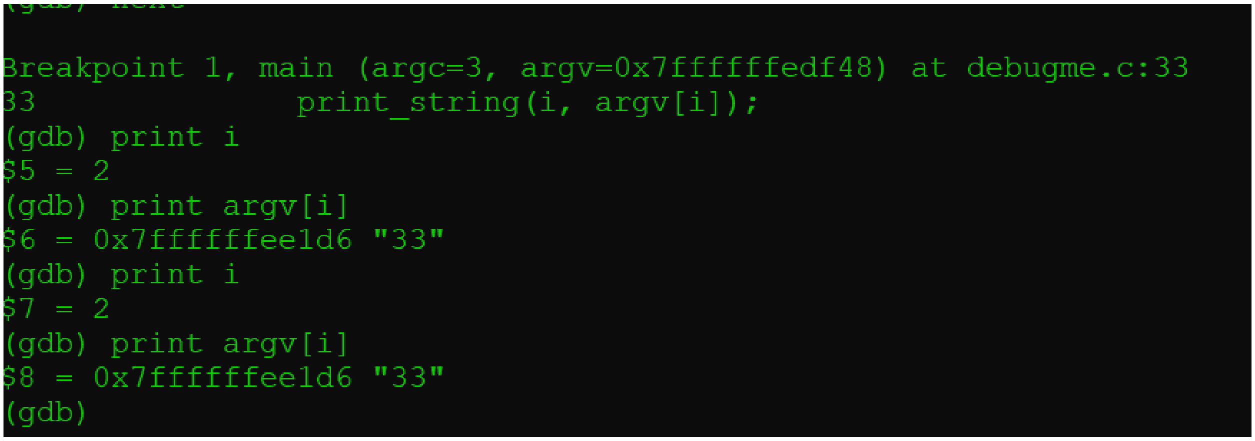


```
(gdb) list
30          /* loop over each argv[i], print them one by one */
31          for (i = 0; i < argc; i++)
32            {
33              print_string(i, argv[i]);
34            }
35
36          printf("Total number of command-line arguments: %d\n", argc - 1);
37
38          return 0;
39      }
(gdb) break 33
Breakpoint 1 at 0x6f9: file debugme.c, line 33.
(gdb)
```

Step through for loop at breakpoint and print I and argv[i]

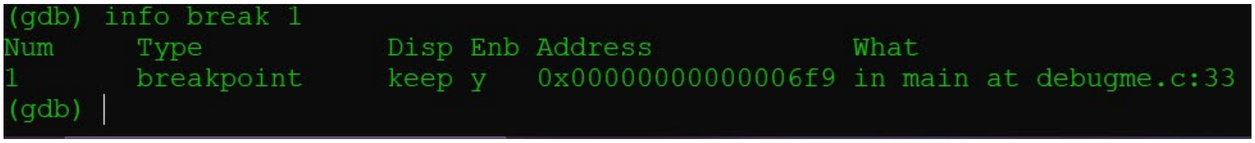


```
(gdb) run break 33
Starting program: /mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7/debugme break 33

Breakpoint 1, main (argc=3, argv=0x7ffffffedf48) at debugme.c:33
33                print_string(i, argv[i]);
(gdb) print i
$1 = 0
(gdb) print argv[i]
$2 = 0x7ffffffee177 "/mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7/debugme"
(gdb) next
String '0' - '/mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7/debugme'
31            for (i = 0; i < argc; i++)
(gdb) print i
$3 = 0
(gdb) print argv[i]
$4 = 0x7ffffffee177 "/mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7/debugme"
(gdb) next

Breakpoint 1, main (argc=3, argv=0x7ffffffedf48) at debugme.c:33
33                print_string(i, argv[i]);
(gdb) print i
$5 = 1
(gdb) print argv[i]
$6 = 0x7ffffffee1d0 "break"
(gdb) print i
$7 = 1
(gdb) next
String '1' - 'break'
31            for (i = 0; i < argc; i++)
(gdb) print i
$8 = 1
(gdb) print argv[i]
$9 = 0x7ffffffee1d0 "break"
(gdb) print i
$10 = 1
(gdb) print argv[i]
$11 = 0x7ffffffee1d0 "break"
(gdb) |
```



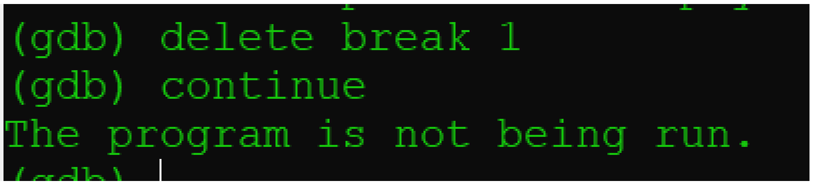
```
(gdb) next

Breakpoint 1, main (argc=3, argv=0x7fffffffedf48) at debugme.c:33
33                print_string(i, argv[i]);
(gdb) print i
$5 = 2
(gdb) print argv[i]
$6 = 0x7ffffffee1d6 "33"
(gdb) print i
$7 = 2
(gdb) print argv[i]
$8 = 0x7ffffffee1d6 "33"
(gdb)
```

Breakpoint info

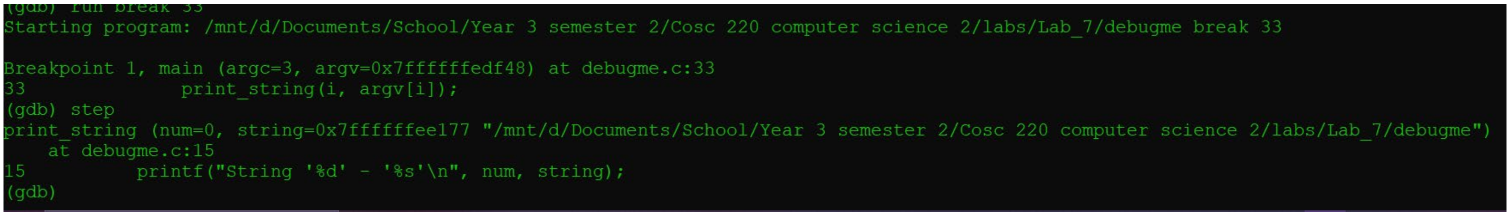


```
(gdb) info break 1
Num     Type           Disp Enb Address            What
1       breakpoint     keep y   0x00000000000006f9 in main at debugme.c:33
(gdb) 
```

Delete breakpoint

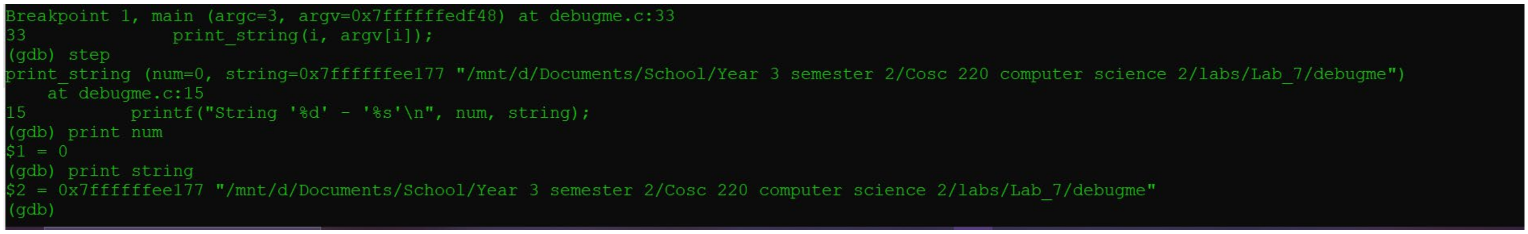


```
(gdb) delete break 1
(gdb) continue
The program is not being run.
(gdb)
```



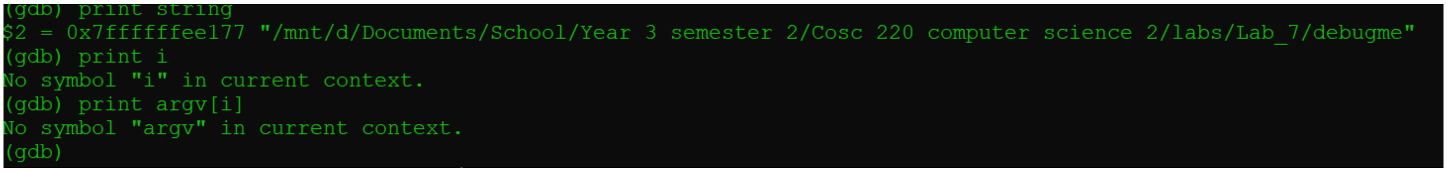
```
(gdb) run break 33
Starting program: /mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7/debugme break 33

Breakpoint 1, main (argc=3, argv=0x7fffffffedf48) at debugme.c:33
33                print_string(i, argv[i]);
(gdb) step
print_string (num=0, string=0x7ffffffee177 "/mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7/debugme")
    at debugme.c:15
15          printf("String '%d' - '%s'\n", num, string);
(gdb)
```

Step through at the breakpoint and print num and string



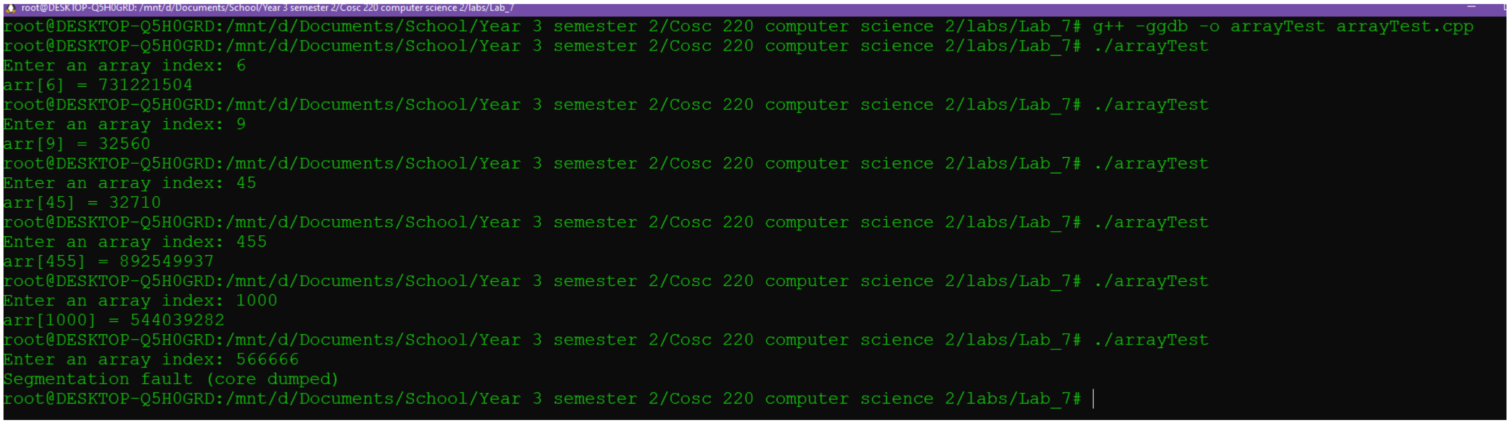
```
Breakpoint 1, main (argc=3, argv=0x7fffffffedf48) at debugme.c:33
33                print_string(i, argv[i]);
(gdb) step
print_string (num=0, string=0x7ffffffee177 "/mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7/debugme")
    at debugme.c:15
15          printf("String '%d' - '%s'\n", num, string);
(gdb) print num
$1 = 0
(gdb) print string
$2 = 0x7ffffffee177 "/mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7/debugme"
(gdb)
```

Print I and Print argv do not work



```
(gdb) print string
$2 = 0x7ffffffee177 "/mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7/debugme"
(gdb) print i
No symbol "i" in current context.
(gdb) print argv[i]
No symbol "argv" in current context.
(gdb)
```

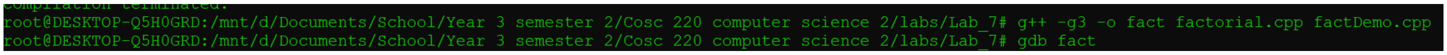
Compile with debug flag -ggdb and enter indexes until segfault



root@DESKTOP-Q5H0GRD: /mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7

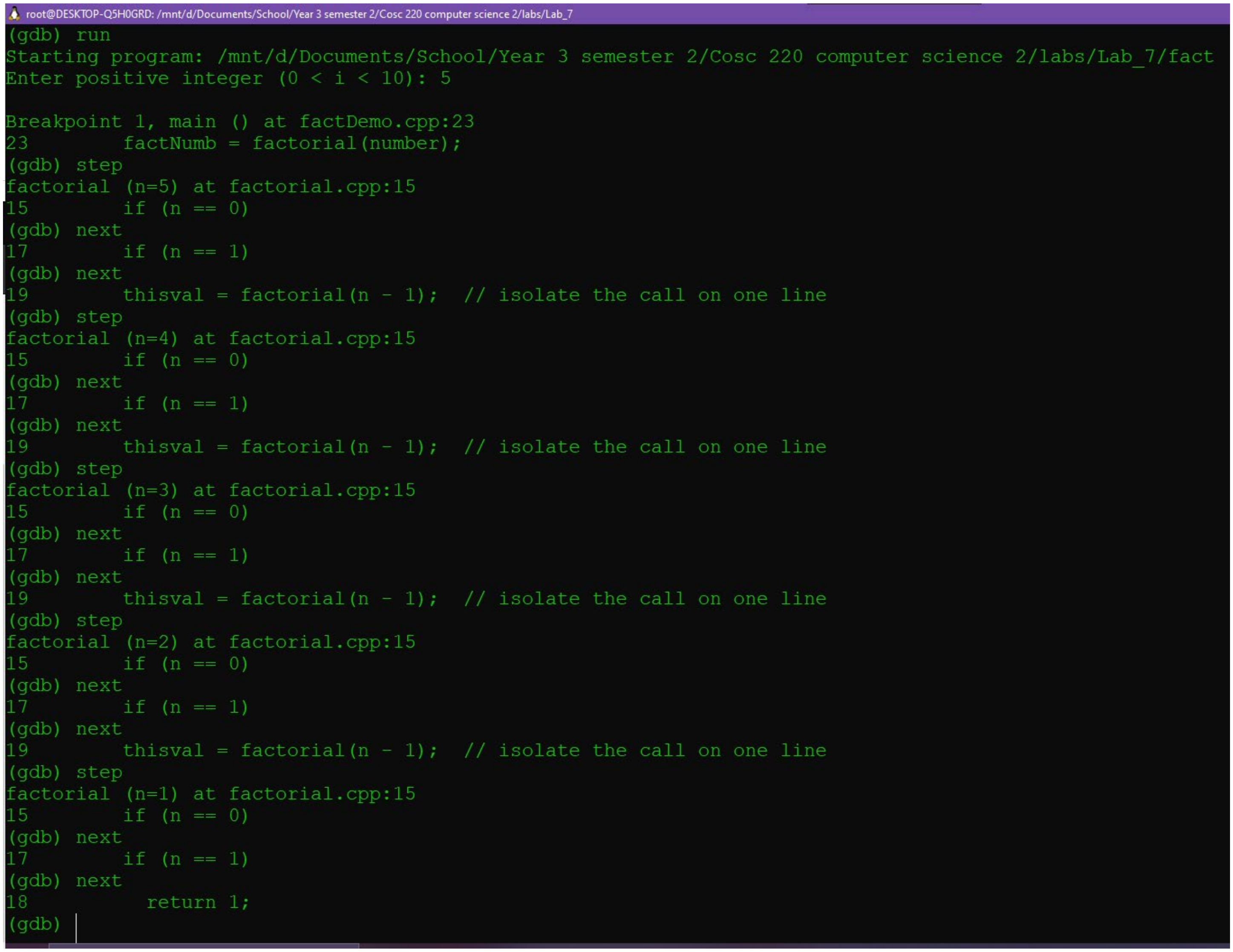
```
root@DESKTOP-Q5H0GRD:/mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7# g++ -ggdb -o arrayTest arrayTest.cpp
root@DESKTOP-Q5H0GRD:/mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7# ./arrayTest
Enter an array index: 6
arr[6] = 731221504
root@DESKTOP-Q5H0GRD:/mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7# ./arrayTest
Enter an array index: 9
arr[9] = 32560
root@DESKTOP-Q5H0GRD:/mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7# ./arrayTest
Enter an array index: 45
arr[45] = 32710
root@DESKTOP-Q5H0GRD:/mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7# ./arrayTest
Enter an array index: 455
arr[455] = 892549937
root@DESKTOP-Q5H0GRD:/mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7# ./arrayTest
Enter an array index: 1000
arr[1000] = 544039282
root@DESKTOP-Q5H0GRD:/mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7# ./arrayTest
Enter an array index: 566666
Segmentation fault (core dumped)
root@DESKTOP-Q5H0GRD:/mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7#
```

Compile and run the factorial



```
compilation terminated.
root@DESKTOP-Q5H0GRD:/mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7# g++ -g3 -o fact factorial.cpp factDemo.cpp
root@DESKTOP-Q5H0GRD:/mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7# gdb fact
```

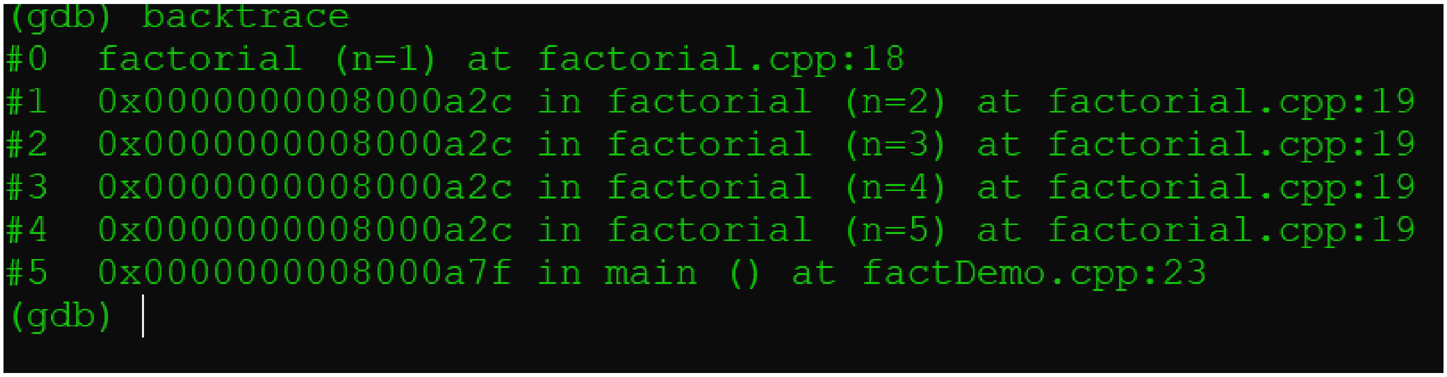
Breakpoint 23 and step through recursively



root@DESKTOP-Q5H0GRD: /mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7

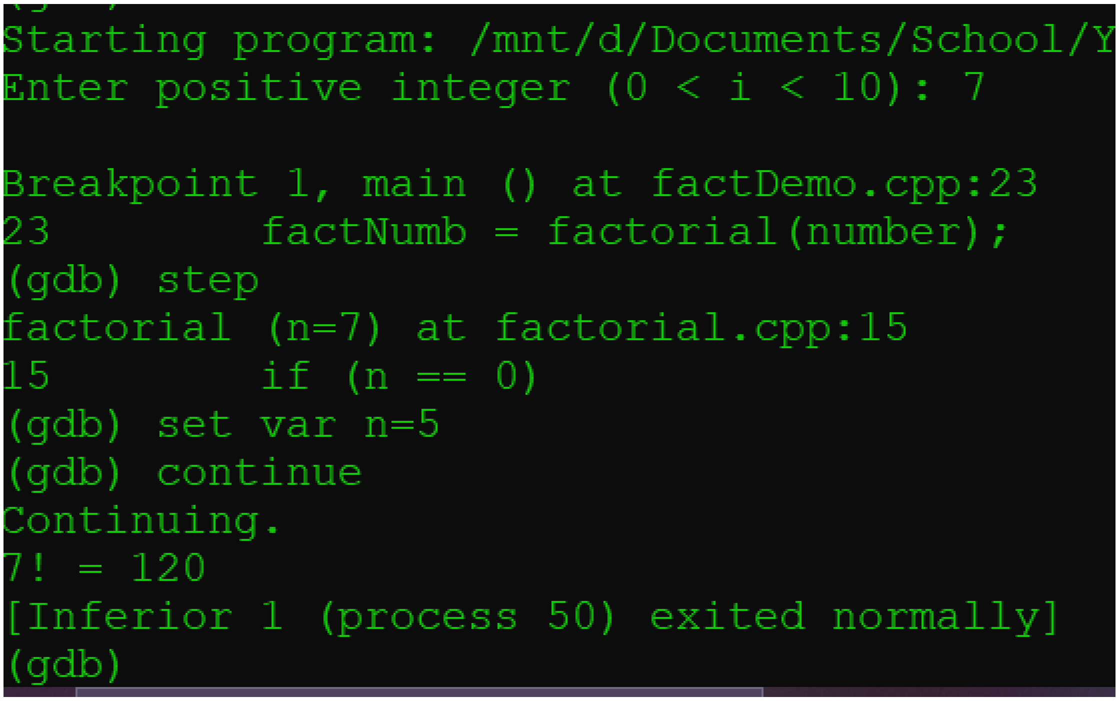
```
(gdb) run
Starting program: /mnt/d/Documents/School/Year 3 semester 2/Cosc 220 computer science 2/labs/Lab_7/fact
Enter positive integer (0 < i < 10): 5

Breakpoint 1, main () at factDemo.cpp:23
23          factNumb = factorial(number);
(gdb) step
factorial (n=5) at factorial.cpp:15
15          if (n == 0)
(gdb) next
17          if (n == 1)
(gdb) next
19          thisval = factorial(n - 1);  // isolate the call on one line
(gdb) step
factorial (n=4) at factorial.cpp:15
15          if (n == 0)
(gdb) next
17          if (n == 1)
(gdb) next
19          thisval = factorial(n - 1);  // isolate the call on one line
(gdb) step
factorial (n=3) at factorial.cpp:15
15          if (n == 0)
(gdb) next
17          if (n == 1)
(gdb) next
19          thisval = factorial(n - 1);  // isolate the call on one line
(gdb) step
factorial (n=2) at factorial.cpp:15
15          if (n == 0)
(gdb) next
17          if (n == 1)
(gdb) next
19          thisval = factorial(n - 1);  // isolate the call on one line
(gdb) step
factorial (n=1) at factorial.cpp:15
15          if (n == 0)
(gdb) next
17          if (n == 1)
(gdb) next
18            return 1;
(gdb)
```

Back trace through the function



```
(gdb) backtrace
#0  factorial (n=1) at factorial.cpp:18
#1  0x0000000008000a2c in factorial (n=2) at factorial.cpp:19
#2  0x0000000008000a2c in factorial (n=3) at factorial.cpp:19
#3  0x0000000008000a2c in factorial (n=4) at factorial.cpp:19
#4  0x0000000008000a2c in factorial (n=5) at factorial.cpp:19
#5  0x0000000008000a7f in main () at factDemo.cpp:23
(gdb)
```

Step into main call print n the n set = to 5



```
Starting program: /mnt/d/Documents/School/Y
Enter positive integer (0 < i < 10): 7

Breakpoint 1, main () at factDemo.cpp:23
23          factNumb = factorial(number);
(gdb) step
factorial (n=7) at factorial.cpp:15
15          if (n == 0)
(gdb) set var n=5
(gdb) continue
Continuing.
7! = 120
[Inferior 1 (process 50) exited normally]
(gdb)
```