CMPSC 297 - Introduction to C Programming

PennState

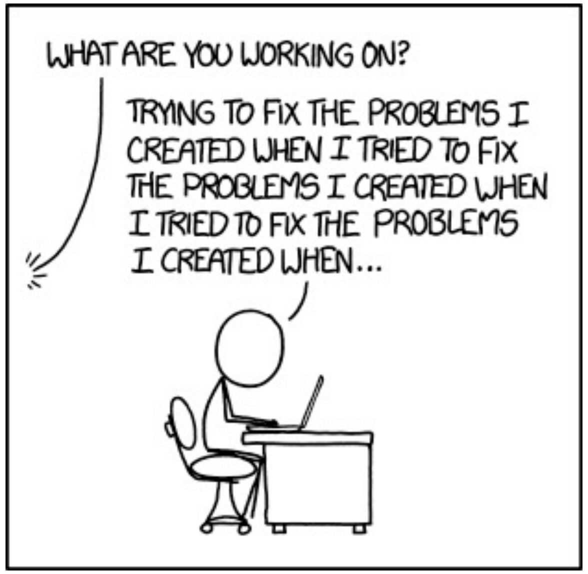
Systems and Internet Infrastructure Security Labratory

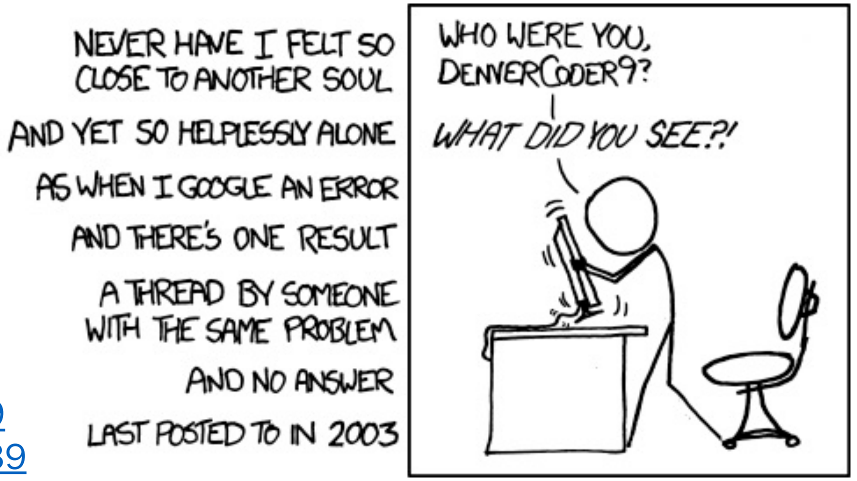
Week #2 – Reading and Fixing Errors - Debugging

Professor Patrick McDaniel

Debugging - Why is it important?

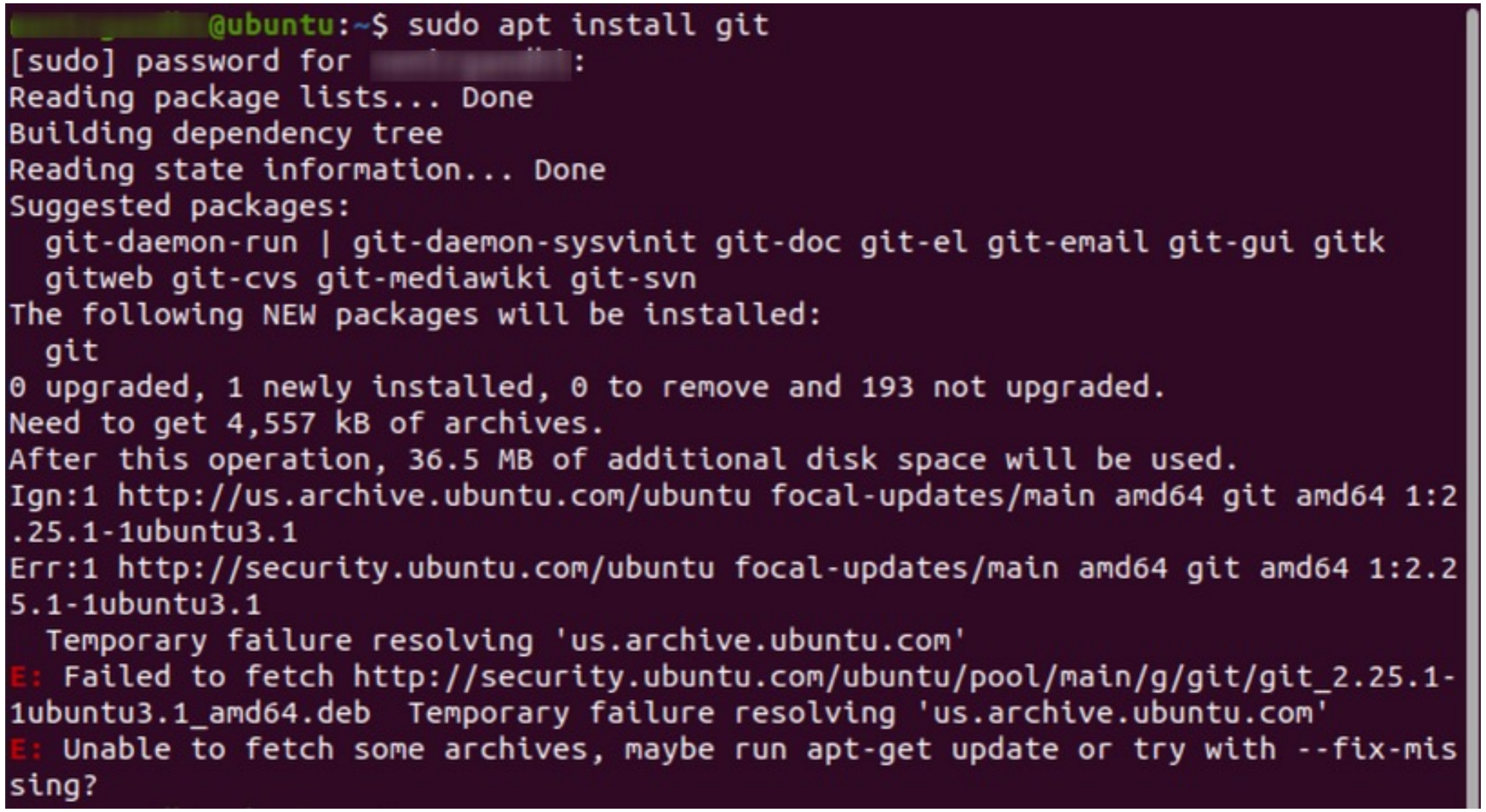
- +50% of the developer time is spent on debugging
- Analyzing errors and being able to fix them is what is going to make your task easier, done faster, etc.
- Unique skills + efficiency = possible career evolution
- You do not want to get stuck calling Helpdesk every 5min….


WHAT ARE YOU WORKING ON?
TRYING TO FIX THE PROBLEMS I CREATED WHEN I TRIED TO FIX THE PROBLEMS I CREATED WHEN I TRIED TO FIX THE PROBLEMS I CREATED WHEN...


NEVER HAVE I FELT SO CLOSE TO ANOTHER SOUL
AND YET SO HELPLESSLY ALONE
AS WHEN I GOOGLE AN ERROR
AND THERE'S ONE RESULT
A THREAD BY SOMEONE WITH THE SAME PROBLEM
AND NO ANSWER
LAST POSTED TO IN 2003
WHO WERE YOU, DENVERCODER9?
WHAT DID YOU SEE?!

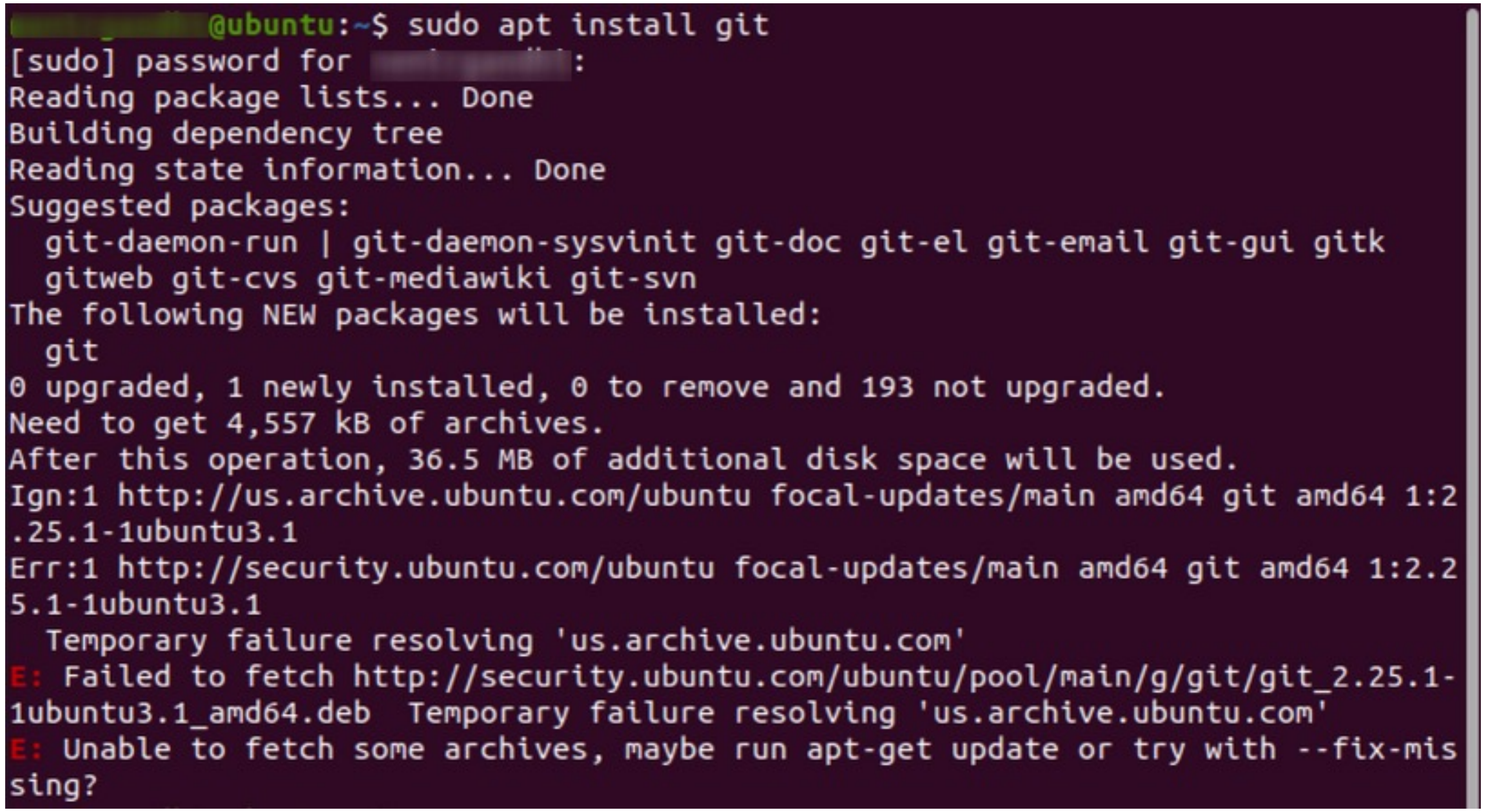
https://xkcd.com/979
https://xkcd.com/1739

Analyzing (and fixing) errors



```
[redacted]@ubuntu:~$ sudo apt install git
[sudo] password for [redacted]:
Reading package lists... Done
Building dependency tree
Reading state information... Done
Suggested packages:
  git-daemon-run | git-daemon-sysvinit git-doc git-el git-email git-gui gitk
  gitweb git-cvs git-mediawiki git-svn
The following NEW packages will be installed:
  git
0 upgraded, 1 newly installed, 0 to remove and 193 not upgraded.
Need to get 4,557 kB of archives.
After this operation, 36.5 MB of additional disk space will be used.
Ign:1 http://us.archive.ubuntu.com/ubuntu focal-updates/main amd64 git amd64 1:2
.25.1-1ubuntu3.1
Err:1 http://security.ubuntu.com/ubuntu focal-updates/main amd64 git amd64 1:2.2
5.1-1ubuntu3.1
  Temporary failure resolving 'us.archive.ubuntu.com'
E: Failed to fetch http://security.ubuntu.com/ubuntu/pool/main/g/git/git_2.25.1-
1ubuntu3.1_amd64.deb  Temporary failure resolving 'us.archive.ubuntu.com'
E: Unable to fetch some archives, maybe run apt-get update or try with --fix-mis
sing?
```

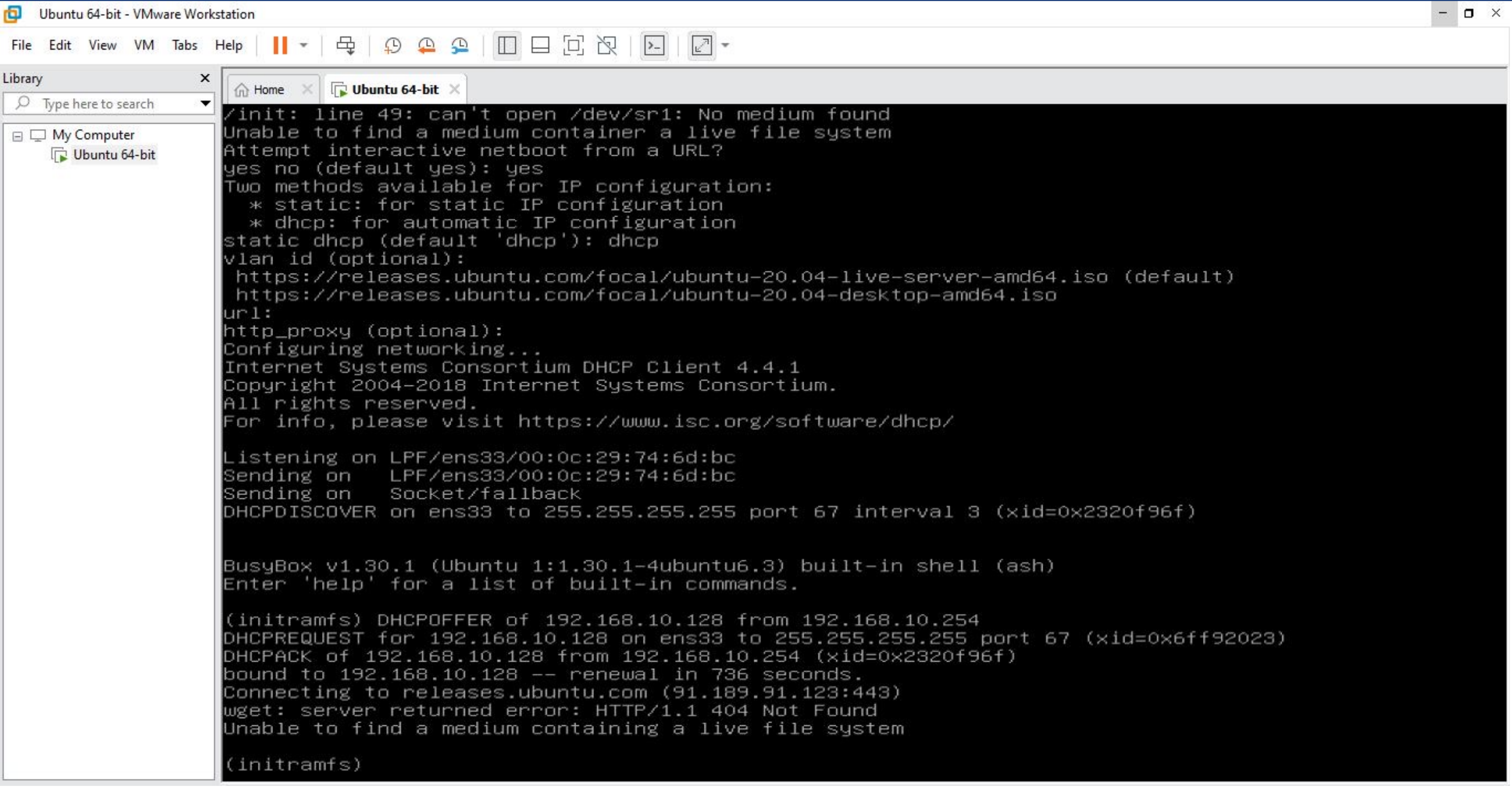
Analyzing (and fixing) errors



```
[redacted]@ubuntu:~$ sudo apt install git
[sudo] password for [redacted]:
Reading package lists... Done
Building dependency tree
Reading state information... Done
Suggested packages:
  git-daemon-run | git-daemon-sysvinit git-doc git-el git-email git-gui gitk
  gitweb git-cvs git-mediawiki git-svn
The following NEW packages will be installed:
  git
0 upgraded, 1 newly installed, 0 to remove and 193 not upgraded.
Need to get 4,557 kB of archives.
After this operation, 36.5 MB of additional disk space will be used.
Ign:1 http://us.archive.ubuntu.com/ubuntu focal-updates/main amd64 git amd64 1:2
.25.1-1ubuntu3.1
Err:1 http://security.ubuntu.com/ubuntu focal-updates/main amd64 git amd64 1:2.2
5.1-1ubuntu3.1
  Temporary failure resolving 'us.archive.ubuntu.com'
E: Failed to fetch http://security.ubuntu.com/ubuntu/pool/main/g/git/git_2.25.1-
1ubuntu3.1_amd64.deb  Temporary failure resolving 'us.archive.ubuntu.com'
E: Unable to fetch some archives, maybe run apt-get update or try with --fix-mis
sing?
```

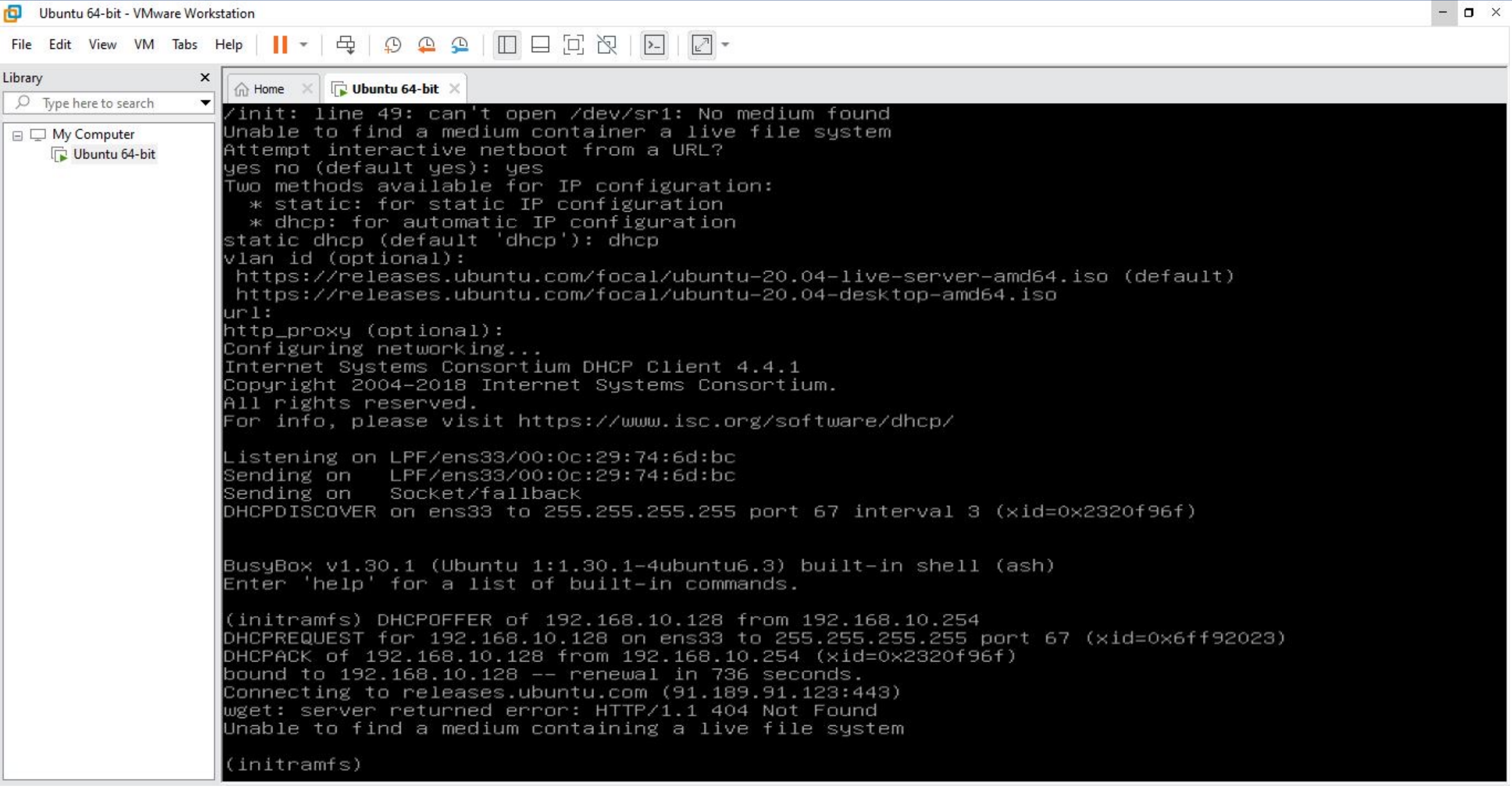
Network error, most probably not connected to internet...

Analyzing (and fixing) errors



```
/init: line 49: can't open /dev/sr1: No medium found
Unable to find a medium container a live file system
Attempt interactive netboot from a URL?
yes no (default yes): yes
Two methods available for IP configuration:
  * static: for static IP configuration
  * dhcp: for automatic IP configuration
static dhcp (default 'dhcp'): dhcp
vlan id (optional):
 https://releases.ubuntu.com/focal/ubuntu-20.04-live-server-amd64.iso (default)
 https://releases.ubuntu.com/focal/ubuntu-20.04-desktop-amd64.iso
url:
http_proxy (optional):
Configuring networking...
Internet Systems Consortium DHCP Client 4.4.1
Copyright 2004-2018 Internet Systems Consortium.
All rights reserved.
For info, please visit https://www.isc.org/software/dhcp/

Listening on LPF/ens33/00:0c:29:74:6d:bc
Sending on   LPF/ens33/00:0c:29:74:6d:bc
Sending on   Socket/fallback
DHCPDISCOVER on ens33 to 255.255.255.255 port 67 interval 3 (xid=0x2320f96f)


BusyBox v1.30.1 (Ubuntu 1:1.30.1-4ubuntu6.3) built-in shell (ash)
Enter 'help' for a list of built-in commands.

(initramfs) DHCPOFFER of 192.168.10.128 from 192.168.10.254
DHCPREQUEST for 192.168.10.128 on ens33 to 255.255.255.255 port 67 (xid=0x6ff92023)
DHCPACK of 192.168.10.128 from 192.168.10.254 (xid=0x2320f96f)
bound to 192.168.10.128 -- renewal in 736 seconds.
Connecting to releases.ubuntu.com (91.189.91.123:443)
wget: server returned error: HTTP/1.1 404 Not Found
Unable to find a medium containing a live file system

(initramfs)
```

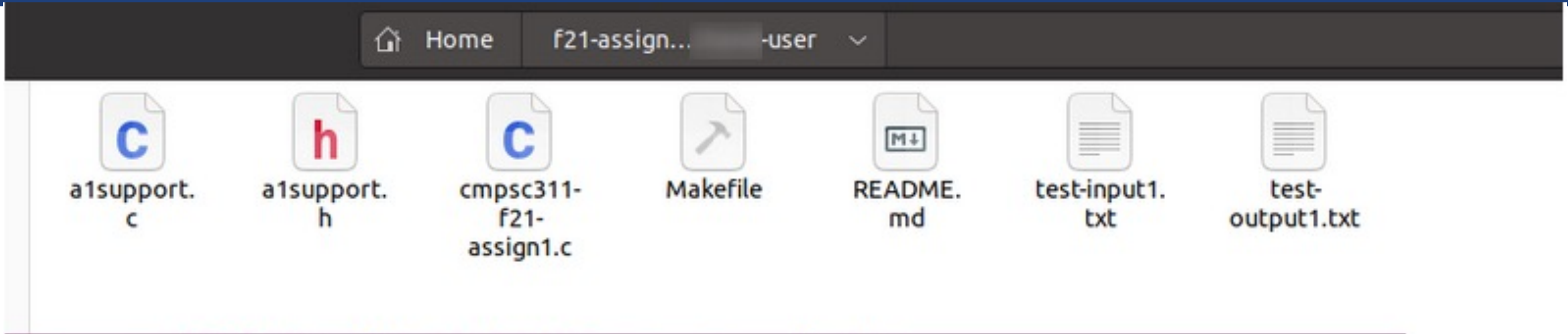
Analyzing (and fixing) errors

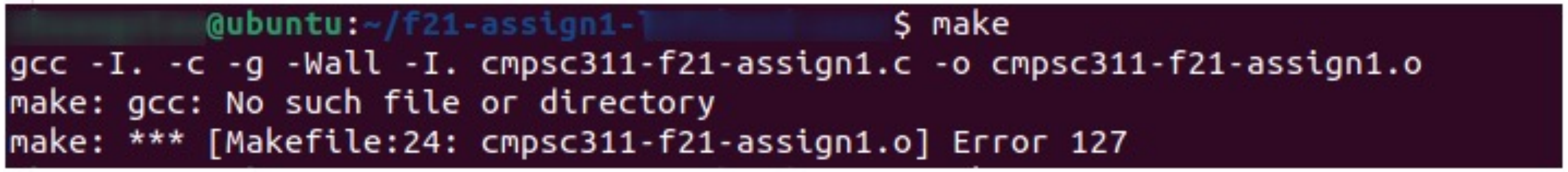


```
/init: line 49: can't open /dev/sr1: No medium found
Unable to find a medium container a live file system
Attempt interactive netboot from a URL?
yes no (default yes): yes
Two methods available for IP configuration:
  * static: for static IP configuration
  * dhcp: for automatic IP configuration
static dhcp (default 'dhcp'): dhcp
vlan id (optional):
 https://releases.ubuntu.com/focal/ubuntu-20.04-live-server-amd64.iso (default)
 https://releases.ubuntu.com/focal/ubuntu-20.04-desktop-amd64.iso
url:
http_proxy (optional):
Configuring networking...
Internet Systems Consortium DHCP Client 4.4.1
Copyright 2004-2018 Internet Systems Consortium.
All rights reserved.
For info, please visit https://www.isc.org/software/dhcp/

Listening on LPF/ens33/00:0c:29:74:6d:bc
Sending on   LPF/ens33/00:0c:29:74:6d:bc
Sending on   Socket/fallback
DHCPDISCOVER on ens33 to 255.255.255.255 port 67 interval 3 (xid=0x2320f96f)


BusyBox v1.30.1 (Ubuntu 1:1.30.1-4ubuntu6.3) built-in shell (ash)
Enter 'help' for a list of built-in commands.

(initramfs) DHCPOFFER of 192.168.10.128 from 192.168.10.254
DHCPREQUEST for 192.168.10.128 on ens33 to 255.255.255.255 port 67 (xid=0x6ff92023)
DHCPACK of 192.168.10.128 from 192.168.10.254 (xid=0x2320f96f)
bound to 192.168.10.128 -- renewal in 736 seconds.
Connecting to releases.ubuntu.com (91.189.91.123:443)
wget: server returned error: HTTP/1.1 404 Not Found
Unable to find a medium containing a live file system

(initramfs)
```

ISO file not specified (did not follow VM creation instructions carefully)

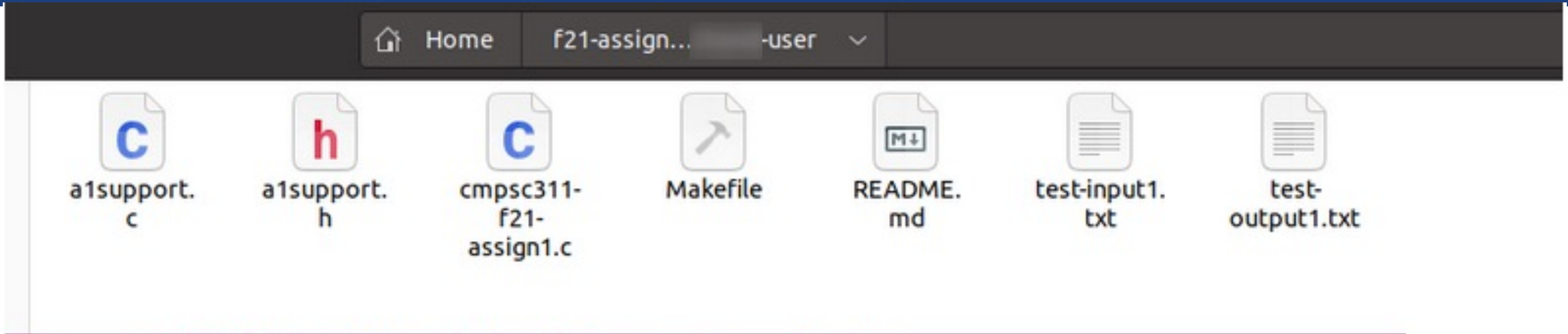
Analyzing (and fixing) errors

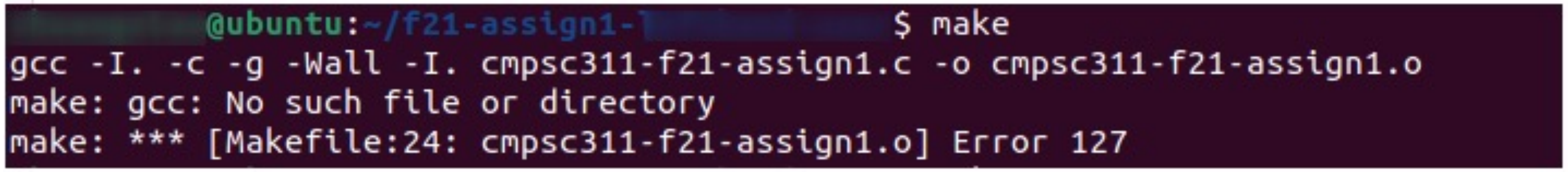




```
[redacted]@ubuntu:~/f21-assign1-[redacted]$ make
gcc -I. -c -g -Wall -I. cmpsc311-f21-assign1.c -o cmpsc311-f21-assign1.o
make: gcc: No such file or directory
make: *** [Makefile:24: cmpsc311-f21-assign1.o] Error 127
```

Analyzing (and fixing) errors



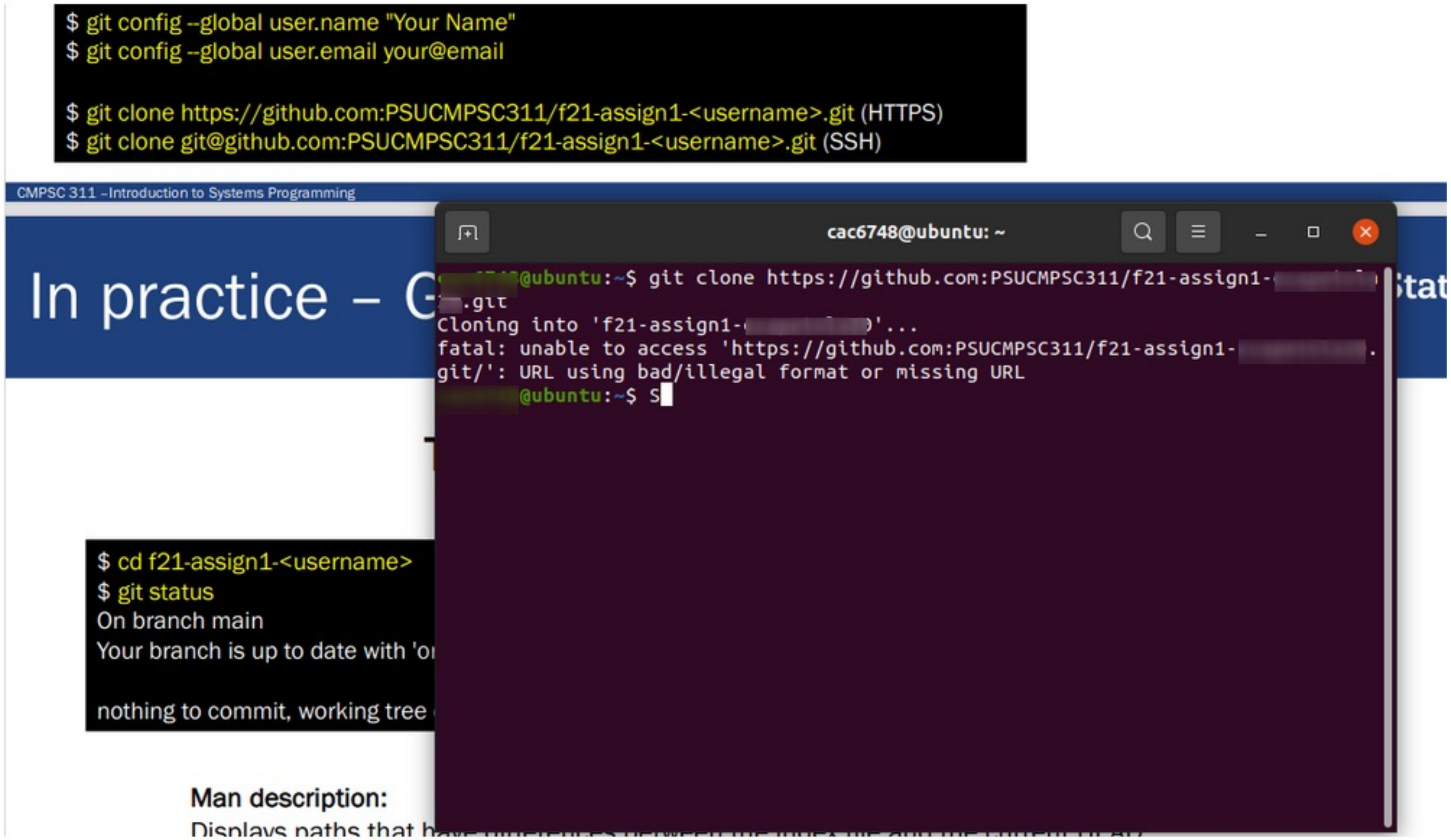


```
[redacted]@ubuntu:~/f21-assign1-[redacted] $ make
gcc -I. -c -g -Wall -I. cmpsc311-f21-assign1.c -o cmpsc311-f21-assign1.o
make: gcc: No such file or directory
make: *** [Makefile:24: cmpsc311-f21-assign1.o] Error 127
```

gcc not installed, did not run the prerequisites…

`sudo apt-get install build-essential`

Analyzing (and fixing) errors



```
$ git config --global user.name "Your Name"
$ git config --global user.email your@email

$ git clone https://github.com:PSUCMPSC311/f21-assign1-<username>.git (HTTPS)
$ git clone git@github.com:PSUCMPSC311/f21-assign1-<username>.git (SSH)
```

CMPSC 311 – Introduction to Systems Programming

In practice – G

```
$ cd f21-assign1-<username>
$ git status
On branch main
Your branch is up to date with 'or

nothing to commit, working tree
```

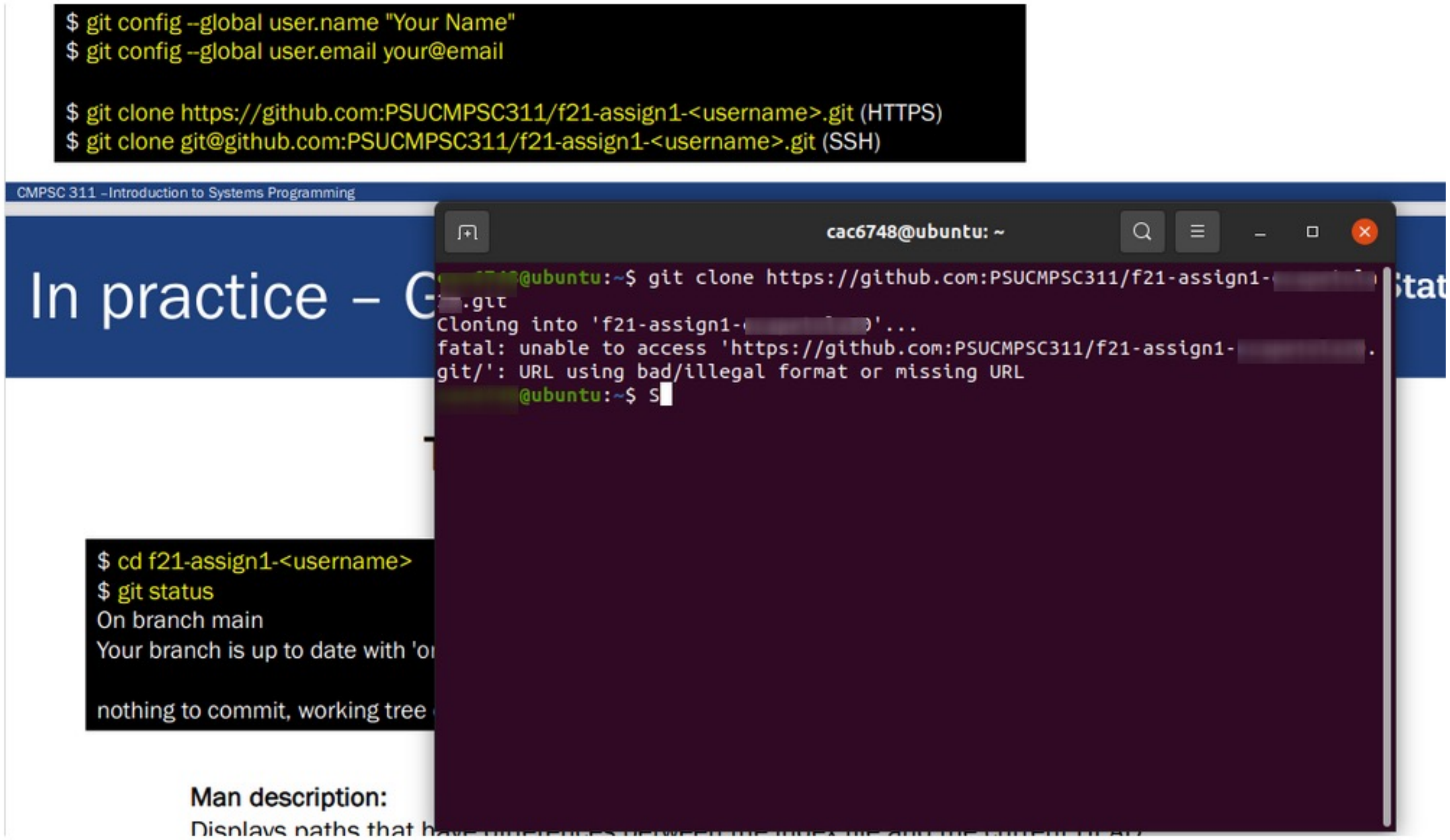
Man description:

Displays paths that h

cac6748@ubuntu: ~

```
[redacted]@ubuntu:~$ git clone https://github.com:PSUCMPSC311/f21-assign1-[redacted]
[redacted].git
Cloning into 'f21-assign1-[redacted]'...
fatal: unable to access 'https://github.com:PSUCMPSC311/f21-assign1-[redacted].
git/': URL using bad/illegal format or missing URL
[redacted]@ubuntu:~$ S
```

Analyzing (and fixing) errors



```
$ git config --global user.name "Your Name"
$ git config --global user.email your@email

$ git clone https://github.com:PSUCMPSC311/f21-assign1-<username>.git (HTTPS)
$ git clone git@github.com:PSUCMPSC311/f21-assign1-<username>.git (SSH)
```

CMPSC 311 – Introduction to Systems Programming

In practice – G

Stat

T

```
$ cd f21-assign1-<username>
$ git status
On branch main
Your branch is up to date with 'or

nothing to commit, working tree
```

Man description:

Displays paths that h

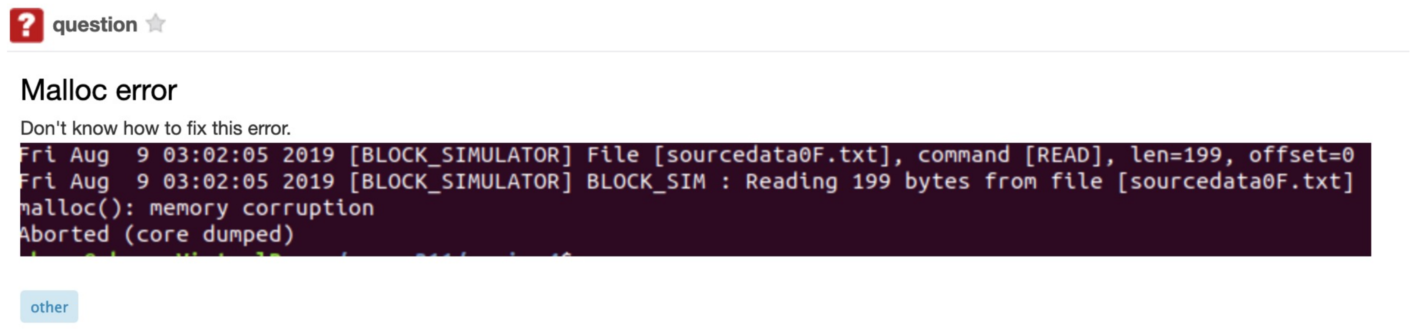
cac6748@ubuntu: ~

```
[redacted]@ubuntu:~$ git clone https://github.com:PSUCMPSC311/f21-assign1-[redacted]
[redacted].git
Cloning into 'f21-assign1-[redacted]'...
fatal: unable to access 'https://github.com:PSUCMPSC311/f21-assign1-[redacted].
git/': URL using bad/illegal format or missing URL
[redacted]@ubuntu:~$ S
```

"/" at the end of the URL...

Debugging

- Often the most complicated and time-consuming part of developing a program is *debugging*.
  - Figuring out where your program diverges from your idea of what the code should be doing.
  - Confirm that your program is doing what you expect to be doing.
  - Finding and fixing bugs …


question

Malloc error

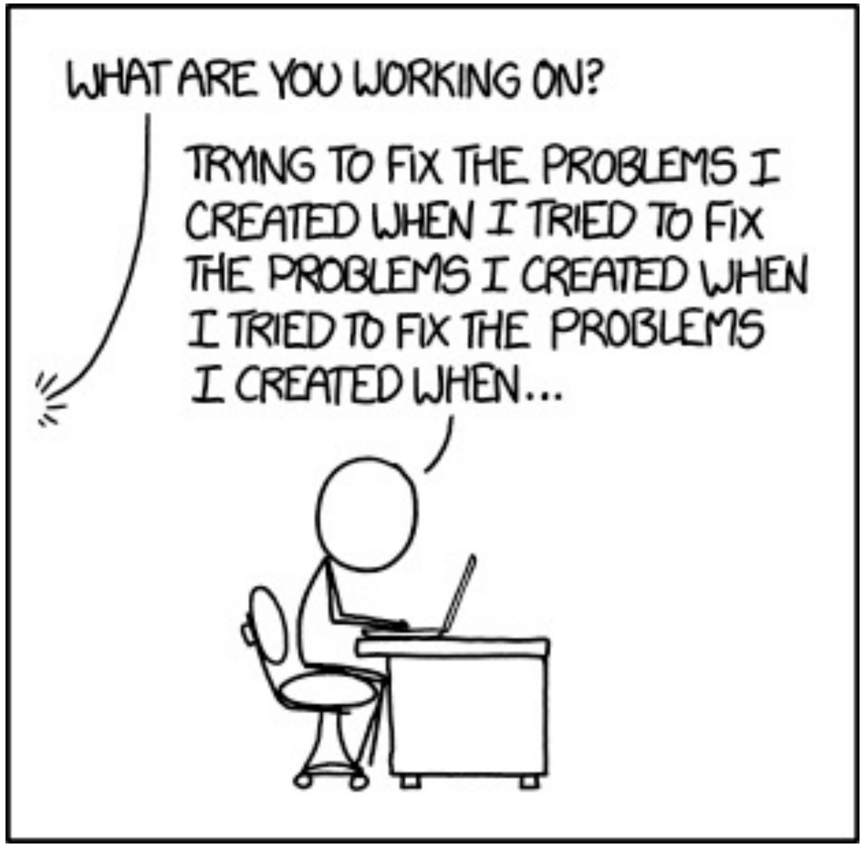
Don't know how to fix this error.

```
Fri Aug  9 03:02:05 2019 [BLOCK_SIMULATOR] File [sourcedata0F.txt], command [READ], len=199, offset=0
Fri Aug  9 03:02:05 2019 [BLOCK_SIMULATOR] BLOCK_SIM : Reading 199 bytes from file [sourcedata0F.txt]
malloc(): memory corruption
Aborted (core dumped)
```

other

Common debugging patterns

1. Identifying the cause of a program crash
2. Identifying the source of incorrect program output
3. Identifying an infinite loop or recursion


WHAT ARE YOU WORKING ON?
TRYING TO FIX THE PROBLEMS I CREATED WHEN I TRIED TO FIX THE PROBLEMS I CREATED WHEN I TRIED TO FIX THE PROBLEMS I CREATED WHEN...

Printing/Logging

- One way to debug is to print out the values of variables and memory at different points
    - e.g., `printf( “My variable value is %d”, myvar );`
- Logging (such as LogMessage()) provides more sophisticated interfaces to simple prints, log to file
    - Turning on an off “debug levels”
        - LOG_INFO_LEVEL
        - LOG_WARNING_LEVEL
        - LOG_ERROR_LEVEL
        - LOG_OUTPUT_LEVEL

```
#include <cmpsc311_log.h>
...
enableLogLevels( LOG_INFO_LEVEL );
...
logMessage( LOG_OUTPUT_LEVEL,  “The log message is %d”, value );
...
Fri Oct 18 10:26:04 2013 [OUTPUT] The log message is 11
```

gdb

- You run the debugger by passing the program to gdb

```
$ gdb [program name]
```

- This is an interactive terminal-based debugger
- Invoking the debugger does not start the program, but simply drops you into the gdb environment.

```
$ gdb debugging
GNU gdb (GDB) 7.5.91.20130417-cvs-ubuntu
Copyright (C) 2013 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".
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>...
Reading symbols from /home/mcdaniel/src/debugging/debugging...done.
(gdb)
```

Running the program

- Once you enter the program, you must start the program running, using the `run` command

```
(gdb) run
Starting program: /root/project/debugging
Factorial of 5: 120
 [Inferior 1 (process 149) exited normally]
(gdb)
```

- If you have arguments to pass to the program, simply add them to the `run` command line

```
(gdb) run 12
Starting program: /root/project/debugging 12
Factorial of 12: 479001600
 [Inferior 1 (process 153) exited normally]
(gdb)
```

Looking at code

- If want to look at regions of code, so use the `list` command
  - shows 10 lines at a time, centered around the target
  - you can specify a line number (in the current file),
  - or specify a function name

```
(gdb) list 4
1         #include <assert.h>
2         #include <stdio.h>
3         #include <stdlib.h>
4
5         int factorial(int i)
6         {
7             assert(i >= 0); // ** CHECK **
8             if (i <= 1) {
9                 return (i);
10            }
(gdb)
```

```
(gdb) l main
10                }
11                return (factorial(i - 1) * i);
12            }
13
14            int main(int argc, char** argv)
15            {
16                int i;
17                if (argc > 1) {
18                    i = atoi(argv[1]);
19                }
(gdb)
```

- Most commands are aliased with single character (l)

Breakpoints

- A breakpoint is a position in the code you wish for the debugger to stop and wait for your commands

```
break [function_name | line_number]
```

  - Breakpoints are set using the break (b) command
  - Each one is assigned a number you can reference later
- You can delete the breakpoint by using the delete (d) command

```
delete [breakpoint_number]
```

```
(gdb) b factorial
Breakpoint 1 at 0x400587: file debugging.c, line 6.
(gdb) b 16
Breakpoint 2 at 0x4005db: file debugging.c, line 16.
(gdb) delete 1
(gdb) d 2
```

Conditional Breakpoints

- A conditional breakpoint is a point where you want the debugger only if the condition holds
  - Breakpoints are set using the `cond` command

```
cond [breakpoint_number] (expr)
```

```
(gdb) l 7
7               assert(i >= 0); // ** CHECK **
(gdb) b 7
Breakpoint 1 at 0x6e5: file debugging.c, line 7.
(gdb) cond 1 i<=1
(gdb) r
Starting program: /root/project/debugging

Breakpoint 1, factorial (i=1) at debugging.c:7
7               assert(i >= 0); // ** CHECK **
(gdb) c
Continuing.
Factorial of 5: 120
 [Inferior 1 (process 157) exited normally]
(gdb)
```

Seeing breakpoints

- If you want to see your breakpoints use the *info breakpoints* command

```
(gdb) info breakpoints
Num     Type           Disp Enb Address            What
1       breakpoint     keep y   0x00000000000006e5 in factorial at debugging.c:7
2       breakpoint     keep y   0x000000000000075c in main at debugging.c:22
(gdb)
```

- The info command allows you see lots of information about the state of your environment and program

```
(gdb) help info
Generic command for showing things about the program being debugged.

List of info subcommands:

info address -- Describe where symbol SYM is stored
info all-registers -- List of all registers and their contents
info args -- Argument variables of current stack frame
...
```

Examining the stack

- You can always tell where you are in the program by using the where command, which gives you a stack and the specific line number you are one

```
(gdb) b 7 if i<=1
Breakpoint 1 at 0x6e5: file debugging.c, line 7.
(gdb) run 4
Starting program: /root/project/debugging 4

Breakpoint 1, factorial (i=1) at debugging.c:7
7              assert(i >= 0); // ** CHECK **
(gdb) where
#0  factorial (i=1) at debugging.c:7
#1  0x0000555555554722 in factorial (i=2) at debugging.c:11
#2  0x0000555555554722 in factorial (i=3) at debugging.c:11
#3  0x0000555555554722 in factorial (i=4) at debugging.c:11
#4  0x0000555555554764 in main (argc=2, argv=0x7fffffffe718) at debugging.c:20
(gdb)
```

Examining memory

- You examine memory regions using the x command

```
x [/<num><format><size>] address
```

- Modify the output using a number of values formatted with `[oxdutfais]` type and size are `b`(byte), `h`(halfword), `w`(word), `g`(giant, 8 bytes).

```
(gdb) x buf
0x555555756260:      0xefefefef
(gdb) x/8xb buf
0x555555756260:      0xef      0xef      0xef      0xef      0xef      0xef      0xef      0xef
(gdb) x/xg buf
0x555555756260:      0xefefefefefefefef
(gdb) x buf
0x555555756260:      0xefefefefefefefef
(gdb) x &buf
0x7fffffffe5f8:      0x0000555555756260
(gdb)
```

```
int myexamine() {
    char *buf = NULL;
    buf = malloc( 8 );
    memset( buf, 0xef, 8 );
    return( 0 ); // breakpoint here
}
```

Printing variables

- At any point in the debug session can print the value of any variable you want by printing its value using

```
print[/<format>] variable
```

- Dictate the output formatted with `o`(octal), `x`(hex), `d`(decimal), `u`(unsigned decimal), `t`(binary), `f`(float), `a`(address), `i`(instruction), and `s`(string)

```
(gdb) p values
$1 = "\001\002\003\004"
(gdb) p/x values
$2 = {0x1, 0x2, 0x3, 0x4}
(gdb) p val1
$3 = 4283787007
(gdb) p/x val1
$4 = 0xff5566ff
(gdb) p val2
$5 = 2.45677996
(gdb)
```

```c
int myvalues() {
    char values[] = { 0x1, 0x2, 0x3, 0x4 };
    uint32_t val1 = 0xff5566ff;
    float val2 = 2.45678;
    return( 0 ); // breakpoint here
}
```

Practice Exercise – Week #2

1. Debugging C and program repair
2. (Re-)implement assign#1 from 311

- Clone GitHub Classroom repo: https://classroom.github.com/a/Gy_DvcGF

Part 1 – Debugging C & Program Repair

- Fix issues in the `.c` files
- Hints:
  - Format code
  - Locate bugs (syntax, type errors, typos, implementation errors, logical errors)
  - Fix them
  - Etc.

Part 2 – Assignment #1 from CMPSC311

- (Re-)Implement assignment#1 from CMPSC311 without looking at your code.