1. All code follows the Linux Kernel Coding Style

2. Final filename follows established standard.
```
    enscript --header='$n %E %*|$%|Spencer Goulette' ECE331-FN-Goulette-Spencer.txt -o - |
ps2pdf - ECE331-FN-Goulette-Spencer.pdf
```

3.
```
# Makefile for Final

TARGET=forme
OBJS=fileone.o filetwo.o filethree.o
CFLAGS=-Wall -g
LIBS=-lm

all: $(TARGET)
$(TARGET): $(OBJS)
    $(CC) -o $(TARGET) $(OBJS) $(LIBS)

clean:
    rm -f $(TARGET) $(OBJS) core*
```

4.
```
#!/usr/bin/perl

# Prints latitude, longitude, and elevation from any and all $GPGGA NMEA sentence.
while(<STDIN>) {
    chomp($_);
    @columns = split /,/, $_;
    if($columns[0] eq "\$GPGGA") {
        print $columns[2] . $columns[3] . " " . $columns[4] . $columns[5] . " " . $columns[
9] . "\n";
    }
}
```

5.

Network(DDN)	IP (DDN)	Netmask(CIDR)	Broadcast (DDN)
80.68.0.0	80.68.105.5	/17	80.68.127.255
101.210.214.64	101.210.214.64	/26	101.210.214.127

```
    IP ADDR: 080.068.105.005
    NETMASK: 255.255.128.000
    bitwise and
    ------------------------
             080.068.000.000 -> network address 80.68.0.0
                                80.68.0.0/17 (the number of 1 bits)

    IP ADDR: 080.068.105.005
   ~NETMASK: 255.255.128.000
    bitwise or
    ------------------------
             080.068.127.255 -> broadcast address 80.68.127.255


    BRDCAST: 101.210.214.127
   ~NETWORK: 101.210.214.64
    bitwise xor
    ------------------------
             255.255.255.192 -> Netmask 255.255.255.192


    IP ADDR: ???.???.???.???
   ~NETMASK: 255.255.255.192
```

bitwise or

101.210.214.127 -> Possible IP ADDR: 101.210.214.64
From this, I know that the IP ADDR must be less than 128 in the last byte since
the bit for 128 is a 0 in the broadcast address

IP ADDR: ???.???.???.???
NETMASK: 255.255.255.192
bitwise and

101.210.214.64 -> Possible IP ADDR Works: 101.210.214.64
From this, I know that the IP ADDR must be at least 64 or greater in the last byte since
the bit for 64 must be a 1 for both the IP and Netmask

From further investigation, the IP addresses can be 101.210.214.64 to 101.210.214.127

Last byte of Netmask must be 11000000
Last byte of the IP must be 01XXXXXX

6.

a. sqlite3 logger.db
CREATE TABLE datalogger(Latitude TEXT, Longitude TEXT, Elevation REAL);

b. INSERT INTO datalogger(Latitude, Longitude, Elevation) VALUES('4439.3381N','06744.4518W',5.6), ('4439.3381N','06744.4518W',5.6), ('4439.3381N','06744.4518W',5.6);

c. SELECT * FROM datalogger ORDER BY rowid DESC LIMIT 200;

7.

```
#!/usr/bin/python

import sys

# Finds number of lines and characters
try:
    for i in range(1,len(sys.argv[1:])+1):
        f = open(sys.argv[i],'r')
        lines = 0
        characters = 0
        for l in f.readlines():
            lines = lines + 1
            characters = characters + len(l)
        f.close()
        print "{}: Lines - {}, Characters - {}\n".format(sys.argv[i],lines,characters)

except:
    print "Unable to open files\n"
```

8.

a. No it does not execute properly under all conditions

b. First of all, there needs to be one more) on the gpiod_set_value line. It seems like there is a possible race condition since there is uncontrolled access to shared data (the data buffer). It seems that if processes are run concurrently, data could become corrupt or incorrect. Another condition that won't work is if the data passed is greater than 4096 uint8_t. Probably should kalloc enough data based off of the count passed instead of presetting it to 4096 uint8_t.

c. No it does not successfully protect shared resources

d. It does not successfully protect shared resources because if two processes were running concurrently, they both would copy_from_user into the same data buffer. Then when the f

or loops are run by a single process at a time, the data used will probably not be the corr
ect data.

e. To correct the code I would put the mutex_lock_interruptible if statement before the copy_from_user if statement, so that copy_from_user is "locked". This way, the shared data is controlled and only one process at a time stores data from the user and sets the gpio v
alue.

9. ^(1[0-2]|[1-9]):([0-5][0-9]) ?([AP]M)$

10.

```
#include <stdio.h>
#include <stdlib.h>
#include <stdint.h>
#include <sys/types.h>
#include <sys/stat.h>

void fileinfo(int size, char *files[]);

int main(int argc, char *argv[])
{
    fileinfo(argc, argv);

    return 0;
}

// Gets total size for a set of files
void fileinfo(int size, char *files[])
{
    struct stat sfile;
    int bytes = 0;
    for(int i = 0; i < size - 1; i++) {
        // Error checking
        if (stat(files[i+1],&sfile) == -1) {
            printf("Error reading file info!\n");
            printf("Usage: ./fileinfo (Filenames)\n");
            return;
        }
        bytes += sfile.st_size;
    }
    printf("Size: %d\n",bytes);

    return;
}
```

11.

a. sudo chown pi:pi -R /usr/src/

b. sudo ln -s /var/lib/systimer/logs/abc /usr/arm/opt/bin/foobar

c. sudo chmod -R og+rx-w /opt/ngspice/

d. egrep "^[0-9]+$" [0-9][0-9]

12. After all of this, I would contact my supervisor and let them know of what I saw and th
e files that I found. By not investigating these files and letting my supervisor take care
of the matter, I'm less likely to get in trouble. Also, it no longer becomes my issue.

13.

```
45 */12 1 * 1,5 toe_nail_clipping
 |  |   | |  |
 |  |   | |  ----- On Monday and Friday
 |  |   | --------- Any month
 |  |   ----------- 1st day of the month
 |  --------------- When hour is divisible by 12, so 12am and 12pm
  ------------------ Has to be the 45th minute of the hour
```

So the cron job runs toe_nail_clipping at 12:45am and 12:45pm on the first day of the m

onth and on Mondays and Fridays

```
    Random examples of when it runs: 12:45am on May 22nd, 2020      - Friday
                                     12:45pm on May 25th, 2020      - Monday
                                     12:45pm on July 1st, 2020      - 1st day of the month
```

14. scp -P 666 simulation wizard@summit.ornl.gov

15. thd, triggerhappy global hotkey daemon, watches all configured input devices for key, s
witch or button events and can launch arbitrary commands specified by the administrator.

    The command used to find this out: man thd