Report for CSC3150 Assignment1

Name: Hu Wenhan

Student id: 120090565

Task 1

1. How to design:

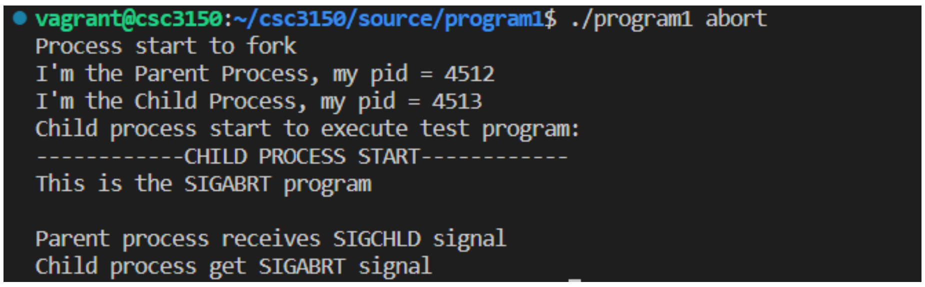
This task needed us to fork a child process in user mode to execute 15 test programs respectively. In my program1.c file, there is a main() function which was used to complete all operations. Firstly, the program forked a child process by fork() function. Then the child process executed the test program by execve() function. And during this period, the parent process was waiting for the termination of the child process by waitpid() function. Also the parent process got the SIGCHLD signal by waitpid() function. And then the program printed out the termination information of the child proess by the signal.

2. How to set up the development environment:

In task 1, the only environment needed was the VM setting which I learned from tutorial 1.

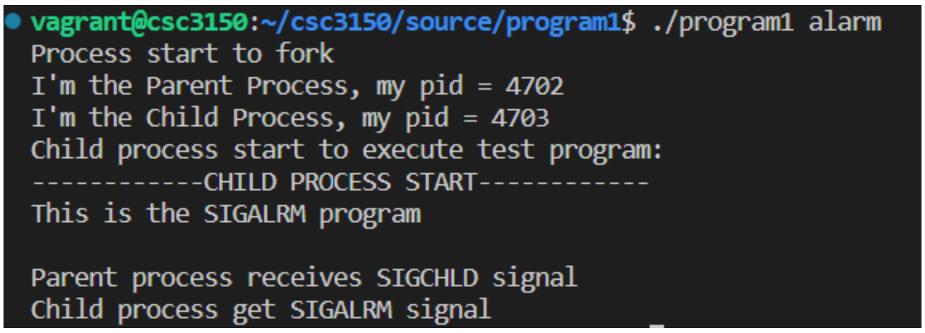
3. Screenshots of output:

a) Abort:

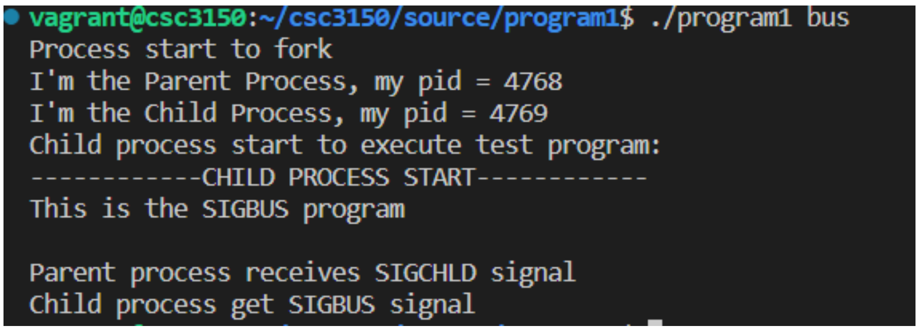


```
vagrant@csc3150:~/csc3150/source/program1$ ./program1 abort
Process start to fork
I'm the Parent Process, my pid = 4512
I'm the Child Process, my pid = 4513
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGABRT program

Parent process receives SIGCHLD signal
Child process get SIGABRT signal
```

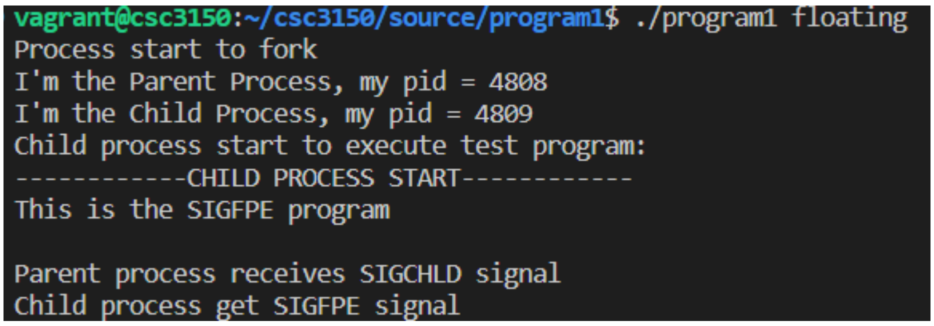
b) Alarm:


```
vagrant@csc3150:~/csc3150/source/program1$ ./program1 alarm
Process start to fork
I'm the Parent Process, my pid = 4702
I'm the Child Process, my pid = 4703
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGALRM program

Parent process receives SIGCHLD signal
Child process get SIGALRM signal
```

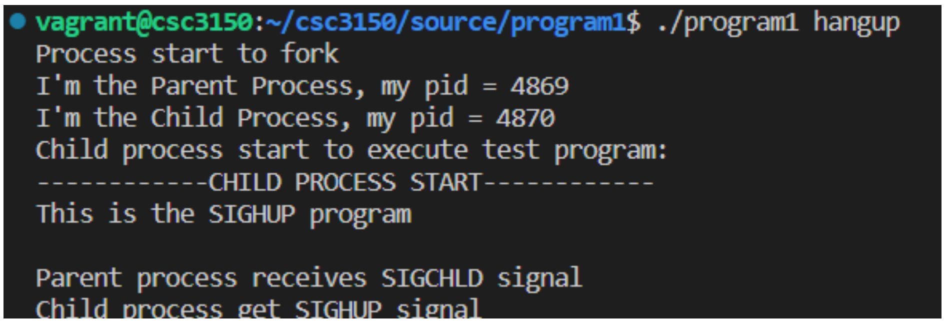
c) Bus:


```
vagrant@csc3150:~/csc3150/source/program1$ ./program1 bus
Process start to fork
I'm the Parent Process, my pid = 4768
I'm the Child Process, my pid = 4769
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGBUS program

Parent process receives SIGCHLD signal
Child process get SIGBUS signal
```

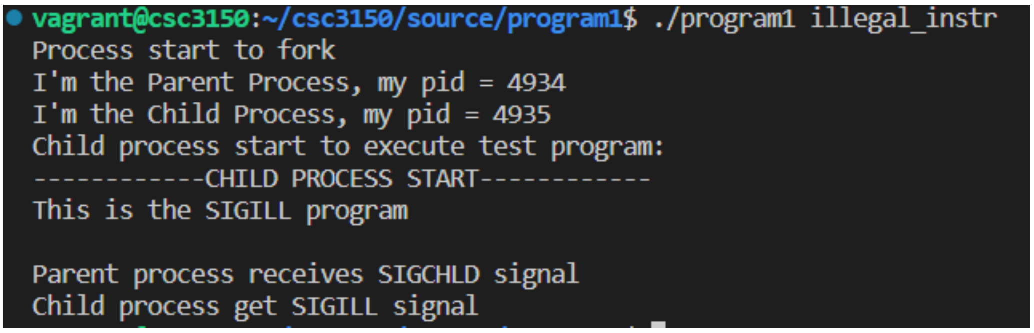
d) Floating:


```
vagrant@csc3150:~/csc3150/source/program1$ ./program1 floating
Process start to fork
I'm the Parent Process, my pid = 4808
I'm the Child Process, my pid = 4809
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGFPE program

Parent process receives SIGCHLD signal
Child process get SIGFPE signal
```

e) Hangup:

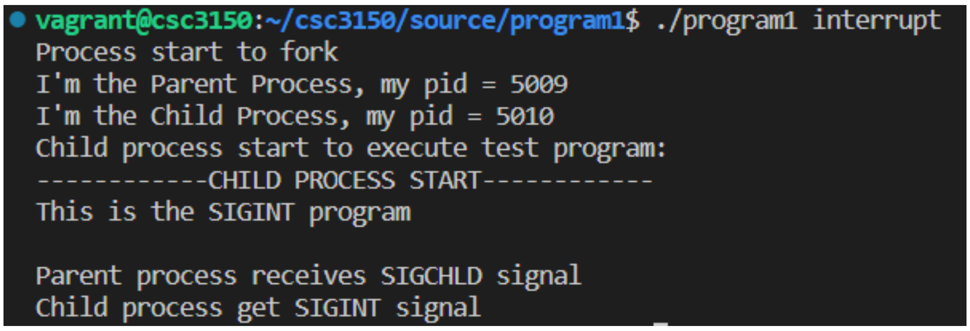

```
vagrant@csc3150:~/csc3150/source/program1$ ./program1 hangup
Process start to fork
I'm the Parent Process, my pid = 4869
I'm the Child Process, my pid = 4870
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGHUP program

Parent process receives SIGCHLD signal
Child process get SIGHUP signal
```

f) Illegal_instr:



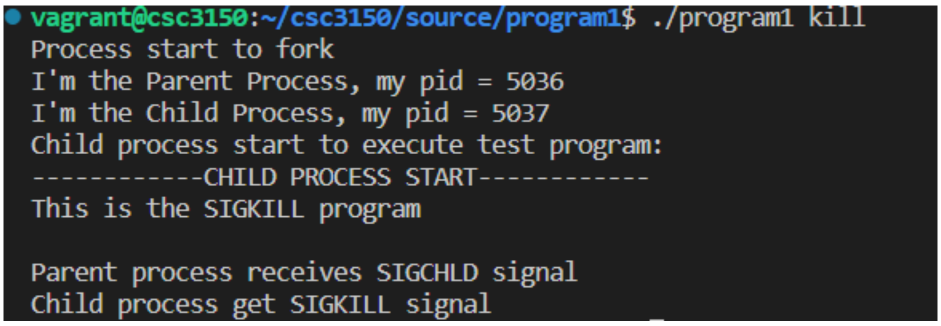
```
vagrant@csc3150:~/csc3150/source/program1$ ./program1 illegal_instr
Process start to fork
I'm the Parent Process, my pid = 4934
I'm the Child Process, my pid = 4935
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGILL program

Parent process receives SIGCHLD signal
Child process get SIGILL signal
```

g) Interrupt:



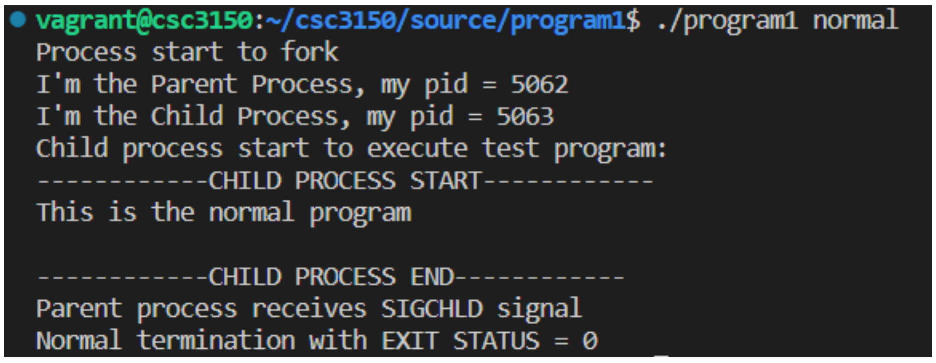
```
vagrant@csc3150:~/csc3150/source/program1$ ./program1 interrupt
Process start to fork
I'm the Parent Process, my pid = 5009
I'm the Child Process, my pid = 5010
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGINT program

Parent process receives SIGCHLD signal
Child process get SIGINT signal
```

h) Kill:



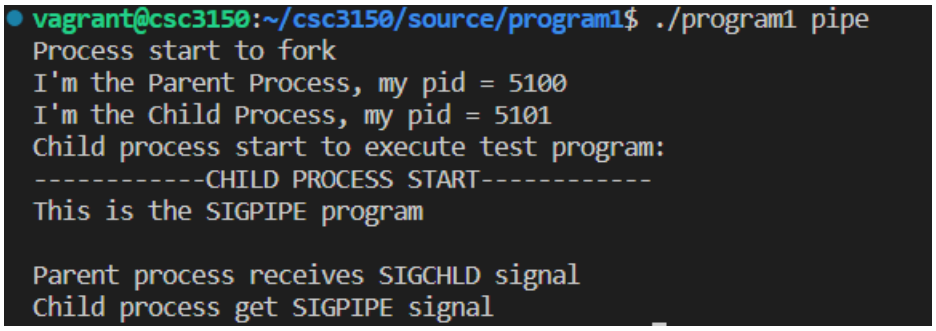
```
vagrant@csc3150:~/csc3150/source/program1$ ./program1 kill
Process start to fork
I'm the Parent Process, my pid = 5036
I'm the Child Process, my pid = 5037
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGKILL program

Parent process receives SIGCHLD signal
Child process get SIGKILL signal
```

i) Normal:



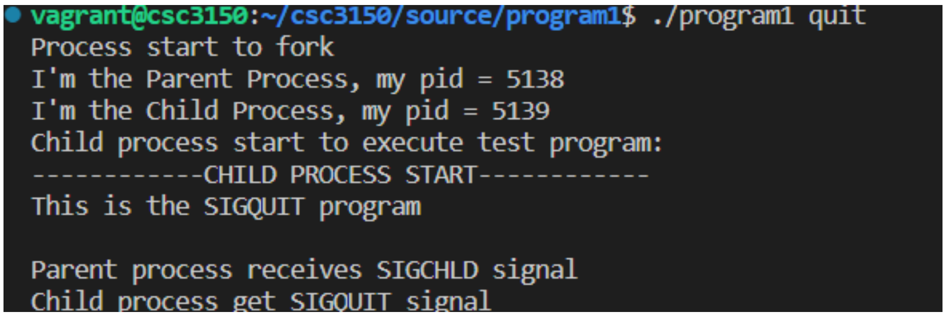
```
vagrant@csc3150:~/csc3150/source/program1$ ./program1 normal
Process start to fork
I'm the Parent Process, my pid = 5062
I'm the Child Process, my pid = 5063
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the normal program

------------CHILD PROCESS END------------
Parent process receives SIGCHLD signal
Normal termination with EXIT STATUS = 0
```

j) Pipe:



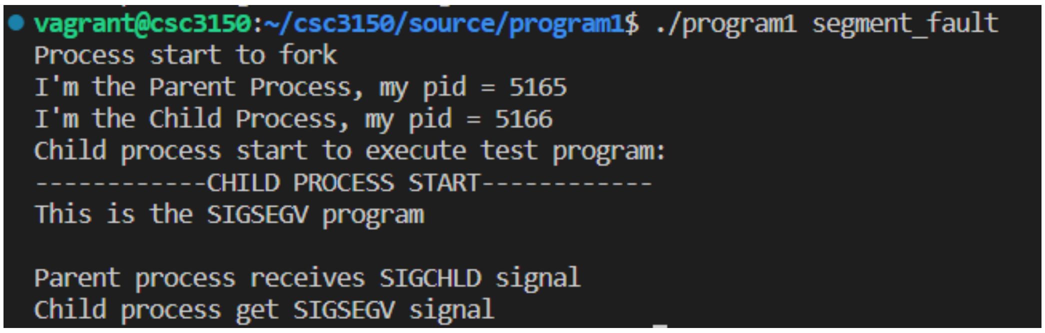
```
vagrant@csc3150:~/csc3150/source/program1$ ./program1 pipe
Process start to fork
I'm the Parent Process, my pid = 5100
I'm the Child Process, my pid = 5101
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGPIPE program

Parent process receives SIGCHLD signal
Child process get SIGPIPE signal
```

k) Quit:



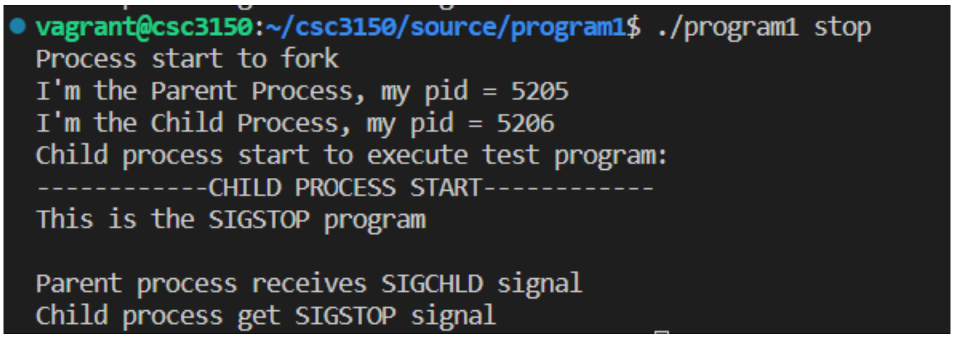
```
vagrant@csc3150:~/csc3150/source/program1$ ./program1 quit
Process start to fork
I'm the Parent Process, my pid = 5138
I'm the Child Process, my pid = 5139
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGQUIT program

Parent process receives SIGCHLD signal
Child process get SIGQUIT signal
```

l) Segment_fault:



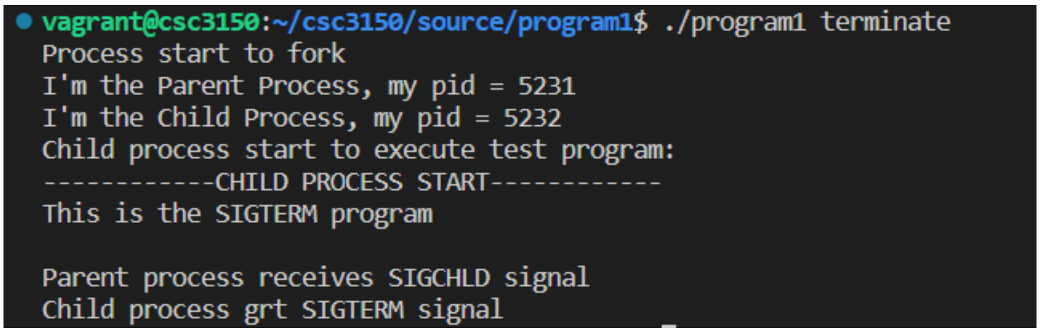
```
vagrant@csc3150:~/csc3150/source/program1$ ./program1 segment_fault
Process start to fork
I'm the Parent Process, my pid = 5165
I'm the Child Process, my pid = 5166
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGSEGV program

Parent process receives SIGCHLD signal
Child process get SIGSEGV signal
```

m) Stop:



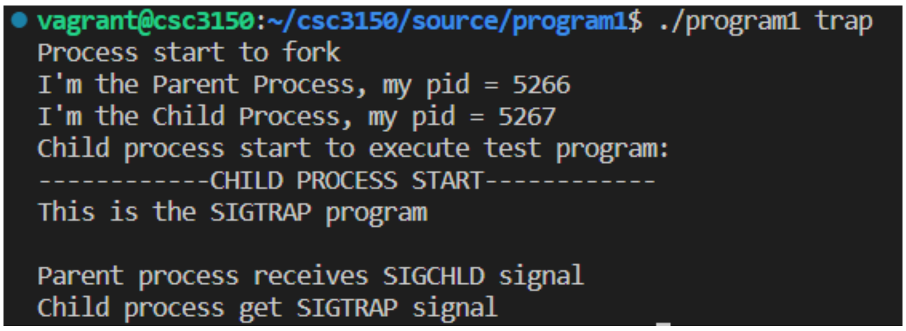
```
vagrant@csc3150:~/csc3150/source/program1$ ./program1 stop
Process start to fork
I'm the Parent Process, my pid = 5205
I'm the Child Process, my pid = 5206
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGSTOP program

Parent process receives SIGCHLD signal
Child process get SIGSTOP signal
```

n) Terminate:



```
vagrant@csc3150:~/csc3150/source/program1$ ./program1 terminate
Process start to fork
I'm the Parent Process, my pid = 5231
I'm the Child Process, my pid = 5232
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGTERM program

Parent process receives SIGCHLD signal
Child process grt SIGTERM signal
```

o) Trap:



```
vagrant@csc3150:~/csc3150/source/program1$ ./program1 trap
Process start to fork
I'm the Parent Process, my pid = 5266
I'm the Child Process, my pid = 5267
Child process start to execute test program:
------------CHILD PROCESS START------------
This is the SIGTRAP program

Parent process receives SIGCHLD signal
Child process get SIGTRAP signal
```

4. *What did I learn from this task:*

I think that I have learnt some functions and their roles in this task, such as fork(), execve(), and so on. Especially, my biggest gain is the waitpid() function. Firstly, I used the wait() function in the parent process to wait for the termination of the child process. However, when the stop.c test program raise SIGSTOP signal, the child process stopped, and then the whole program could not terminate. Then, I spent a lot of time to debug. Finally, I found that I should use waitpid() and add WUNTRACED so that the program can get the SIGSTOP signal raised by the child process.

Task 2

1. *How to design:*

a) Summary: The purpose of task2 is similar to task1, but the whole program needed to be complete in kernel mode. Firstly, my program called module_int() function to start the whole program. And within this function, the program called my_fork() function by kthread_create() function and wake_up_process() function called. Then in my_fork() function, firstly the program called kernel_clone() function to fork a child process, and call the my_exec() function which made the child process execute the test program. Also, the process id for both the parent and child process were printed out. Then still in my_fork() function, the program called my_wait() function so that the parent process could wait until the child process terminated. The signal received from the child process and related massage were printed out in kernel log also by my_wait() function. Finally, the program called module_exit() function to finish the whole program.

b) My_exec() function: Firstly the program called getname_kernel() function to reach the test program. Then do_execve() function was called to execute the test program.

c) My_wait() function: Firstly the program called do_wait() function to wait the termination of the child process. Then the program printed out the signal and related information by my_WTERMIG(), my_WIFEXITED(), my_WIFSIGNALED() and my_WIFSTOPPED() functions.

d) Also, there are four non-static functions in my program: do_wait, do_execve, kernel_clone, getname_kernel. In order to use them, firstly I used EXPORT_SYMBOL() function in linux-5.10.146 file, and also I type "extern" before them to clarify in my program.

2. *How to set up the development environment:*

a) Firstly, I downloaded kernel source code (linux-5.10.146) on Internet.

b) Then, I installed Dependency and development tools.

c) Then I decompressed the source code within the program2 folder.

d) After that, I add EXPORT_SYMBOL() into four c files in the linux-5.10.146 folder, and during this process I used chmod 777 to change the c files.

e) And then, I compiled kernel by make mrproper, make clean, make menuconfig, save the config and exit, make bzImage, make modules, make modules_install, make install, reboot.

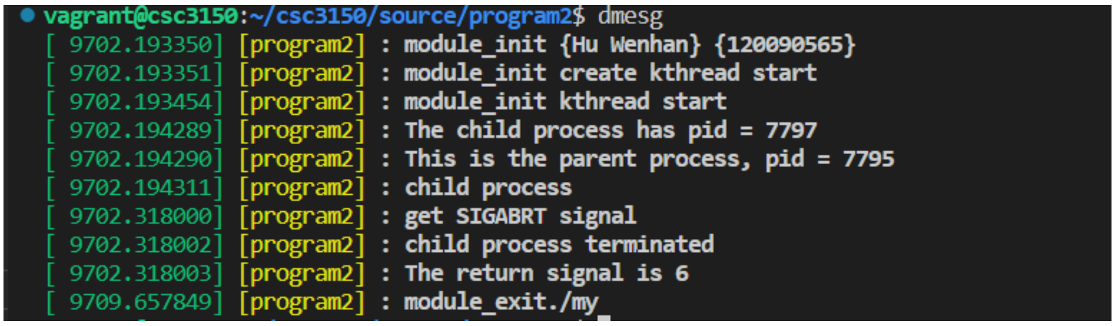
f) Then, I went back to the program2 folder, and type "make" to get ko file.

g) Then I inserted and removed kernel module by insmod program2.ko, rmmod program2.ko, dmesg.

h) And each time I inserted and removed kernel module, I use dmesg -c to clear.

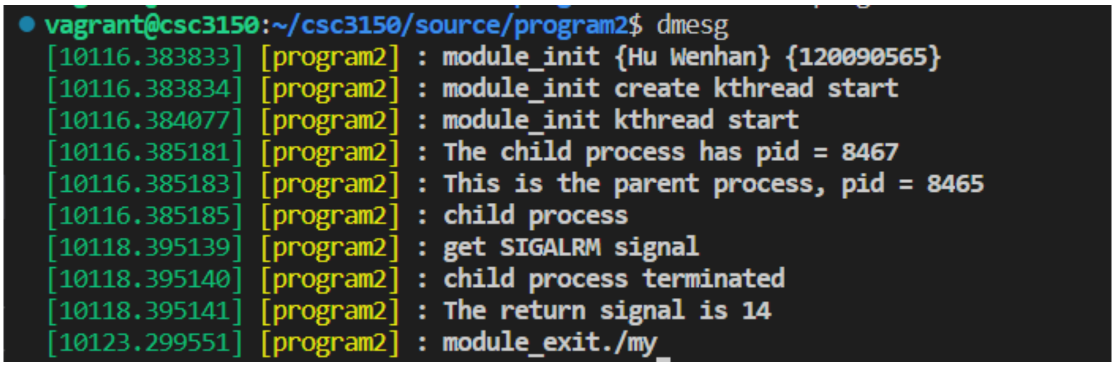
3. *Screenshots of output:*

a) Abort:



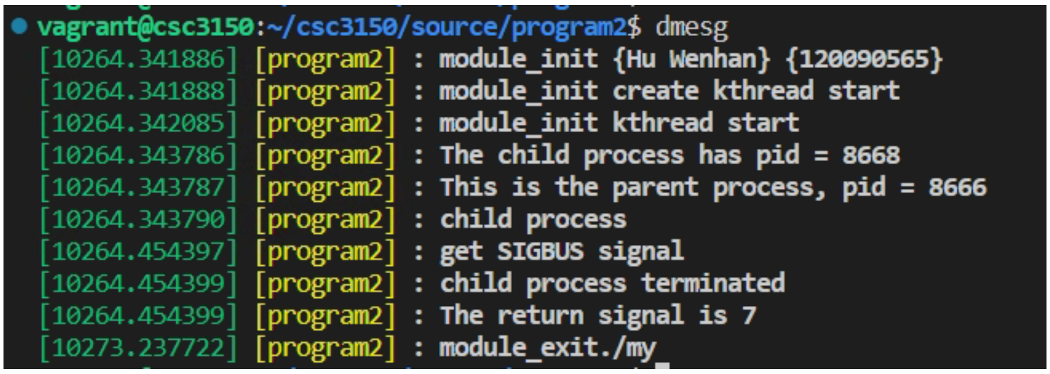
```
vagrant@csc3150:~/csc3150/source/program2$ dmesg
[ 9702.193350] [program2] : module_init {Hu Wenhan} {120090565}
[ 9702.193351] [program2] : module_init create kthread start
[ 9702.193454] [program2] : module_init kthread start
[ 9702.194289] [program2] : The child process has pid = 7797
[ 9702.194290] [program2] : This is the parent process, pid = 7795
[ 9702.194311] [program2] : child process
[ 9702.318000] [program2] : get SIGABRT signal
[ 9702.318002] [program2] : child process terminated
[ 9702.318003] [program2] : The return signal is 6
[ 9709.657849] [program2] : module_exit./my
```

b) Alarm:



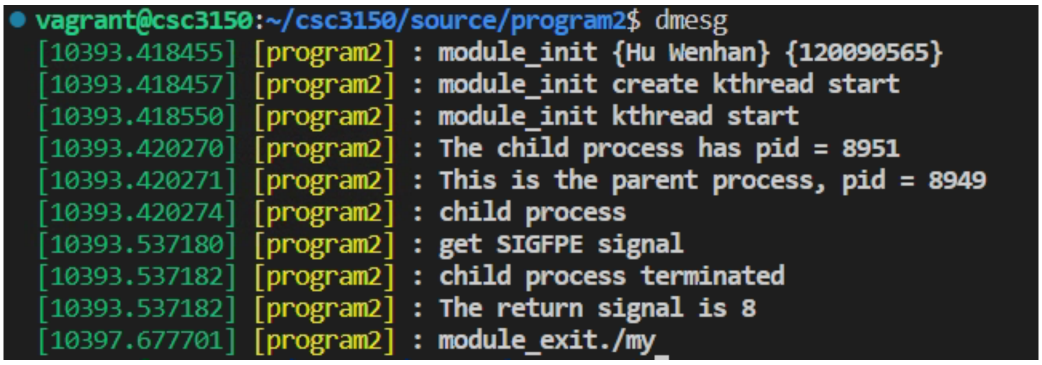
```
vagrant@csc3150:~/csc3150/source/program2$ dmesg
[10116.383833] [program2] : module_init {Hu Wenhan} {120090565}
[10116.383834] [program2] : module_init create kthread start
[10116.384077] [program2] : module_init kthread start
[10116.385181] [program2] : The child process has pid = 8467
[10116.385183] [program2] : This is the parent process, pid = 8465
[10116.385185] [program2] : child process
[10118.395139] [program2] : get SIGALRM signal
[10118.395140] [program2] : child process terminated
[10118.395141] [program2] : The return signal is 14
[10123.299551] [program2] : module_exit./my
```

c) Bus:



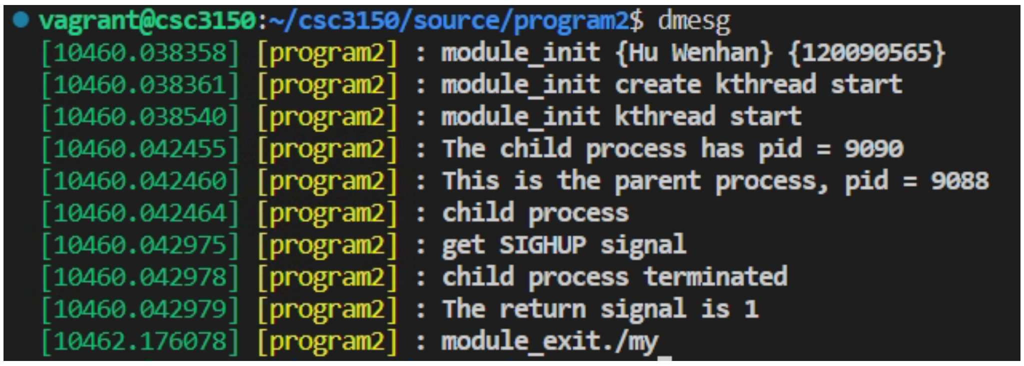
```
vagrant@csc3150:~/csc3150/source/program2$ dmesg
[10264.341886] [program2] : module_init {Hu Wenhan} {120090565}
[10264.341888] [program2] : module_init create kthread start
[10264.342085] [program2] : module_init kthread start
[10264.343786] [program2] : The child process has pid = 8668
[10264.343787] [program2] : This is the parent process, pid = 8666
[10264.343790] [program2] : child process
[10264.454397] [program2] : get SIGBUS signal
[10264.454399] [program2] : child process terminated
[10264.454399] [program2] : The return signal is 7
[10273.237722] [program2] : module_exit./my
```

d) Floating:



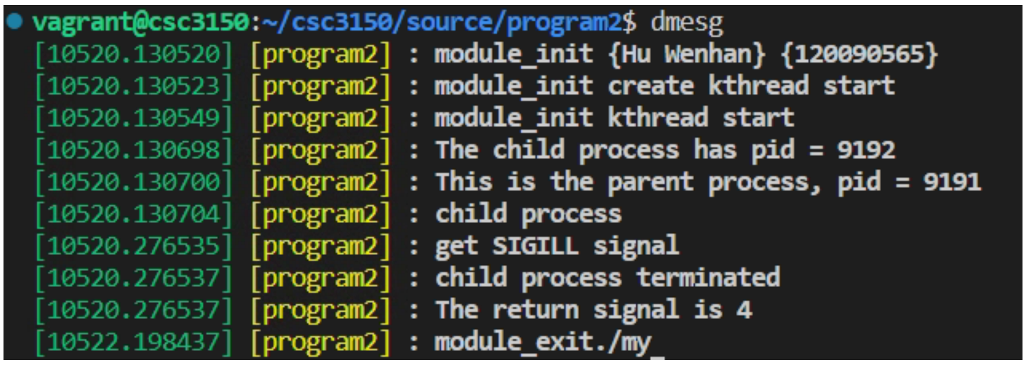
```
vagrant@csc3150:~/csc3150/source/program2$ dmesg
[10393.418455] [program2] : module_init {Hu Wenhan} {120090565}
[10393.418457] [program2] : module_init create kthread start
[10393.418550] [program2] : module_init kthread start
[10393.420270] [program2] : The child process has pid = 8951
[10393.420271] [program2] : This is the parent process, pid = 8949
[10393.420274] [program2] : child process
[10393.537180] [program2] : get SIGFPE signal
[10393.537182] [program2] : child process terminated
[10393.537182] [program2] : The return signal is 8
[10397.677701] [program2] : module_exit./my
```

e) Hangup:



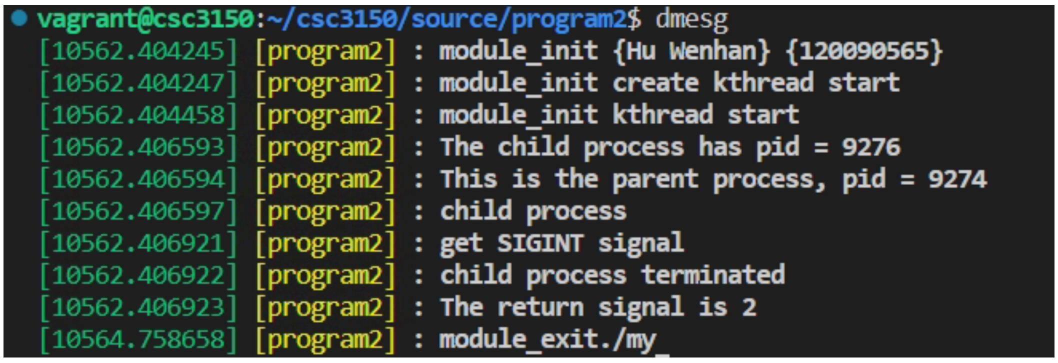
```
● vagrant@csc3150:~/csc3150/source/program2$ dmesg
  [10460.038358] [program2] : module_init {Hu Wenhan} {120090565}
  [10460.038361] [program2] : module_init create kthread start
  [10460.038540] [program2] : module_init kthread start
  [10460.042455] [program2] : The child process has pid = 9090
  [10460.042460] [program2] : This is the parent process, pid = 9088
  [10460.042464] [program2] : child process
  [10460.042975] [program2] : get SIGHUP signal
  [10460.042978] [program2] : child process terminated
  [10460.042979] [program2] : The return signal is 1
  [10462.176078] [program2] : module_exit./my
```

f) Illegal_instr:



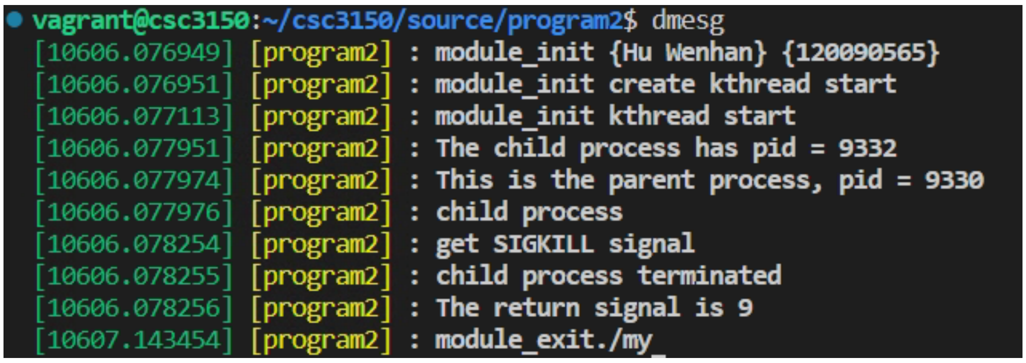
```
● vagrant@csc3150:~/csc3150/source/program2$ dmesg
  [10520.130520] [program2] : module_init {Hu Wenhan} {120090565}
  [10520.130523] [program2] : module_init create kthread start
  [10520.130549] [program2] : module_init kthread start
  [10520.130698] [program2] : The child process has pid = 9192
  [10520.130700] [program2] : This is the parent process, pid = 9191
  [10520.130704] [program2] : child process
  [10520.276535] [program2] : get SIGILL signal
  [10520.276537] [program2] : child process terminated
  [10520.276537] [program2] : The return signal is 4
  [10522.198437] [program2] : module_exit./my
```

g) Interrupt:



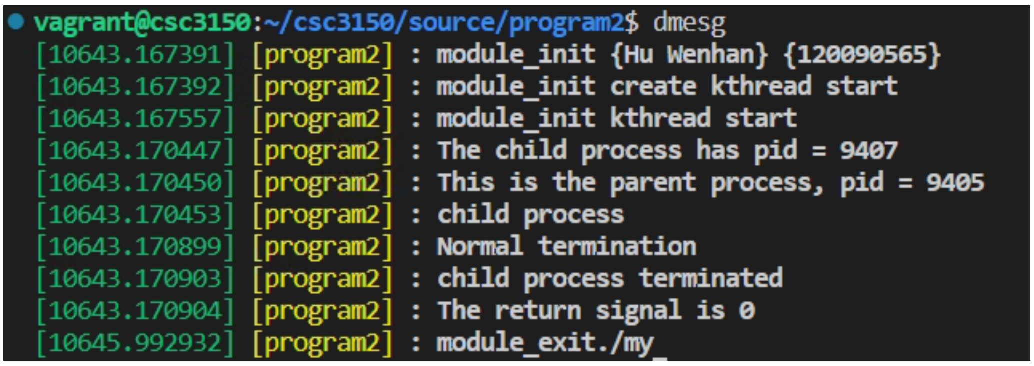
```
● vagrant@csc3150:~/csc3150/source/program2$ dmesg
  [10562.404245] [program2] : module_init {Hu Wenhan} {120090565}
  [10562.404247] [program2] : module_init create kthread start
  [10562.404458] [program2] : module_init kthread start
  [10562.406593] [program2] : The child process has pid = 9276
  [10562.406594] [program2] : This is the parent process, pid = 9274
  [10562.406597] [program2] : child process
  [10562.406921] [program2] : get SIGINT signal
  [10562.406922] [program2] : child process terminated
  [10562.406923] [program2] : The return signal is 2
  [10564.758658] [program2] : module_exit./my
```

h) Kill:



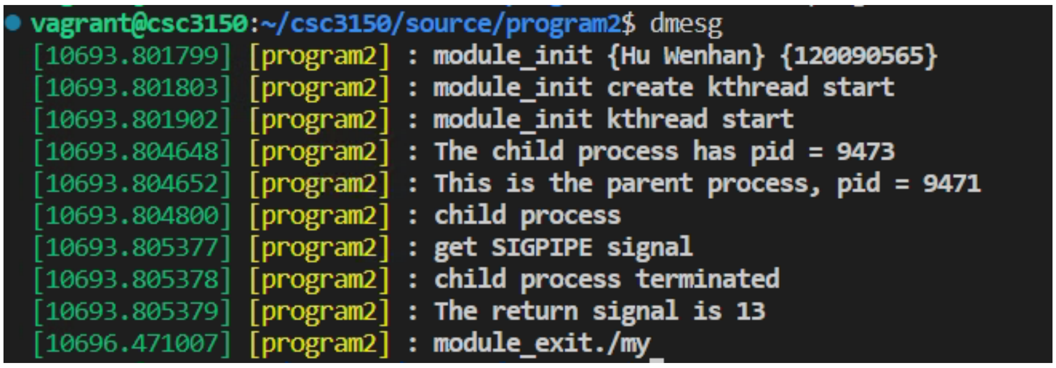
```
● vagrant@csc3150:~/csc3150/source/program2$ dmesg
  [10606.076949] [program2] : module_init {Hu Wenhan} {120090565}
  [10606.076951] [program2] : module_init create kthread start
  [10606.077113] [program2] : module_init kthread start
  [10606.077951] [program2] : The child process has pid = 9332
  [10606.077974] [program2] : This is the parent process, pid = 9330
  [10606.077976] [program2] : child process
  [10606.078254] [program2] : get SIGKILL signal
  [10606.078255] [program2] : child process terminated
  [10606.078256] [program2] : The return signal is 9
  [10607.143454] [program2] : module_exit./my
```

i) Normal:



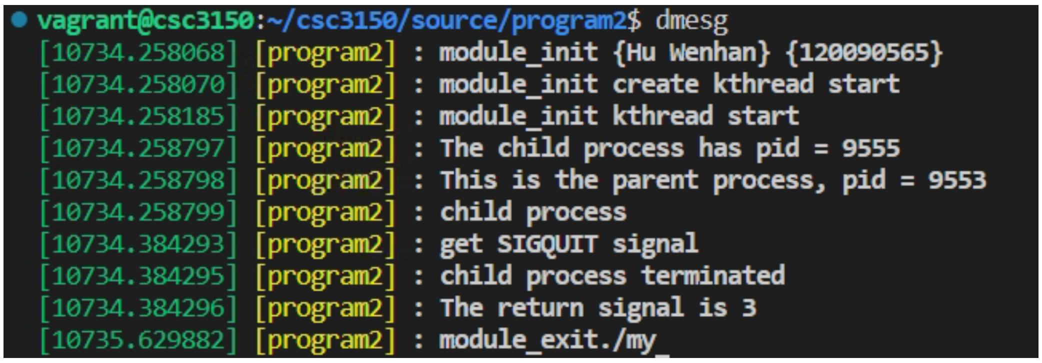
```
vagrant@csc3150:~/csc3150/source/program2$ dmesg
 [10643.167391] [program2] : module_init {Hu Wenhan} {120090565}
 [10643.167392] [program2] : module_init create kthread start
 [10643.167557] [program2] : module_init kthread start
 [10643.170447] [program2] : The child process has pid = 9407
 [10643.170450] [program2] : This is the parent process, pid = 9405
 [10643.170453] [program2] : child process
 [10643.170899] [program2] : Normal termination
 [10643.170903] [program2] : child process terminated
 [10643.170904] [program2] : The return signal is 0
 [10645.992932] [program2] : module_exit./my_
```

j) Pipe:



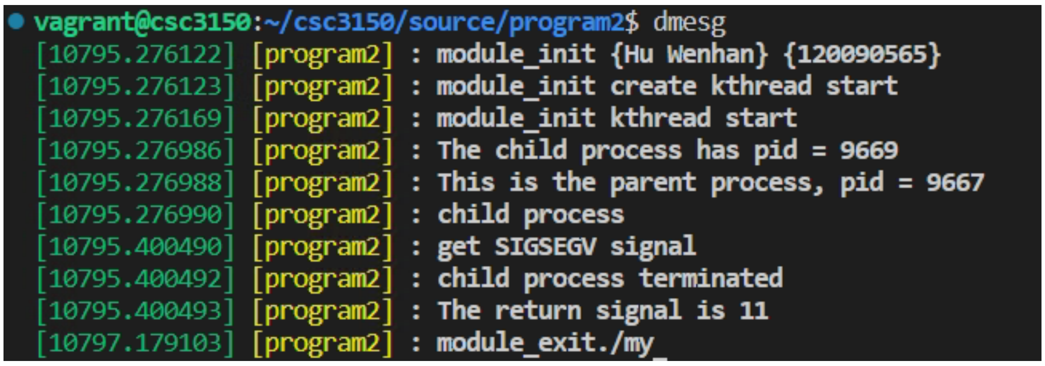
```
vagrant@csc3150:~/csc3150/source/program2$ dmesg
 [10693.801799] [program2] : module_init {Hu Wenhan} {120090565}
 [10693.801803] [program2] : module_init create kthread start
 [10693.801902] [program2] : module_init kthread start
 [10693.804648] [program2] : The child process has pid = 9473
 [10693.804652] [program2] : This is the parent process, pid = 9471
 [10693.804800] [program2] : child process
 [10693.805377] [program2] : get SIGPIPE signal
 [10693.805378] [program2] : child process terminated
 [10693.805379] [program2] : The return signal is 13
 [10696.471007] [program2] : module_exit./my_
```

k) Quit:



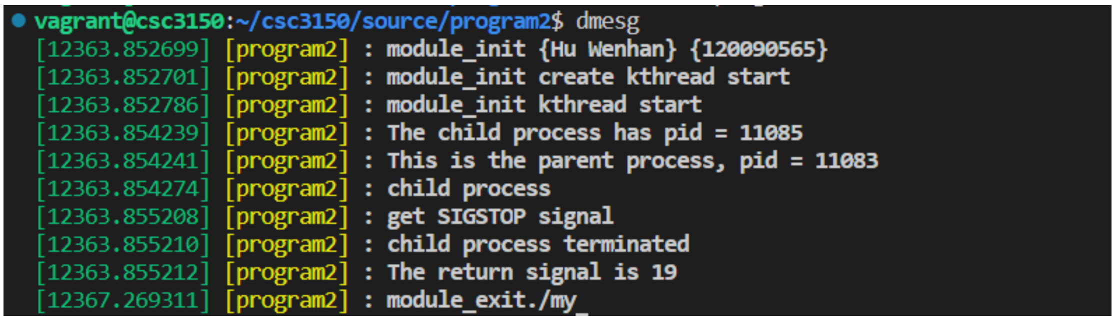
```
vagrant@csc3150:~/csc3150/source/program2$ dmesg
 [10734.258068] [program2] : module_init {Hu Wenhan} {120090565}
 [10734.258070] [program2] : module_init create kthread start
 [10734.258185] [program2] : module_init kthread start
 [10734.258797] [program2] : The child process has pid = 9555
 [10734.258798] [program2] : This is the parent process, pid = 9553
 [10734.258799] [program2] : child process
 [10734.384293] [program2] : get SIGQUIT signal
 [10734.384295] [program2] : child process terminated
 [10734.384296] [program2] : The return signal is 3
 [10735.629882] [program2] : module_exit./my_
```

l) Segment_fault:

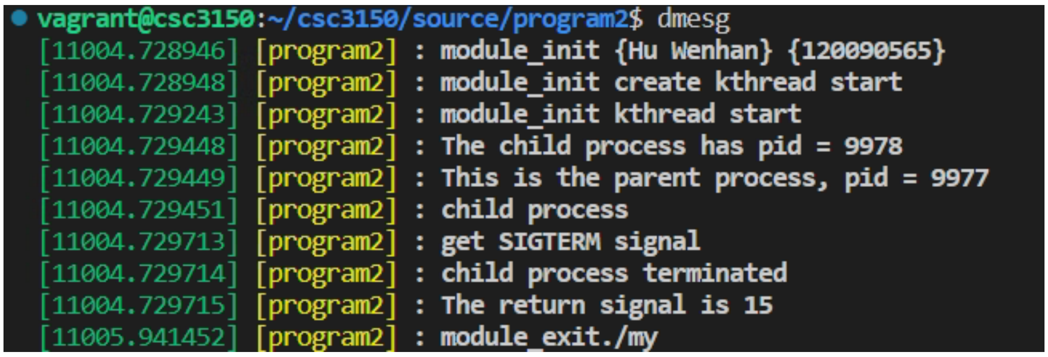


```
vagrant@csc3150:~/csc3150/source/program2$ dmesg
 [10795.276122] [program2] : module_init {Hu Wenhan} {120090565}
 [10795.276123] [program2] : module_init create kthread start
 [10795.276169] [program2] : module_init kthread start
 [10795.276986] [program2] : The child process has pid = 9669
 [10795.276988] [program2] : This is the parent process, pid = 9667
 [10795.276990] [program2] : child process
 [10795.400490] [program2] : get SIGSEGV signal
 [10795.400492] [program2] : child process terminated
 [10795.400493] [program2] : The return signal is 11
 [10797.179103] [program2] : module_exit./my_
```

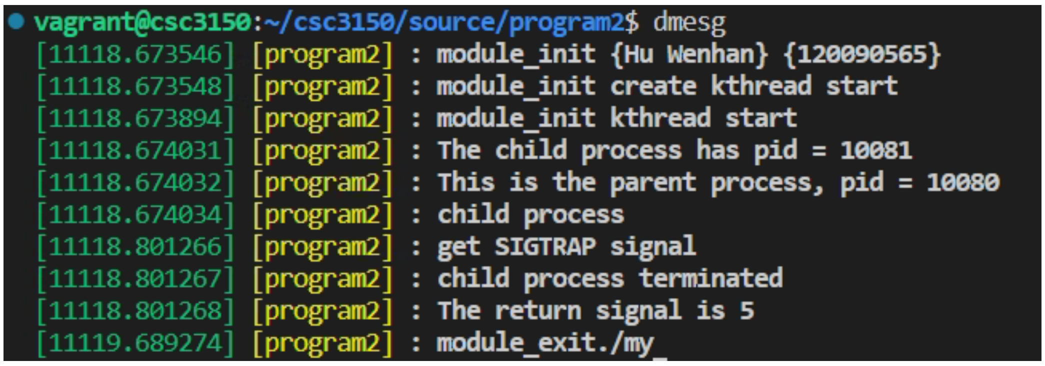
m) Stop:


```
● vagrant@csc3150:~/csc3150/source/program2$ dmesg
  [12363.852699] [program2] : module_init {Hu Wenhan} {120090565}
  [12363.852701] [program2] : module_init create kthread start
  [12363.852786] [program2] : module_init kthread start
  [12363.854239] [program2] : The child process has pid = 11085
  [12363.854241] [program2] : This is the parent process, pid = 11083
  [12363.854274] [program2] : child process
  [12363.855208] [program2] : get SIGSTOP signal
  [12363.855210] [program2] : child process terminated
  [12363.855212] [program2] : The return signal is 19
  [12367.269311] [program2] : module_exit./my_
```

n) Terminate:


```
● vagrant@csc3150:~/csc3150/source/program2$ dmesg
  [11004.728946] [program2] : module_init {Hu Wenhan} {120090565}
  [11004.728948] [program2] : module_init create kthread start
  [11004.729243] [program2] : module_init kthread start
  [11004.729448] [program2] : The child process has pid = 9978
  [11004.729449] [program2] : This is the parent process, pid = 9977
  [11004.729451] [program2] : child process
  [11004.729713] [program2] : get SIGTERM signal
  [11004.729714] [program2] : child process terminated
  [11004.729715] [program2] : The return signal is 15
  [11005.941452] [program2] : module_exit./my
```

o) Trap:


```
● vagrant@csc3150:~/csc3150/source/program2$ dmesg
  [11118.673546] [program2] : module_init {Hu Wenhan} {120090565}
  [11118.673548] [program2] : module_init create kthread start
  [11118.673894] [program2] : module_init kthread start
  [11118.674031] [program2] : The child process has pid = 10081
  [11118.674032] [program2] : This is the parent process, pid = 10080
  [11118.674034] [program2] : child process
  [11118.801266] [program2] : get SIGTRAP signal
  [11118.801267] [program2] : child process terminated
  [11118.801268] [program2] : The return signal is 5
  [11119.689274] [program2] : module_exit./my_
```

4. *What did I learn from this task:*

I learned how to compile kernel. Also I learned a lot of functions and their roles, such as kthread_create(), do_execve(), do_wait(), kernel_clone(), getname_kernel(), and so on. Especially, when the program received the SIGBUS signal from the child process, firstly it get 135, and then I found that the signal should be &0x7f to be turned into 7. Also after finished the task, I think that I am more familiar with the roles of the pointer.

Bonus Task

1. *How to design:*

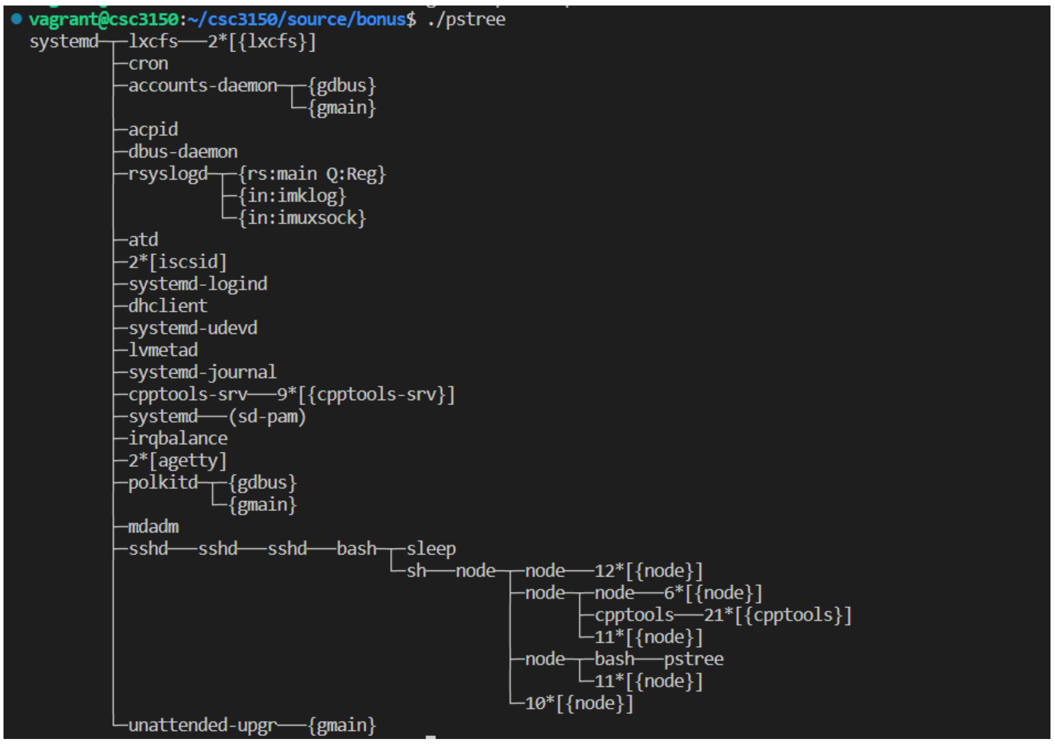
a) This task needed us to create a pstree.c file to implement the linux command pstree. Firstly, I used scandir() to filter out the process directory. Then the main() called ScanProcess() which was used to scan the message of the pstree. Then, still in main() function, the program called BuildTree() function which was used to build the structure of the tree. And within BuildTree() function, the program called BuildNode() function. After that, the main() function called DrawTree() function which was used to print out the pstree. Also like BuildTree() function, DrawTree() called DrawNode() function. After that, the pstree could be printed out on the terminal log.
b) I also wrote another 5 options of pstree, which include -A, -c, -n, -p and -V.
c) -A: I just changed some print-out part within the DrawNode() function in order to print out ASCII symbols.
d) -c: I added some judgement conditions before combination processes, so that the same terms would not be compacted under -c option.
e) -n: I added a sort part when building the tree so that the program could sort nodes by the pid number of each node under -n option.
f) -p: I added a judgement signal so that the program could print out the pid of each node under -p option.
g) -V: The program just printed out the version message without printing the tree.

2. *How to set up the development environment:*

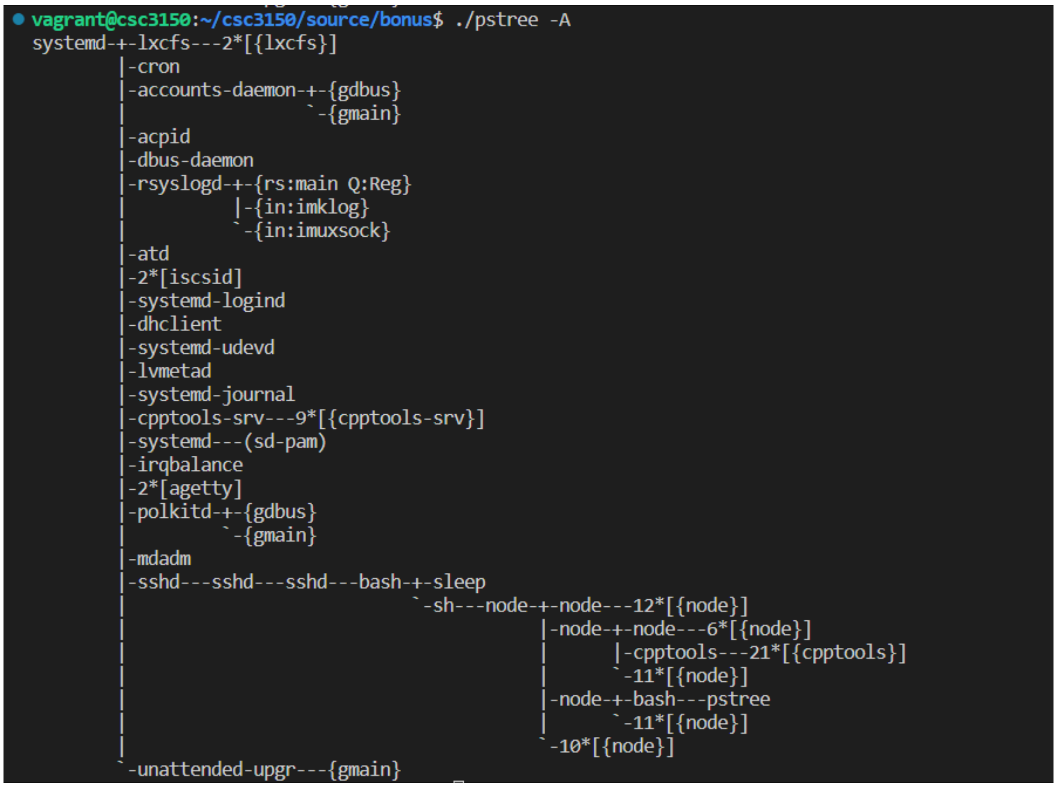
No environment needed to be developed in the bonus task. I just copied the Makefile file of task1, and also developed a c file named pstree.c.

3. *Screenshots of output:*

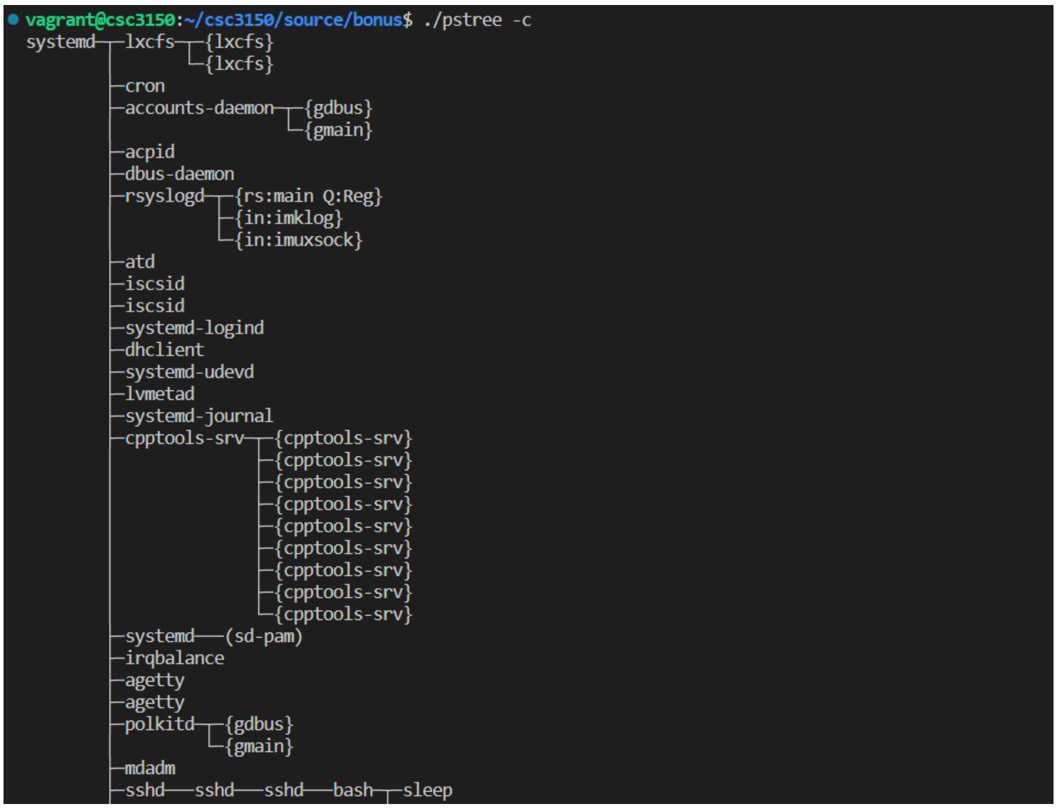
a) pstree:


```
● vagrant@csc3150:~/csc3150/source/bonus$ ./pstree
  systemd─┬─lxcfs───2*[{lxcfs}]
          ├─cron
          ├─accounts-daemon─┬─{gdbus}
          │                 └─{gmain}
          ├─acpid
          ├─dbus-daemon
          ├─rsyslogd─┬─{rs:main Q:Reg}
          │          ├─{in:imklog}
          │          └─{in:imuxsock}
          ├─atd
          ├─2*[iscsid]
          ├─systemd-logind
          ├─dhclient
          ├─systemd-udevd
          ├─lvmetad
          ├─systemd-journal
          ├─cpptools-srv───9*[{cpptools-srv}]
          ├─systemd───(sd-pam)
          ├─irqbalance
          ├─2*[agetty]
          ├─polkitd─┬─{gdbus}
          │         └─{gmain}
          ├─mdadm
          ├─sshd───sshd───sshd───bash─┬─sleep
          │                           └─sh───node─┬─node───12*[{node}]
          │                                       ├─node─┬─node───6*[{node}]
          │                                       │      ├─cpptools───21*[{cpptools}]
          │                                       │      └─11*[{node}]
          │                                       ├─node─┬─bash───pstree
          │                                       │      └─11*[{node}]
          │                                       └─10*[{node}]
          └─unattended-upgr───{gmain}
```

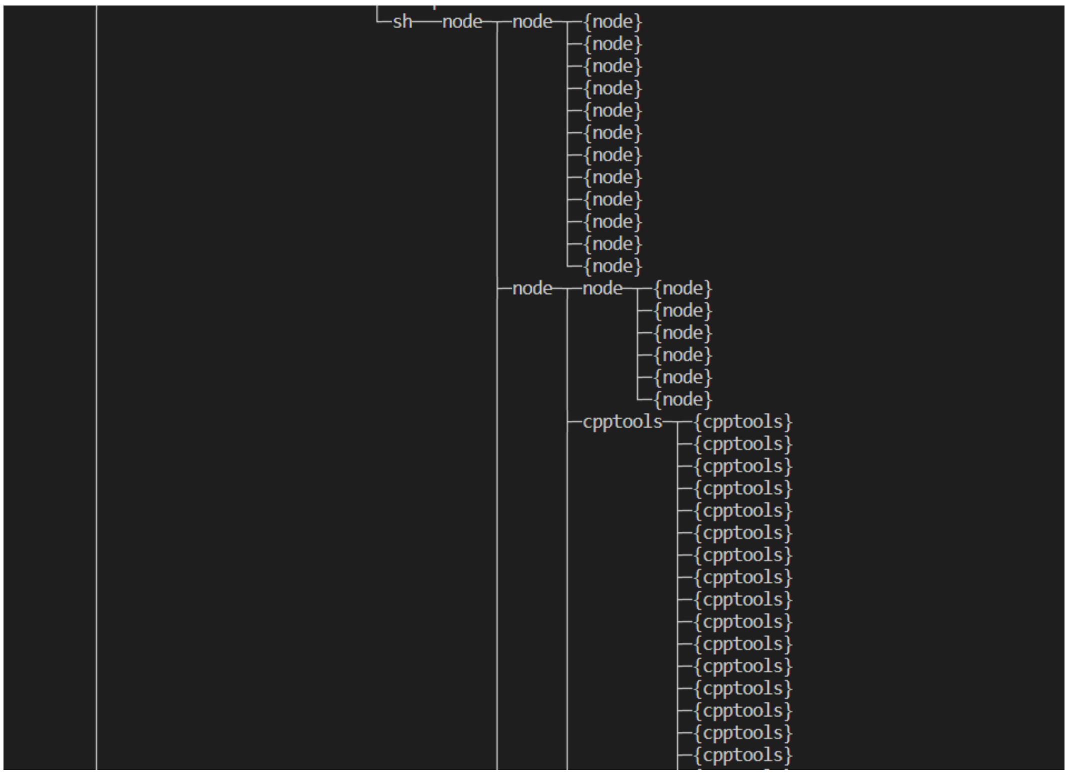
b) pstree -A:


```
● vagrant@csc3150:~/csc3150/source/bonus$ ./pstree -A
  systemd-+-lxcfs---2*[{lxcfs}]
          |-cron
          |-accounts-daemon-+-{gdbus}
          |                 `-{gmain}
          |-acpid
          |-dbus-daemon
          |-rsyslogd-+-{rs:main Q:Reg}
          |          |-{in:imklog}
          |          `-{in:imuxsock}
          |-atd
          |-2*[iscsid]
          |-systemd-logind
          |-dhclient
          |-systemd-udevd
          |-lvmetad
          |-systemd-journal
          |-cpptools-srv---9*[{cpptools-srv}]
          |-systemd---(sd-pam)
          |-irqbalance
          |-2*[agetty]
          |-polkitd-+-{gdbus}
          |         `-{gmain}
          |-mdadm
          |-sshd---sshd---sshd---bash-+-sleep
          |                           `-sh---node-+-node---12*[{node}]
          |                                       |-node-+-node---6*[{node}]
          |                                       |      |-cpptools---21*[{cpptools}]
          |                                       |      `-11*[{node}]
          |                                       |-node-+-bash---pstree
          |                                       |      `-11*[{node}]
          |                                       `-10*[{node}]
          `-unattended-upgr---{gmain}
```

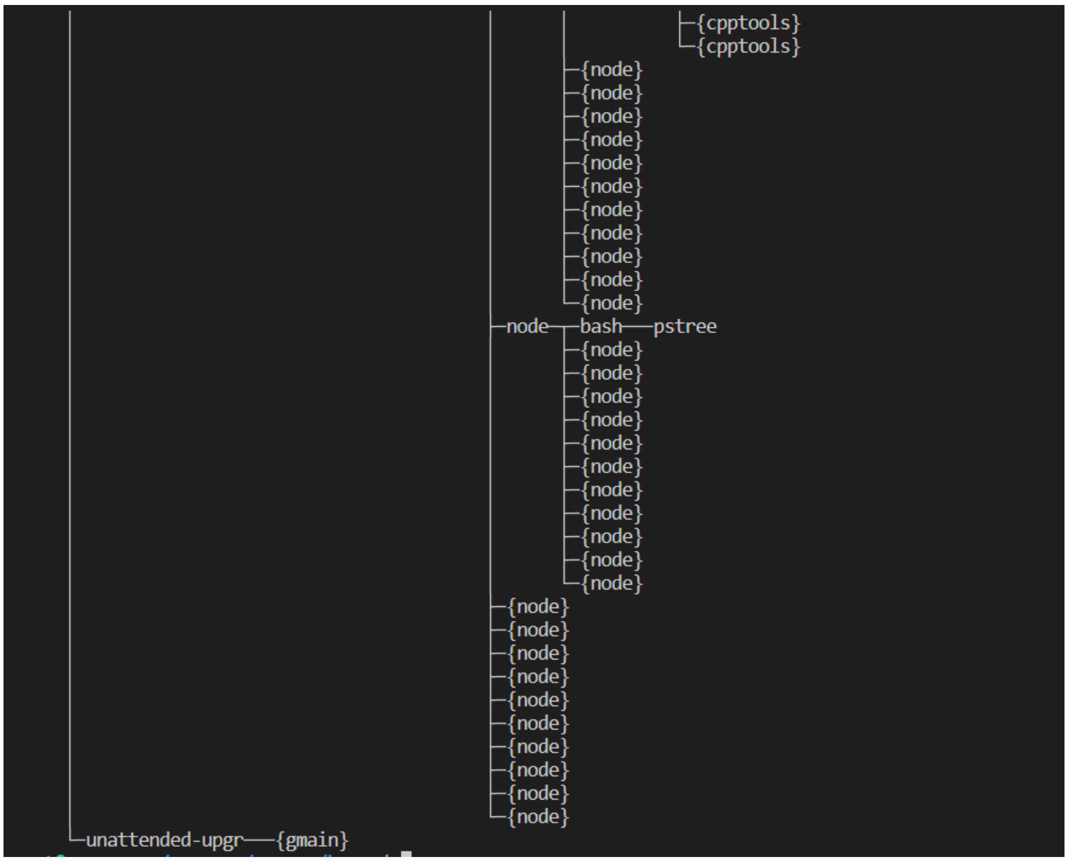
c) pstree -c:



```
● vagrant@csc3150:~/csc3150/source/bonus$ ./pstree -c
systemd─┬─lxcfs─┬─{lxcfs}
        │       └─{lxcfs}
        ├─cron
        ├─accounts-daemon─┬─{gdbus}
        │                 └─{gmain}
        ├─acpid
        ├─dbus-daemon
        ├─rsyslogd─┬─{rs:main Q:Reg}
        │          ├─{in:imklog}
        │          └─{in:imuxsock}
        ├─atd
        ├─iscsid
        ├─iscsid
        ├─systemd-logind
        ├─dhclient
        ├─systemd-udevd
        ├─lvmetad
        ├─systemd-journal
        ├─cpptools-srv─┬─{cpptools-srv}
        │              ├─{cpptools-srv}
        │              ├─{cpptools-srv}
        │              ├─{cpptools-srv}
        │              ├─{cpptools-srv}
        │              ├─{cpptools-srv}
        │              ├─{cpptools-srv}
        │              ├─{cpptools-srv}
        │              └─{cpptools-srv}
        ├─systemd───(sd-pam)
        ├─irqbalance
        ├─agetty
        ├─agetty
        ├─polkitd─┬─{gdbus}
        │         └─{gmain}
        ├─mdadm
        ├─sshd───sshd───sshd───bash─┬─sleep
```

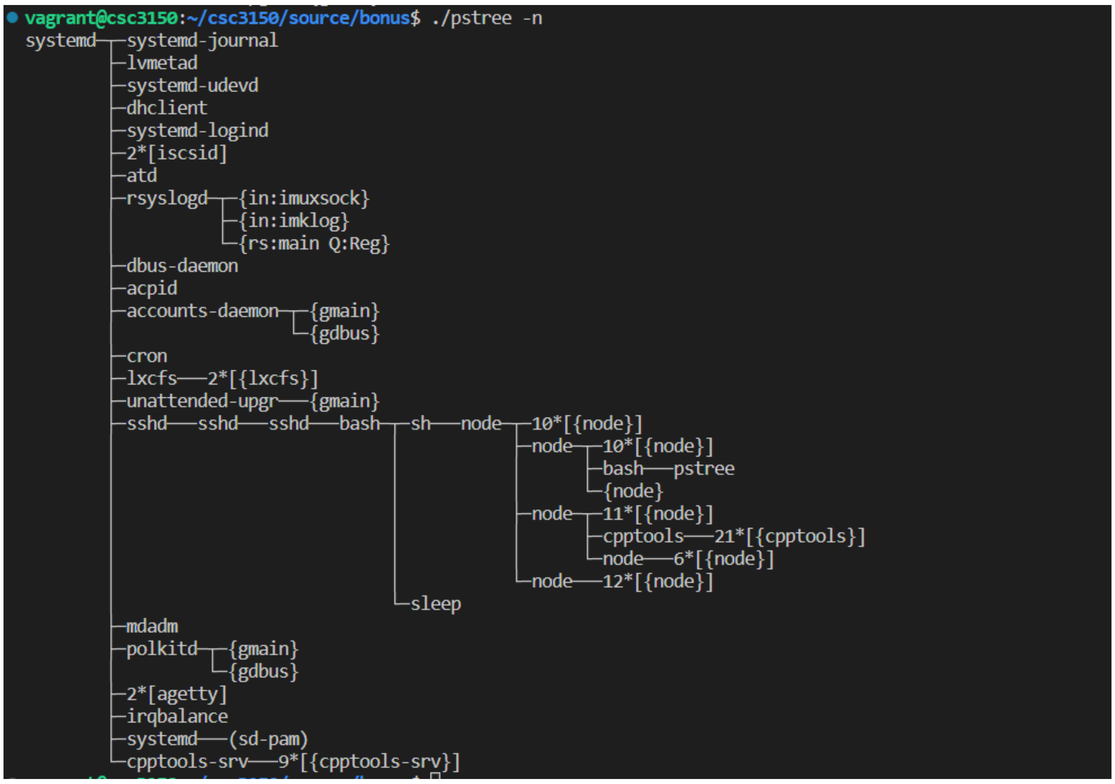


```
│                            └─sh───node─┬─node─┬─{node}
│                                        │      ├─{node}
│                                        │      ├─{node}
│                                        │      ├─{node}
│                                        │      ├─{node}
│                                        │      ├─{node}
│                                        │      ├─{node}
│                                        │      ├─{node}
│                                        │      ├─{node}
│                                        │      ├─{node}
│                                        │      ├─{node}
│                                        │      └─{node}
│                                        ├─node─┬─node─┬─{node}
│                                        │      │      ├─{node}
│                                        │      │      ├─{node}
│                                        │      │      ├─{node}
│                                        │      │      ├─{node}
│                                        │      │      └─{node}
│                                        │      ├─cpptools─┬─{cpptools}
│                                        │      │          ├─{cpptools}
│                                        │      │          ├─{cpptools}
│                                        │      │          ├─{cpptools}
│                                        │      │          ├─{cpptools}
│                                        │      │          ├─{cpptools}
│                                        │      │          ├─{cpptools}
│                                        │      │          ├─{cpptools}
│                                        │      │          ├─{cpptools}
│                                        │      │          ├─{cpptools}
│                                        │      │          ├─{cpptools}
│                                        │      │          ├─{cpptools}
│                                        │      │          ├─{cpptools}
│                                        │      │          ├─{cpptools}
│                                        │      │          ├─{cpptools}
│                                        │      │          ├─{cpptools}
```



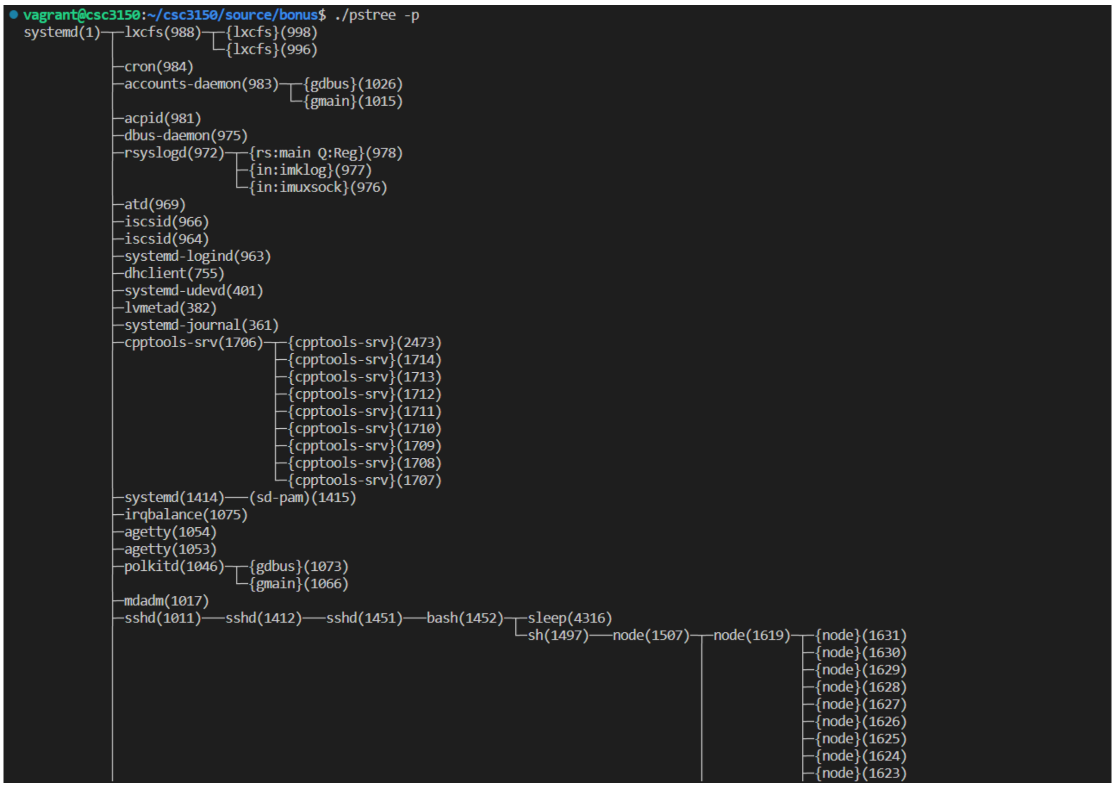
```
    │                                       │      │          ├─{cpptools}
    │                                       │      │          └─{cpptools}
    │                                       │      ├─{node}
    │                                       │      ├─{node}
    │                                       │      ├─{node}
    │                                       │      ├─{node}
    │                                       │      ├─{node}
    │                                       │      ├─{node}
    │                                       │      ├─{node}
    │                                       │      ├─{node}
    │                                       │      ├─{node}
    │                                       │      ├─{node}
    │                                       │      └─{node}
    │                                       ├─node─┬─bash───pstree
    │                                       │      ├─{node}
    │                                       │      ├─{node}
    │                                       │      ├─{node}
    │                                       │      ├─{node}
    │                                       │      ├─{node}
    │                                       │      ├─{node}
    │                                       │      ├─{node}
    │                                       │      ├─{node}
    │                                       │      ├─{node}
    │                                       │      ├─{node}
    │                                       │      └─{node}
    │                                       ├─{node}
    │                                       ├─{node}
    │                                       ├─{node}
    │                                       ├─{node}
    │                                       ├─{node}
    │                                       ├─{node}
    │                                       ├─{node}
    │                                       ├─{node}
    │                                       ├─{node}
    │                                       └─{node}
    └─unattended-upgr───{gmain}
```

d) pstree -n:

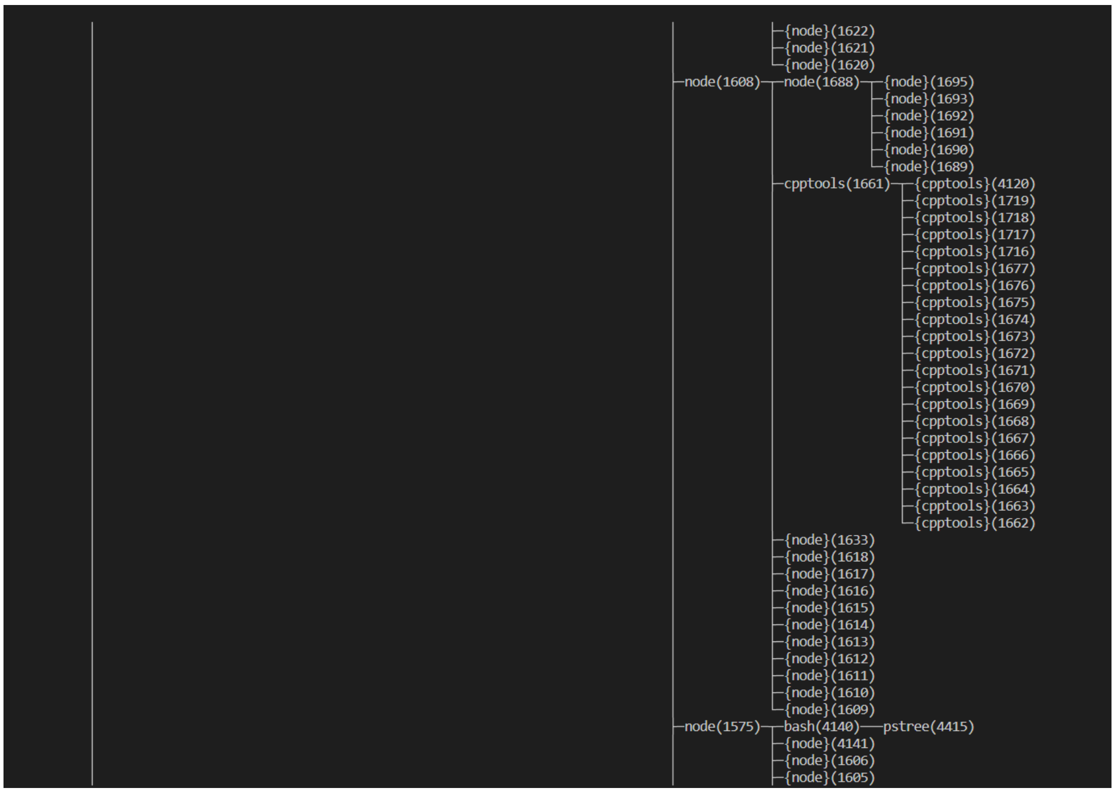


```
vagrant@csc3150:~/csc3150/source/bonus$ ./pstree -n
systemd─┬─systemd-journal
        ├─lvmetad
        ├─systemd-udevd
        ├─dhclient
        ├─systemd-logind
        ├─2*[iscsid]
        ├─atd
        ├─rsyslogd─┬─{in:imuxsock}
        │          ├─{in:imklog}
        │          └─{rs:main Q:Reg}
        ├─dbus-daemon
        ├─acpid
        ├─accounts-daemon─┬─{gmain}
        │                 └─{gdbus}
        ├─cron
        ├─lxcfs───2*[{lxcfs}]
        ├─unattended-upgr───{gmain}
        ├─sshd───sshd───sshd───bash─┬─sh───node─┬─10*[{node}]
        │                           │           ├─node─┬─10*[{node}]
        │                           │           │      ├─bash───pstree
        │                           │           │      └─{node}
        │                           │           ├─node─┬─11*[{node}]
        │                           │           │      ├─cpptools───21*[{cpptools}]
        │                           │           │      └─node───6*[{node}]
        │                           │           └─node───12*[{node}]
        │                           └─sleep
        ├─mdadm
        ├─polkitd─┬─{gmain}
        │         └─{gdbus}
        ├─2*[agetty]
        ├─irqbalance
        ├─systemd───(sd-pam)
        └─cpptools-srv───9*[{cpptools-srv}]
```

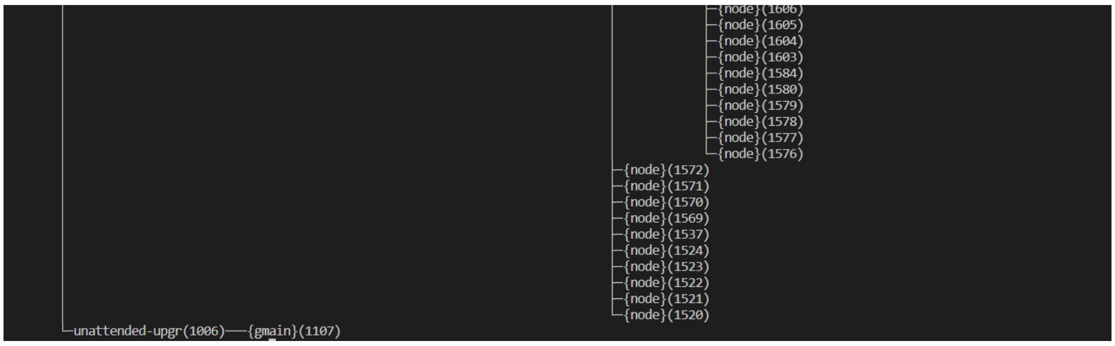
e) pstree -p:



```
● vagrant@csc3150:~/csc3150/source/bonus$ ./pstree -p
  systemd(1)─┬─lxcfs(988)─┬─{lxcfs}(998)
             │            └─{lxcfs}(996)
             ├─cron(984)
             ├─accounts-daemon(983)─┬─{gdbus}(1026)
             │                      └─{gmain}(1015)
             ├─acpid(981)
             ├─dbus-daemon(975)
             ├─rsyslogd(972)─┬─{rs:main Q:Reg}(978)
             │               ├─{in:imklog}(977)
             │               └─{in:imuxsock}(976)
             ├─atd(969)
             ├─iscsid(966)
             ├─iscsid(964)
             ├─systemd-logind(963)
             ├─dhclient(755)
             ├─systemd-udevd(401)
             ├─lvmetad(382)
             ├─systemd-journal(361)
             ├─cpptools-srv(1706)─┬─{cpptools-srv}(2473)
             │                    ├─{cpptools-srv}(1714)
             │                    ├─{cpptools-srv}(1713)
             │                    ├─{cpptools-srv}(1712)
             │                    ├─{cpptools-srv}(1711)
             │                    ├─{cpptools-srv}(1710)
             │                    ├─{cpptools-srv}(1709)
             │                    ├─{cpptools-srv}(1708)
             │                    └─{cpptools-srv}(1707)
             ├─systemd(1414)───(sd-pam)(1415)
             ├─irqbalance(1075)
             ├─agetty(1054)
             ├─agetty(1053)
             ├─polkitd(1046)─┬─{gdbus}(1073)
             │               └─{gmain}(1066)
             ├─mdadm(1017)
             ├─sshd(1011)───sshd(1412)───sshd(1451)───bash(1452)─┬─sleep(4316)
             │                                                   └─sh(1497)───node(1507)─┬─node(1619)─┬─{node}(1631)
             │                                                                           │            ├─{node}(1630)
             │                                                                           │            ├─{node}(1629)
             │                                                                           │            ├─{node}(1628)
             │                                                                           │            ├─{node}(1627)
             │                                                                           │            ├─{node}(1626)
             │                                                                           │            ├─{node}(1625)
             │                                                                           │            ├─{node}(1624)
             │                                                                           │            ├─{node}(1623)
```

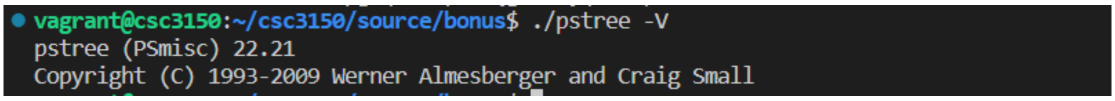


```
             │                                                                           │            ├─{node}(1622)
             │                                                                           │            ├─{node}(1621)
             │                                                                           │            └─{node}(1620)
             │                                                                           ├─node(1608)─┬─node(1688)─┬─{node}(1695)
             │                                                                           │            │            ├─{node}(1693)
             │                                                                           │            │            ├─{node}(1692)
             │                                                                           │            │            ├─{node}(1691)
             │                                                                           │            │            ├─{node}(1690)
             │                                                                           │            │            └─{node}(1689)
             │                                                                           │            ├─cpptools(1661)─┬─{cpptools}(4120)
             │                                                                           │            │                ├─{cpptools}(1719)
             │                                                                           │            │                ├─{cpptools}(1718)
             │                                                                           │            │                ├─{cpptools}(1717)
             │                                                                           │            │                ├─{cpptools}(1716)
             │                                                                           │            │                ├─{cpptools}(1677)
             │                                                                           │            │                ├─{cpptools}(1676)
             │                                                                           │            │                ├─{cpptools}(1675)
             │                                                                           │            │                ├─{cpptools}(1674)
             │                                                                           │            │                ├─{cpptools}(1673)
             │                                                                           │            │                ├─{cpptools}(1672)
             │                                                                           │            │                ├─{cpptools}(1671)
             │                                                                           │            │                ├─{cpptools}(1670)
             │                                                                           │            │                ├─{cpptools}(1669)
             │                                                                           │            │                ├─{cpptools}(1668)
             │                                                                           │            │                ├─{cpptools}(1667)
             │                                                                           │            │                ├─{cpptools}(1666)
             │                                                                           │            │                ├─{cpptools}(1665)
             │                                                                           │            │                ├─{cpptools}(1664)
             │                                                                           │            │                ├─{cpptools}(1663)
             │                                                                           │            │                └─{cpptools}(1662)
             │                                                                           │            ├─{node}(1633)
             │                                                                           │            ├─{node}(1618)
             │                                                                           │            ├─{node}(1617)
             │                                                                           │            ├─{node}(1616)
             │                                                                           │            ├─{node}(1615)
             │                                                                           │            ├─{node}(1614)
             │                                                                           │            ├─{node}(1613)
             │                                                                           │            ├─{node}(1612)
             │                                                                           │            ├─{node}(1611)
             │                                                                           │            ├─{node}(1610)
             │                                                                           │            └─{node}(1609)
             │                                                                           ├─node(1575)─┬─bash(4140)───pstree(4415)
             │                                                                           │            ├─{node}(4141)
             │                                                                           │            ├─{node}(1606)
             │                                                                           │            ├─{node}(1605)
```



```
        │                                                                           │            ├─{node}(1606)
        │                                                                           │            ├─{node}(1605)
        │                                                                           │            ├─{node}(1604)
        │                                                                           │            ├─{node}(1603)
        │                                                                           │            ├─{node}(1584)
        │                                                                           │            ├─{node}(1580)
        │                                                                           │            ├─{node}(1579)
        │                                                                           │            ├─{node}(1578)
        │                                                                           │            ├─{node}(1577)
        │                                                                           │            └─{node}(1576)
        │                                                                           ├─{node}(1572)
        │                                                                           ├─{node}(1571)
        │                                                                           ├─{node}(1570)
        │                                                                           ├─{node}(1569)
        │                                                                           ├─{node}(1537)
        │                                                                           ├─{node}(1524)
        │                                                                           ├─{node}(1523)
        │                                                                           ├─{node}(1522)
        │                                                                           ├─{node}(1521)
        │                                                                           └─{node}(1520)
        └─unattended-upgr(1006)───{gmain}(1107)
```

f) pstree -V:



```
vagrant@csc3150:~/csc3150/source/bonus$ ./pstree -V
pstree (PSmisc) 22.21
Copyright (C) 1993-2009 Werner Almesberger and Craig Small
```

4. What did I learn from this task:

I learned a lot of things from this task. Generally, I think I have learnt how to study though Google on my own. For knowledge, I learned a lot of functions of c language, such as scandir(), strcmp(), strcpy(), and so on. Especially, I learned many things on how to use array. To build the tree, I used some knowledge of the linked list. And to combine same nodes, I also reviewed the knowledge of recursion. There were so many things I have got from this task. However, I think the biggest gain is the development on my ability of debug.