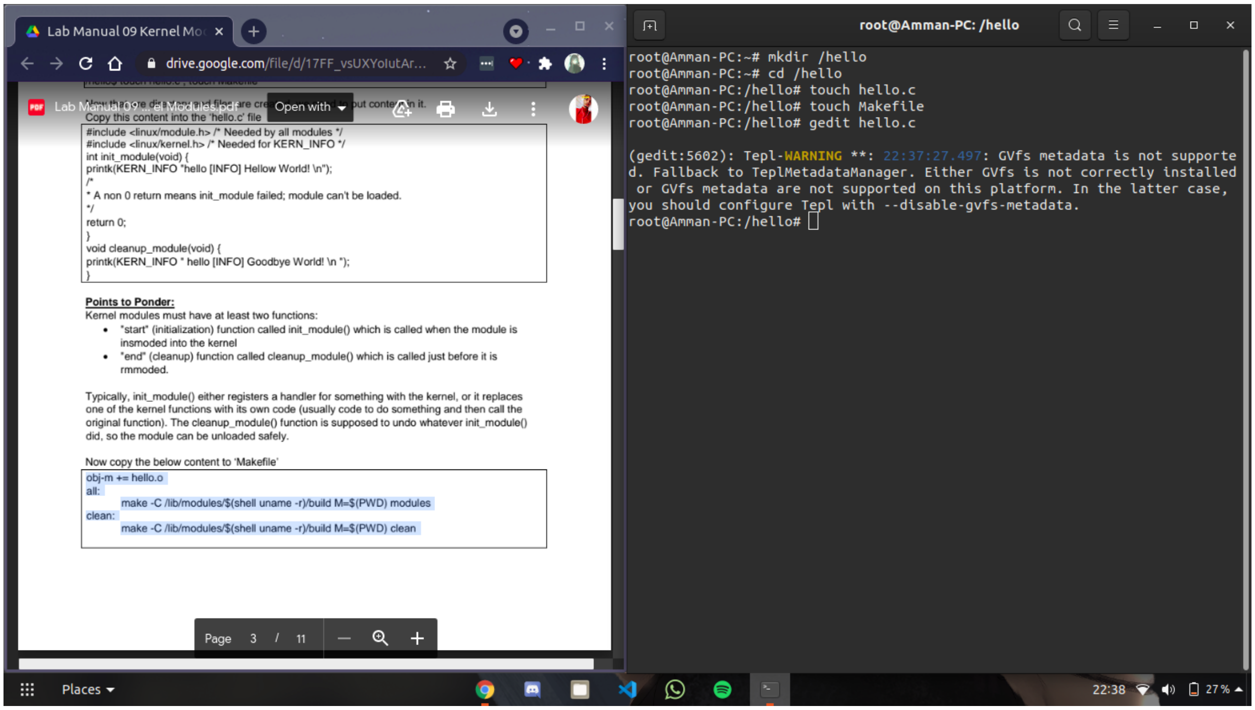
OS LAB 08

QUESTION NO 01

WRITING KERNEL MODULES



/hello$ touch hello.c ; touch Makefile

Now that are directory and files are created, we need to put content in it.
Copy this content into the 'hello.c' file

```
#include <linux/module.h> /* Needed by all modules */
#include <linux/kernel.h> /* Needed for KERN_INFO */
int init_module(void) {
printk(KERN_INFO "hello [INFO] Hellow World! \n");
/*
* A non 0 return means init_module failed; module can't be loaded.
*/
return 0;
}
void cleanup_module(void) {
printk(KERN_INFO " hello [INFO] Goodbye World! \n ");
}
```

Points to Ponder:
Kernel modules must have at least two functions:

- "start" (initialization) function called init_module() which is called when the module is insmoded into the kernel
- "end" (cleanup) function called cleanup_module() which is called just before it is rmmoded.

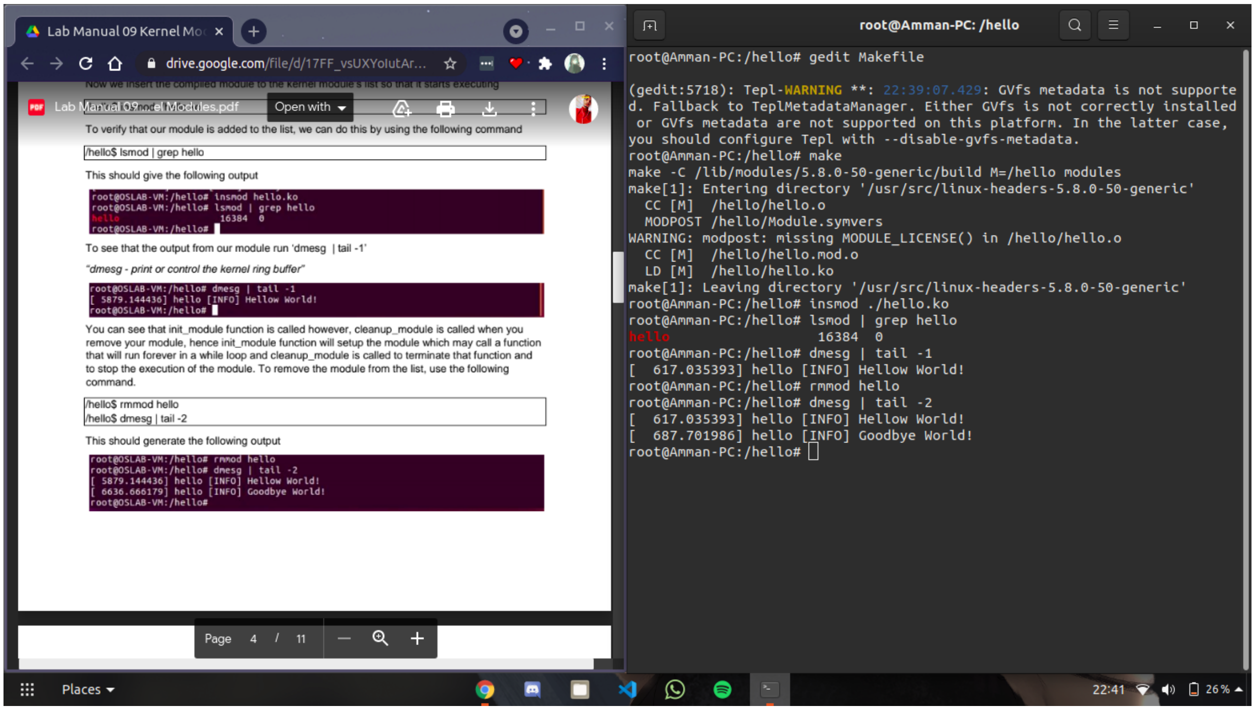
Typically, init_module() either registers a handler for something with the kernel, or it replaces one of the kernel functions with its own code (usually code to do something and then call the original function). The cleanup_module() function is supposed to undo whatever init_module() did, so the module can be unloaded safely.

Now copy the below content to 'Makefile'

```
obj-m += hello.o
all:
        make -C /lib/modules/$(shell uname -r)/build M=$(PWD) modules
clean:
        make -C /lib/modules/$(shell uname -r)/build M=$(PWD) clean
```

```
root@Amman-PC:~# mkdir /hello
root@Amman-PC:~# cd /hello
root@Amman-PC:/hello# touch hello.c
root@Amman-PC:/hello# touch Makefile
root@Amman-PC:/hello# gedit hello.c

(gedit:5602): Tepl-WARNING **: 22:37:27.497: GVfs metadata is not supporte
d. Fallback to TeplMetadataManager. Either GVfs is not correctly installed
 or GVfs metadata are not supported on this platform. In the latter case,
you should configure Tepl with --disable-gvfs-metadata.
root@Amman-PC:/hello# 
```



Now we insert the compiled module to the kernel module's list so that it starts executing

To verify that our module is added to the list, we can do this by using the following command

```
/hello$ lsmod | grep hello
```

This should give the following output

```
root@OSLAB-VM:/hello# insmod hello.ko
root@OSLAB-VM:/hello# lsmod | grep hello
hello                  16384  0
root@OSLAB-VM:/hello# 
```

To see that the output from our module run 'dmesg | tail -1'

"dmesg - print or control the kernel ring buffer"

```
root@OSLAB-VM:/hello# dmesg | tail -1
[ 5879.144436] hello [INFO] Hellow World!
root@OSLAB-VM:/hello# 
```

You can see that init_module function is called however, cleanup_module is called when you remove your module, hence init_module function will setup the module which may call a function that will run forever in a while loop and cleanup_module is called to terminate that function and to stop the execution of the module. To remove the module from the list, use the following command.

```
/hello$ rmmod hello
/hello$ dmesg | tail -2
```

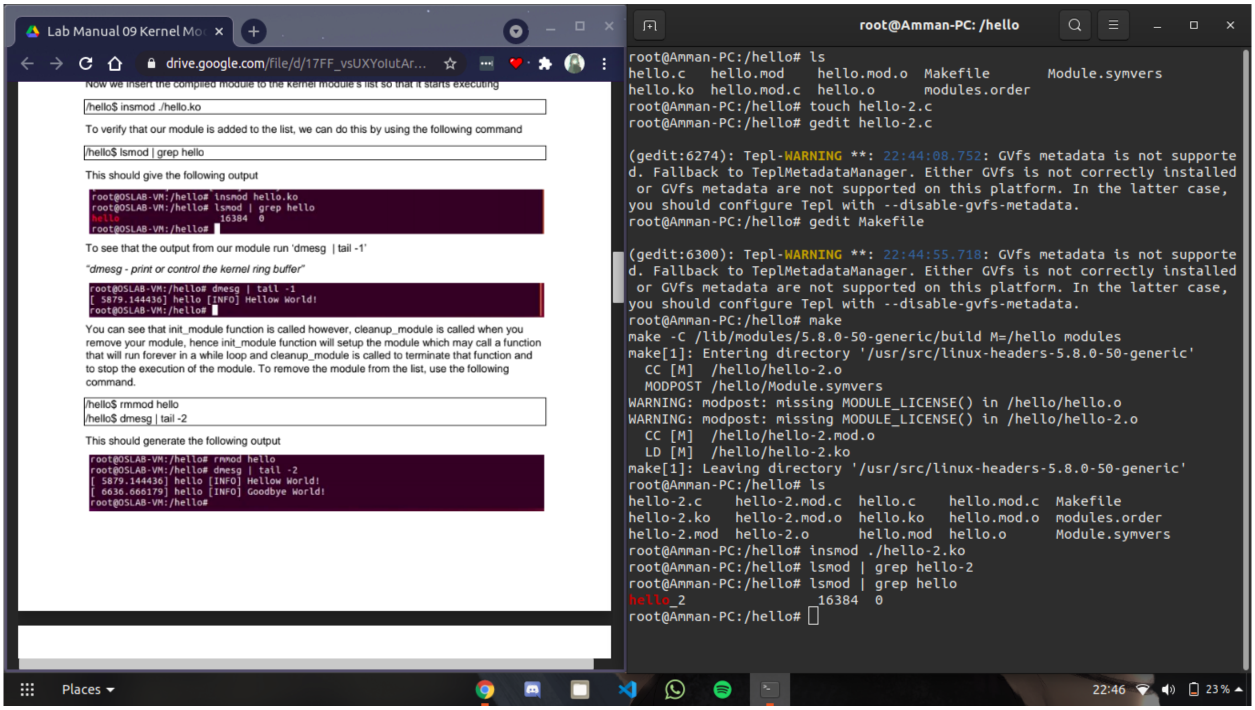
This should generate the following output

```
root@OSLAB-VM:/hello# rmmod hello
root@OSLAB-VM:/hello# dmesg | tail -2
[ 5879.144436] hello [INFO] Hellow World!
[ 6636.666179] hello [INFO] Goodbye World!
root@OSLAB-VM:/hello#
```

```
root@Amman-PC:/hello# gedit Makefile

(gedit:5718): Tepl-WARNING **: 22:39:07.429: GVfs metadata is not supporte
d. Fallback to TeplMetadataManager. Either GVfs is not correctly installed
 or GVfs metadata are not supported on this platform. In the latter case,
you should configure Tepl with --disable-gvfs-metadata.
root@Amman-PC:/hello# make
make -C /lib/modules/5.8.0-50-generic/build M=/hello modules
make[1]: Entering directory '/usr/src/linux-headers-5.8.0-50-generic'
  CC [M]  /hello/hello.o
  MODPOST /hello/Module.symvers
WARNING: modpost: missing MODULE_LICENSE() in /hello/hello.o
  CC [M]  /hello/hello.mod.o
  LD [M]  /hello/hello.ko
make[1]: Leaving directory '/usr/src/linux-headers-5.8.0-50-generic'
root@Amman-PC:/hello# insmod ./hello.ko
root@Amman-PC:/hello# lsmod | grep hello
hello                  16384  0
root@Amman-PC:/hello# dmesg | tail -1
[  617.035393] hello [INFO] Hellow World!
root@Amman-PC:/hello# rmmod hello
root@Amman-PC:/hello# dmesg | tail -2
[  617.035393] hello [INFO] Hellow World!
[  687.701986] hello [INFO] Goodbye World!
root@Amman-PC:/hello# 
```

WITH MODULE_INIT() & MODULE_EXIT() I.E. INIT MACROS



Now we insert the compiled module to the kernel module's list so that it starts executing

```
/hello$ insmod ./hello.ko
```

To verify that our module is added to the list, we can do this by using the following command

```
/hello$ lsmod | grep hello
```

This should give the following output

```
root@OSLAB-VM:/hello# insmod hello.ko
root@OSLAB-VM:/hello# lsmod | grep hello
hello                  16384  0
root@OSLAB-VM:/hello#
```

To see that the output from our module run 'dmesg | tail -1'

"dmesg - print or control the kernel ring buffer"

```
root@OSLAB-VM:/hello# dmesg | tail -1
[ 5879.144436] hello [INFO] Hellow World!
root@OSLAB-VM:/hello#
```

You can see that init_module function is called however, cleanup_module is called when you remove your module, hence init_module function will setup the module which may call a function that will run forever in a while loop and cleanup_module is called to terminate that function and to stop the execution of the module. To remove the module from the list, use the following command.

```
/hello$ rmmod hello
/hello$ dmesg | tail -2
```

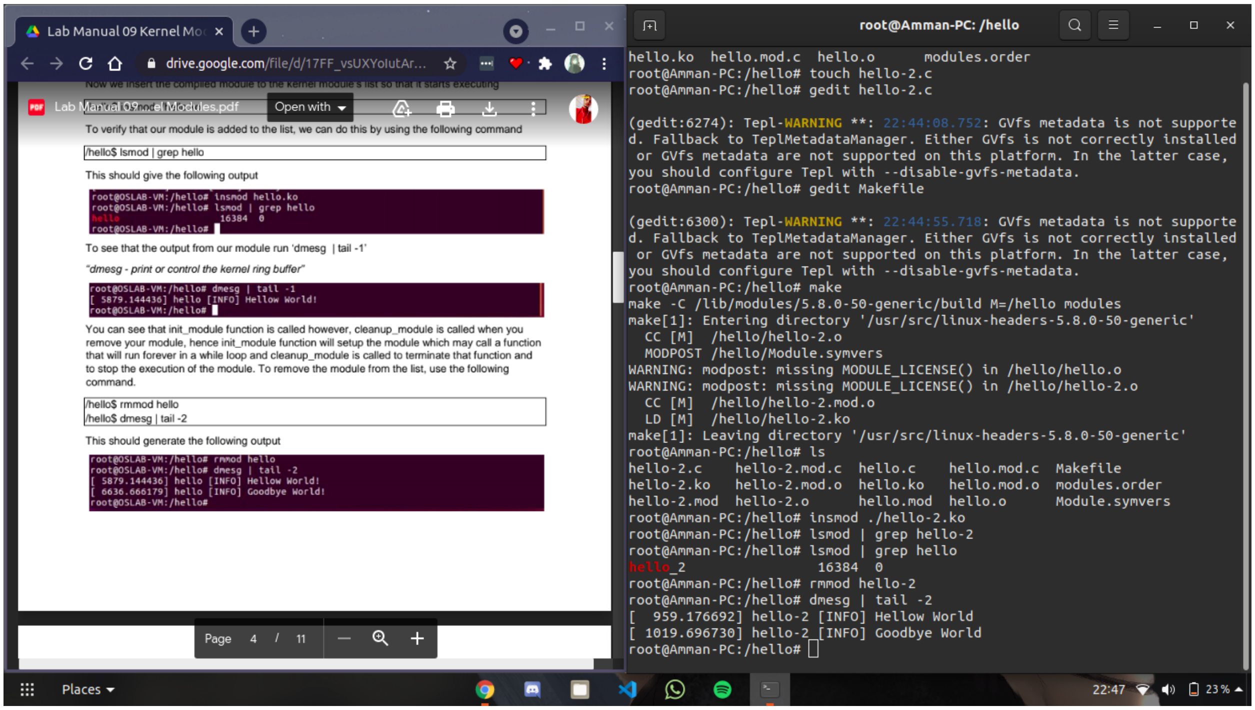
This should generate the following output

```
root@OSLAB-VM:/hello# rmmod hello
root@OSLAB-VM:/hello# dmesg | tail -2
[ 5879.144436] hello [INFO] Hellow World!
[ 6636.666179] hello [INFO] Goodbye World!
root@OSLAB-VM:/hello#
```

```
root@Amman-PC:/hello# ls
hello.c   hello.mod    hello.mod.o  Makefile       Module.symvers
hello.ko  hello.mod.c  hello.o      modules.order
root@Amman-PC:/hello# touch hello-2.c
root@Amman-PC:/hello# gedit hello-2.c

(gedit:6274): Tepl-WARNING **: 22:44:08.752: GVfs metadata is not supporte
d. Fallback to TeplMetadataManager. Either GVfs is not correctly installed
 or GVfs metadata are not supported on this platform. In the latter case,
you should configure Tepl with --disable-gvfs-metadata.
root@Amman-PC:/hello# gedit Makefile

(gedit:6300): Tepl-WARNING **: 22:44:55.718: GVfs metadata is not supporte
d. Fallback to TeplMetadataManager. Either GVfs is not correctly installed
 or GVfs metadata are not supported on this platform. In the latter case,
you should configure Tepl with --disable-gvfs-metadata.
root@Amman-PC:/hello# make
make -C /lib/modules/5.8.0-50-generic/build M=/hello modules
make[1]: Entering directory '/usr/src/linux-headers-5.8.0-50-generic'
  CC [M]  /hello/hello-2.o
  MODPOST /hello/Module.symvers
WARNING: modpost: missing MODULE_LICENSE() in /hello/hello.o
WARNING: modpost: missing MODULE_LICENSE() in /hello/hello-2.o
  CC [M]  /hello/hello-2.mod.o
  LD [M]  /hello/hello-2.ko
make[1]: Leaving directory '/usr/src/linux-headers-5.8.0-50-generic'
root@Amman-PC:/hello# ls
hello-2.c    hello-2.mod.c  hello.c    hello.mod.c  Makefile
hello-2.ko   hello-2.mod.o  hello.ko   hello.mod.o  modules.order
hello-2.mod  hello-2.o      hello.mod  hello.o      Module.symvers
root@Amman-PC:/hello# insmod ./hello-2.ko
root@Amman-PC:/hello# lsmod | grep hello-2
root@Amman-PC:/hello# lsmod | grep hello
hello_2                16384  0
root@Amman-PC:/hello# 
```



Now we insert the compiled module to the kernel module's list so that it starts executing

To verify that our module is added to the list, we can do this by using the following command

/hello$ lsmod | grep hello

This should give the following output

root@OSLAB-VM:/hello# insmod hello.ko
root@OSLAB-VM:/hello# lsmod | grep hello
hello   16384 0
root@OSLAB-VM:/hello#

To see that the output from our module run 'dmesg | tail -1'

"dmesg - print or control the kernel ring buffer"

root@OSLAB-VM:/hello# dmesg | tail -1
[5879.144436] hello [INFO] Hellow World!
root@OSLAB-VM:/hello#

You can see that init_module function is called however, cleanup_module is called when you remove your module, hence init_module function will setup the module which may call a function that will run forever in a while loop and cleanup_module is called to terminate that function and to stop the execution of the module. To remove the module from the list, use the following command.

/hello$ rmmod hello
/hello$ dmesg | tail -2

This should generate the following output

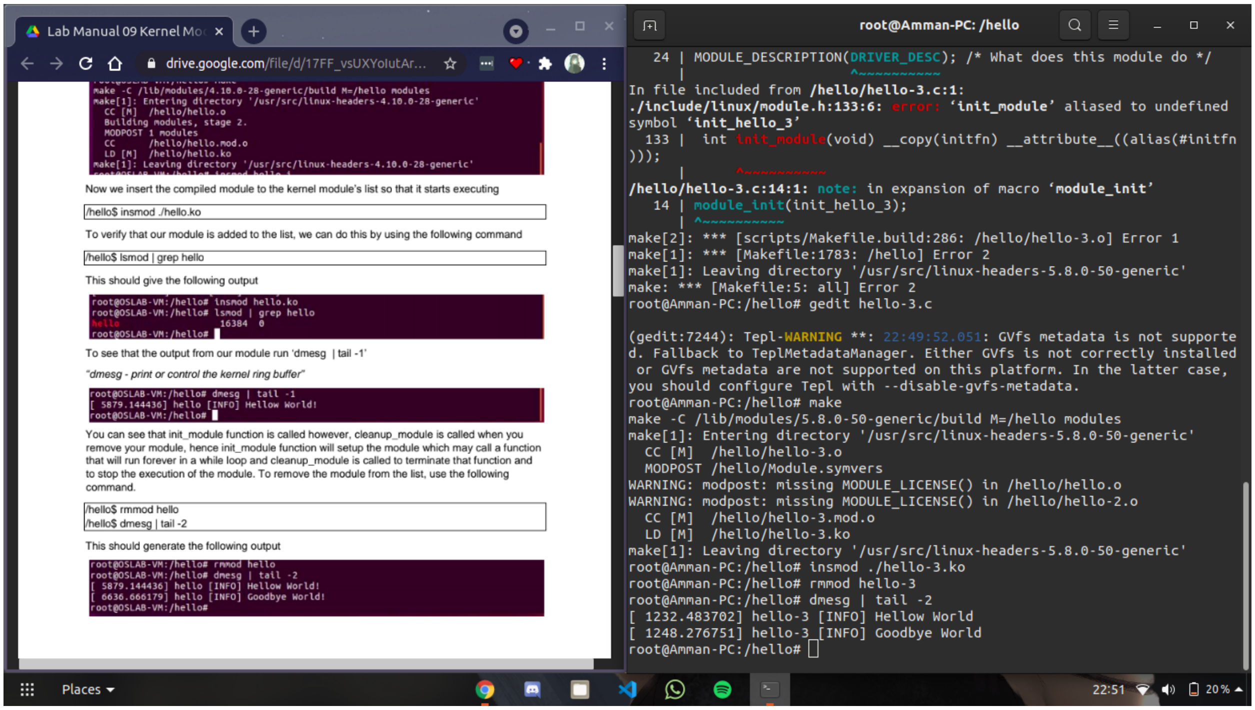
root@OSLAB-VM:/hello# rmmod hello
root@OSLAB-VM:/hello# dmesg | tail -2
[5879.144436] hello [INFO] Hellow World!
[6636.666179] hello [INFO] Goodbye World!
root@OSLAB-VM:/hello#

```
hello.ko  hello.mod.c  hello.o      modules.order
root@Amman-PC:/hello# touch hello-2.c
root@Amman-PC:/hello# gedit hello-2.c

(gedit:6274): Tepl-WARNING **: 22:44:08.752: GVfs metadata is not supporte
d. Fallback to TeplMetadataManager. Either GVfs is not correctly installed
 or GVfs metadata are not supported on this platform. In the latter case,
you should configure Tepl with --disable-gvfs-metadata.
root@Amman-PC:/hello# gedit Makefile

(gedit:6300): Tepl-WARNING **: 22:44:55.718: GVfs metadata is not supporte
d. Fallback to TeplMetadataManager. Either GVfs is not correctly installed
 or GVfs metadata are not supported on this platform. In the latter case,
you should configure Tepl with --disable-gvfs-metadata.
root@Amman-PC:/hello# make
make -C /lib/modules/5.8.0-50-generic/build M=/hello modules
make[1]: Entering directory '/usr/src/linux-headers-5.8.0-50-generic'
  CC [M]  /hello/hello-2.o
  MODPOST /hello/Module.symvers
WARNING: modpost: missing MODULE_LICENSE() in /hello/hello.o
WARNING: modpost: missing MODULE_LICENSE() in /hello/hello-2.o
  CC [M]  /hello/hello-2.mod.o
  LD [M]  /hello/hello-2.ko
make[1]: Leaving directory '/usr/src/linux-headers-5.8.0-50-generic'
root@Amman-PC:/hello# ls
hello-2.c    hello-2.mod.c  hello.c    hello.mod.c  Makefile
hello-2.ko   hello-2.mod.o  hello.ko   hello.mod.o  modules.order
hello-2.mod  hello-2.o      hello.mod  hello.o      Module.symvers
root@Amman-PC:/hello# insmod ./hello-2.ko
root@Amman-PC:/hello# lsmod | grep hello-2
root@Amman-PC:/hello# lsmod | grep hello
hello_2                16384  0
root@Amman-PC:/hello# rmmod hello-2
root@Amman-PC:/hello# dmesg | tail -2
[  959.176692] hello-2 [INFO] Hellow World
[ 1019.696730] hello-2 [INFO] Goodbye World
root@Amman-PC:/hello# 
```

WITH LICENSING AND DOCUMENTATION



```
make -C /lib/modules/4.10.0-28-generic/build M=/hello modules
make[1]: Entering directory '/usr/src/linux-headers-4.10.0-28-generic'
  CC [M]  /hello/hello.o
  Building modules, stage 2.
  MODPOST 1 modules
  CC      /hello/hello.mod.o
  LD [M]  /hello/hello.ko
make[1]: Leaving directory '/usr/src/linux-headers-4.10.0-28-generic'
```

Now we insert the compiled module to the kernel module's list so that it starts executing

```
/hello$ insmod ./hello.ko
```

To verify that our module is added to the list, we can do this by using the following command

```
/hello$ lsmod | grep hello
```

This should give the following output

```
root@OSLAB-VM:/hello# insmod hello.ko
root@OSLAB-VM:/hello# lsmod | grep hello
hello                  16384  0
root@OSLAB-VM:/hello#
```

To see that the output from our module run 'dmesg | tail -1'

"dmesg - print or control the kernel ring buffer"

```
root@OSLAB-VM:/hello# dmesg | tail -1
[ 5879.144436] hello [INFO] Hellow World!
root@OSLAB-VM:/hello#
```

You can see that init_module function is called however, cleanup_module is called when you remove your module, hence init_module function will setup the module which may call a function that will run forever in a while loop and cleanup_module is called to terminate that function and to stop the execution of the module. To remove the module from the list, use the following command.

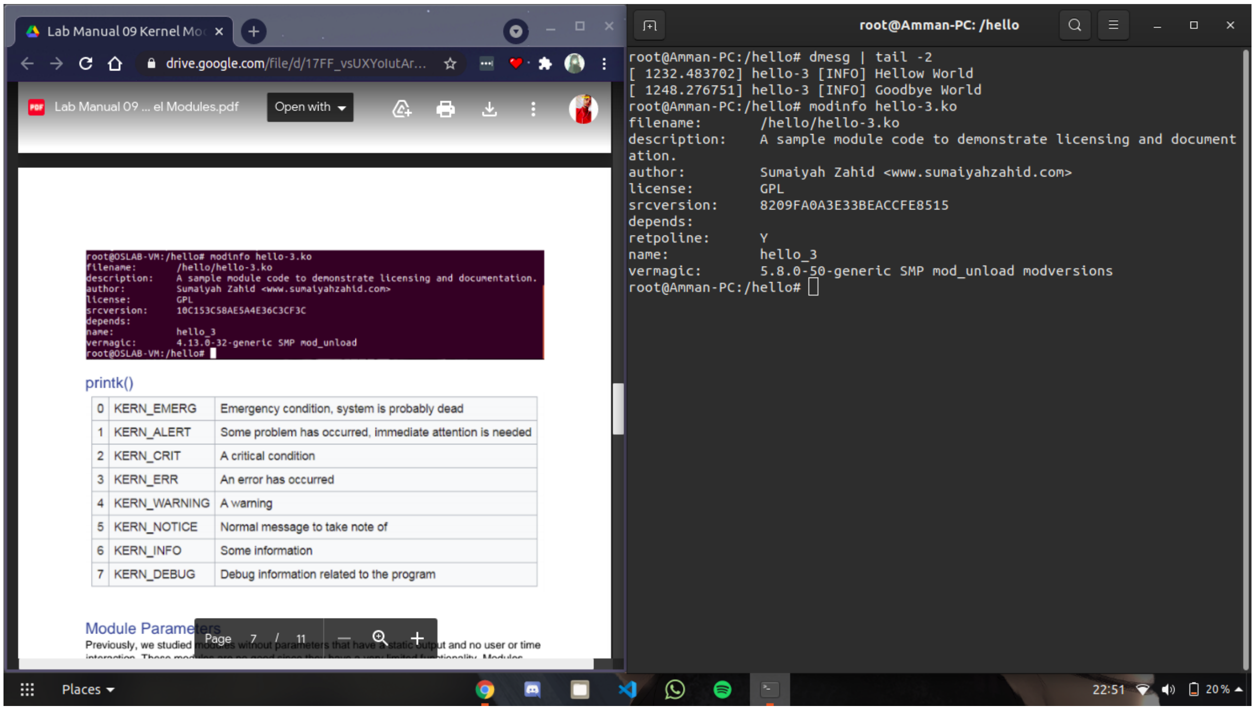
```
/hello$ rmmod hello
/hello$ dmesg | tail -2
```

This should generate the following output

```
root@OSLAB-VM:/hello# rmmod hello
root@OSLAB-VM:/hello# dmesg | tail -2
[ 5879.144436] hello [INFO] Hellow World!
[ 6636.666179] hello [INFO] Goodbye World!
root@OSLAB-VM:/hello#
```

```
   24 | MODULE_DESCRIPTION(DRIVER_DESC); /* What does this module do */
      |                    ^~~~~~~~~~~
In file included from /hello/hello-3.c:1:
./include/linux/module.h:133:6: error: ‘init_module’ aliased to undefined symbol ‘init_hello_3’
  133 |  int init_module(void) __copy(initfn) __attribute__((alias(#initfn
)));
      |      ^~~~~~~~~~~
/hello/hello-3.c:14:1: note: in expansion of macro ‘module_init’
   14 | module_init(init_hello_3);
      | ^~~~~~~~~~~
make[2]: *** [scripts/Makefile.build:286: /hello/hello-3.o] Error 1
make[1]: *** [Makefile:1783: /hello] Error 2
make[1]: Leaving directory '/usr/src/linux-headers-5.8.0-50-generic'
make: *** [Makefile:5: all] Error 2
root@Amman-PC:/hello# gedit hello-3.c

(gedit:7244): Tepl-WARNING **: 22:49:52.051: GVfs metadata is not supporte
d. Fallback to TeplMetadataManager. Either GVfs is not correctly installed
 or GVfs metadata are not supported on this platform. In the latter case,
you should configure Tepl with --disable-gvfs-metadata.
root@Amman-PC:/hello# make
make -C /lib/modules/5.8.0-50-generic/build M=/hello modules
make[1]: Entering directory '/usr/src/linux-headers-5.8.0-50-generic'
  CC [M]  /hello/hello-3.o
  MODPOST /hello/Module.symvers
WARNING: modpost: missing MODULE_LICENSE() in /hello/hello.o
WARNING: modpost: missing MODULE_LICENSE() in /hello/hello-2.o
  CC [M]  /hello/hello-3.mod.o
  LD [M]  /hello/hello-3.ko
make[1]: Leaving directory '/usr/src/linux-headers-5.8.0-50-generic'
root@Amman-PC:/hello# insmod ./hello-3.ko
root@Amman-PC:/hello# rmmod hello-3
root@Amman-PC:/hello# dmesg | tail -2
[ 1232.483702] hello-3 [INFO] Hellow World
[ 1248.276751] hello-3 [INFO] Goodbye World
root@Amman-PC:/hello#
```



```
root@OSLAB-VM:/hello# modinfo hello-3.ko
filename:       /hello/hello-3.ko
description:    A sample module code to demonstrate licensing and documentation.
author:         Sumaiyah Zahid <www.sumaiyahzahid.com>
license:        GPL
srcversion:     10C153C58AE5A4E36C3CF3C
depends:
name:           hello_3
vermagic:       4.13.0-32-generic SMP mod_unload
root@OSLAB-VM:/hello#
```

printk()

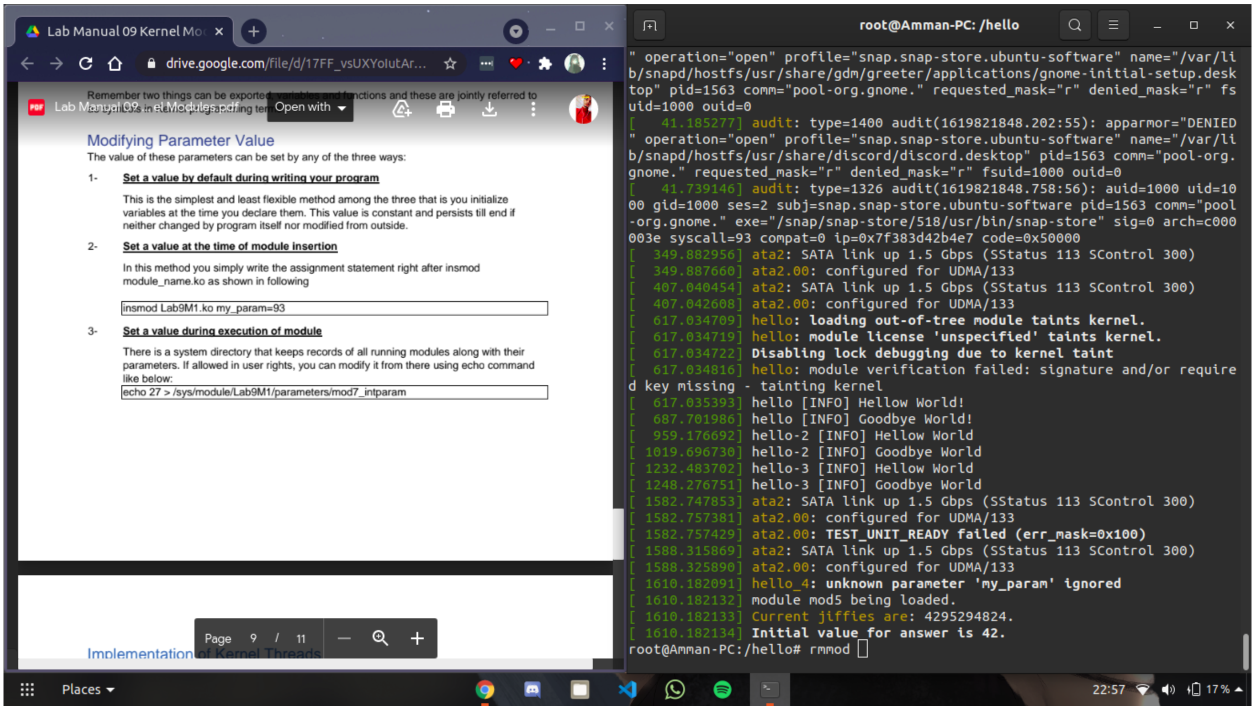
0	KERN_EMERG	Emergency condition, system is probably dead
1	KERN_ALERT	Some problem has occurred, immediate attention is needed
2	KERN_CRIT	A critical condition
3	KERN_ERR	An error has occurred
4	KERN_WARNING	A warning
5	KERN_NOTICE	Normal message to take note of
6	KERN_INFO	Some information
7	KERN_DEBUG	Debug information related to the program

Module Parameters

Previously, we studied modules without parameters that have a static output and no user or time

```
root@Amman-PC:/hello# dmesg | tail -2
[ 1232.483702] hello-3 [INFO] Hellow World
[ 1248.276751] hello-3 [INFO] Goodbye World
root@Amman-PC:/hello# modinfo hello-3.ko
filename:       /hello/hello-3.ko
description:    A sample module code to demonstrate licensing and document
ation.
author:         Sumaiyah Zahid <www.sumaiyahzahid.com>
license:        GPL
srcversion:     8209FA0A3E33BEACCFE8515
depends:
retpoline:      Y
name:           hello_3
vermagic:       5.8.0-50-generic SMP mod_unload modversions
root@Amman-PC:/hello#
```

MODULE PARAMETERS



Lab Manual 09 Kernel Mo
drive.google.com/file/d/17FF_vsUXYolutAr...
Lab Manual 09 ... el Modules.pdf
Open with

Remember two things can be exported, variables and functions and these are jointly referred to

Modifying Parameter Value

The value of these parameters can be set by any of the three ways:

1- **Set a value by default during writing your program**

This is the simplest and least flexible method among the three that is you initialize variables at the time you declare them. This value is constant and persists till end if neither changed by program itself nor modified from outside.

2- **Set a value at the time of module insertion**

In this method you simply write the assignment statement right after insmod module_name.ko as shown in following

```
insmod Lab9M1.ko my_param=93
```

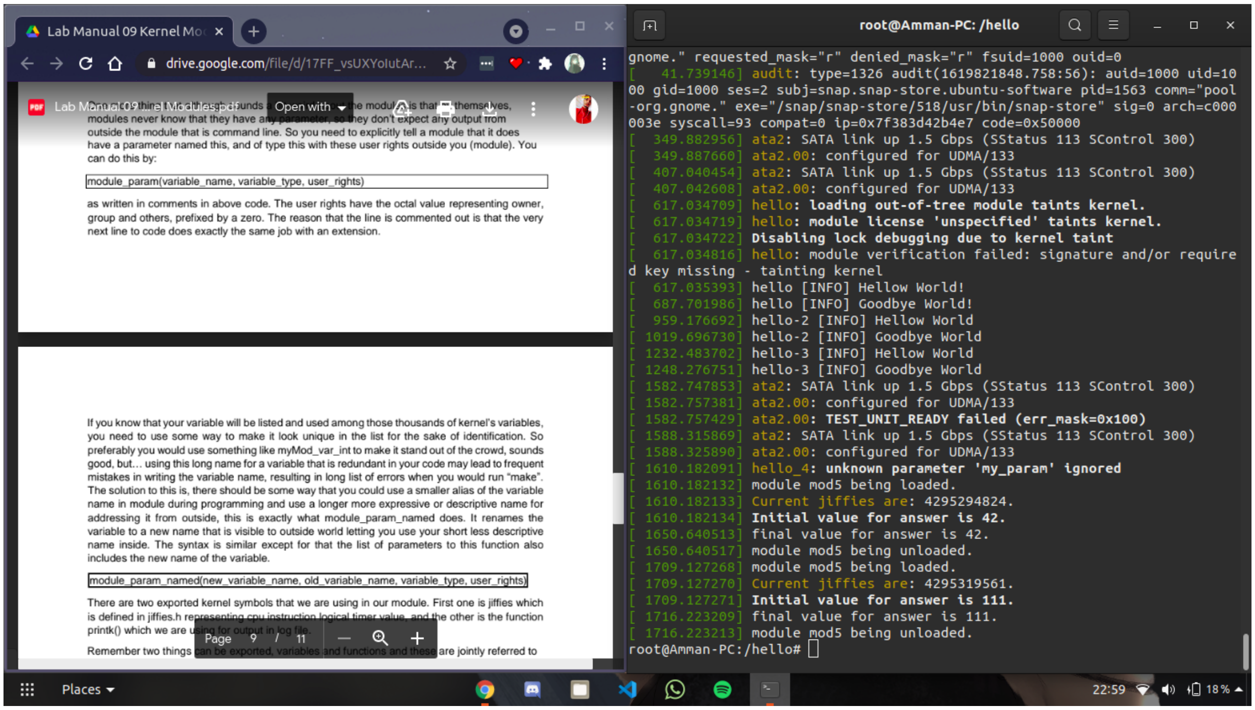
3- **Set a value during execution of module**

There is a system directory that keeps records of all running modules along with their parameters. If allowed in user rights, you can modify it from there using echo command like below:

```
echo 27 > /sys/module/Lab9M1/parameters/mod7_intparam
```

Implementation of Kernel Threads

```
" operation="open" profile="snap.snap-store.ubuntu-software" name="/var/li
b/snapd/hostfs/usr/share/gdm/greeter/applications/gnome-initial-setup.desk
top" pid=1563 comm="pool-org.gnome." requested_mask="r" denied_mask="r" fs
uid=1000 ouid=0
[   41.185277] audit: type=1400 audit(1619821848.202:55): apparmor="DENIED
" operation="open" profile="snap.snap-store.ubuntu-software" name="/var/li
b/snapd/hostfs/usr/share/discord/discord.desktop" pid=1563 comm="pool-org.
gnome." requested_mask="r" denied_mask="r" fsuid=1000 ouid=0
[   41.739146] audit: type=1326 audit(1619821848.758:56): auid=1000 uid=10
00 gid=1000 ses=2 subj=snap.snap-store.ubuntu-software pid=1563 comm="pool
-org.gnome." exe="/snap/snap-store/518/usr/bin/snap-store" sig=0 arch=c000
003e syscall=93 compat=0 ip=0x7f383d42b4e7 code=0x50000
[  349.882956] ata2: SATA link up 1.5 Gbps (SStatus 113 SControl 300)
[  349.887660] ata2.00: configured for UDMA/133
[  407.040454] ata2: SATA link up 1.5 Gbps (SStatus 113 SControl 300)
[  407.042608] ata2.00: configured for UDMA/133
[  617.034709] hello: loading out-of-tree module taints kernel.
[  617.034719] hello: module license 'unspecified' taints kernel.
[  617.034722] Disabling lock debugging due to kernel taint
[  617.034816] hello: module verification failed: signature and/or require
d key missing - tainting kernel
[  617.035393] hello [INFO] Hellow World!
[  687.701986] hello [INFO] Goodbye World!
[  959.176692] hello-2 [INFO] Hellow World
[ 1019.696730] hello-2 [INFO] Goodbye World
[ 1232.483702] hello-3 [INFO] Hellow World
[ 1248.276751] hello-3 [INFO] Goodbye World
[ 1582.747853] ata2: SATA link up 1.5 Gbps (SStatus 113 SControl 300)
[ 1582.757381] ata2.00: configured for UDMA/133
[ 1582.757429] ata2.00: TEST_UNIT_READY failed (err_mask=0x100)
[ 1588.315869] ata2: SATA link up 1.5 Gbps (SStatus 113 SControl 300)
[ 1588.325890] ata2.00: configured for UDMA/133
[ 1610.182091] hello_4: unknown parameter 'my_param' ignored
[ 1610.182132] module mod5 being loaded.
[ 1610.182133] Current jiffies are: 4295294824.
[ 1610.182134] Initial value for answer is 42.
root@Amman-PC:/hello# rmmod
```



Lab Manual 09 Kernel Mo
drive.google.com/file/d/17FF_vsUXYolutAr...
Lab Manual 09 ... el Modules.pdf
Open with

modules never know that they have any parameter, so they don't expect any output from outside the module that is command line. So you need to explicitly tell a module that it does have a parameter named this, and of type this with these user rights outside you (module). You can do this by:

```
module_param(variable_name, variable_type, user_rights)
```

as written in comments in above code. The user rights have the octal value representing owner, group and others, prefixed by a zero. The reason that the line is commented out is that the very next line to code does exactly the same job with an extension.

If you know that your variable will be listed and used among those thousands of kernel's variables, you need to use some way to make it look unique in the list for the sake of identification. So preferably you would use something like myMod_var_int to make it stand out of the crowd, sounds good, but… using this long name for a variable that is redundant in your code may lead to frequent mistakes in writing the variable name, resulting in long list of errors when you would run "make". The solution to this is, there should be some way that you could use a smaller alias of the variable name in module during programming and use a longer more expressive or descriptive name for addressing it from outside, this is exactly what module_param_named does. It renames the variable to a new name that is visible to outside world letting you use your short less descriptive name inside. The syntax is similar except for that the list of parameters to this function also includes the new name of the variable.

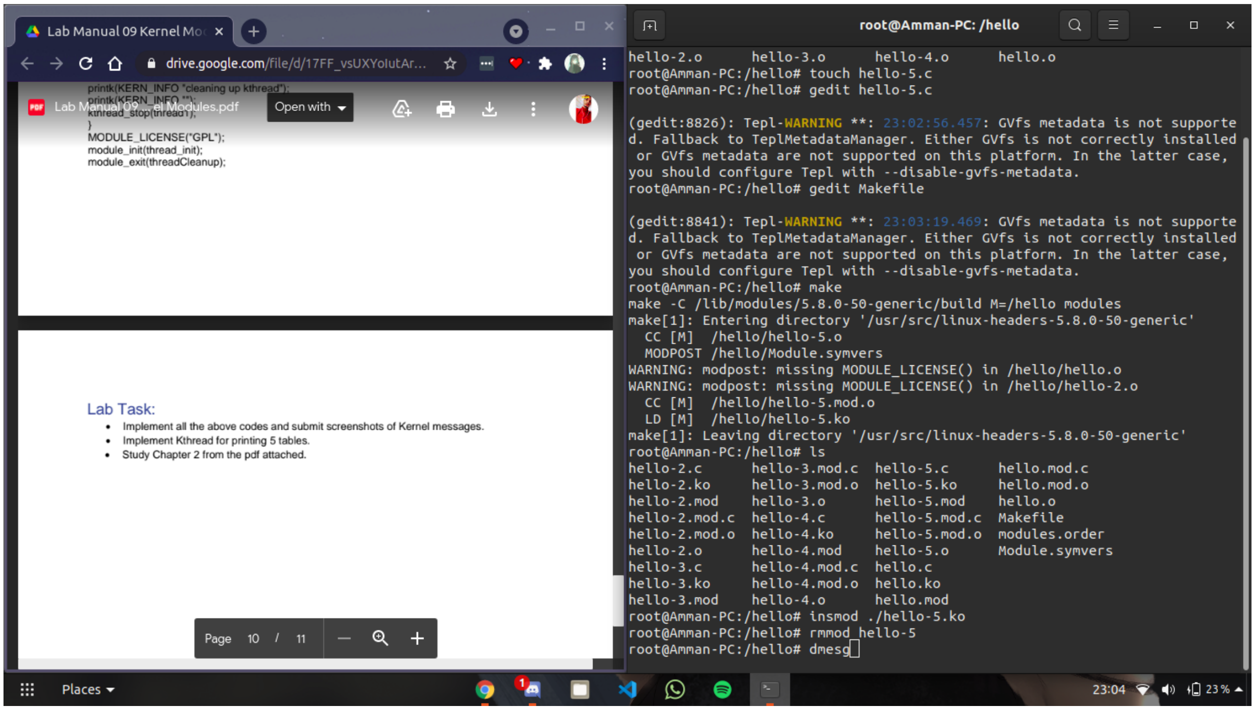
```
module_param_named(new_variable_name, old_variable_name, variable_type, user_rights)
```

There are two exported kernel symbols that we are using in our module. First one is jiffies which is defined in jiffies.h representing cpu instruction logical timer value, and the other is the function printk() which we are using for output in log file.

Remember two things can be exported, variables and functions and these are jointly referred to

```
gnome." requested_mask="r" denied_mask="r" fsuid=1000 ouid=0
[   41.739146] audit: type=1326 audit(1619821848.758:56): auid=1000 uid=10
00 gid=1000 ses=2 subj=snap.snap-store.ubuntu-software pid=1563 comm="pool
-org.gnome." exe="/snap/snap-store/518/usr/bin/snap-store" sig=0 arch=c000
003e syscall=93 compat=0 ip=0x7f383d42b4e7 code=0x50000
[  349.882956] ata2: SATA link up 1.5 Gbps (SStatus 113 SControl 300)
[  349.887660] ata2.00: configured for UDMA/133
[  407.040454] ata2: SATA link up 1.5 Gbps (SStatus 113 SControl 300)
[  407.042608] ata2.00: configured for UDMA/133
[  617.034709] hello: loading out-of-tree module taints kernel.
[  617.034719] hello: module license 'unspecified' taints kernel.
[  617.034722] Disabling lock debugging due to kernel taint
[  617.034816] hello: module verification failed: signature and/or require
d key missing - tainting kernel
[  617.035393] hello [INFO] Hellow World!
[  687.701986] hello [INFO] Goodbye World!
[  959.176692] hello-2 [INFO] Hellow World
[ 1019.696730] hello-2 [INFO] Goodbye World
[ 1232.483702] hello-3 [INFO] Hellow World
[ 1248.276751] hello-3 [INFO] Goodbye World
[ 1582.747853] ata2: SATA link up 1.5 Gbps (SStatus 113 SControl 300)
[ 1582.757381] ata2.00: configured for UDMA/133
[ 1582.757429] ata2.00: TEST_UNIT_READY failed (err_mask=0x100)
[ 1588.315869] ata2: SATA link up 1.5 Gbps (SStatus 113 SControl 300)
[ 1588.325890] ata2.00: configured for UDMA/133
[ 1610.182091] hello_4: unknown parameter 'my_param' ignored
[ 1610.182132] module mod5 being loaded.
[ 1610.182133] Current jiffies are: 4295294824.
[ 1610.182134] Initial value for answer is 42.
[ 1650.640513] final value for answer is 42.
[ 1650.640517] module mod5 being unloaded.
[ 1709.127268] module mod5 being loaded.
[ 1709.127270] Current jiffies are: 4295319561.
[ 1709.127271] Initial value for answer is 111.
[ 1716.223209] final value for answer is 111.
[ 1716.223213] module mod5 being unloaded.
root@Amman-PC:/hello#
```

IMPLEMENTATION OF KTHREADS



Lab Manual 09 Kernel Mo
drive.google.com/file/d/17FF_vsUXYolutAr...
Lab Manual 09 ... el Modules.pdf
Open with
Page 10 / 11
Places
23:04
23 %

```
printk(KERN_INFO "cleaning up kthread");
printk(KERN_INFO "");
kthread_stop(thread1);
}
MODULE_LICENSE("GPL");
module_init(thread_init);
module_exit(threadCleanup);
```

Lab Task:

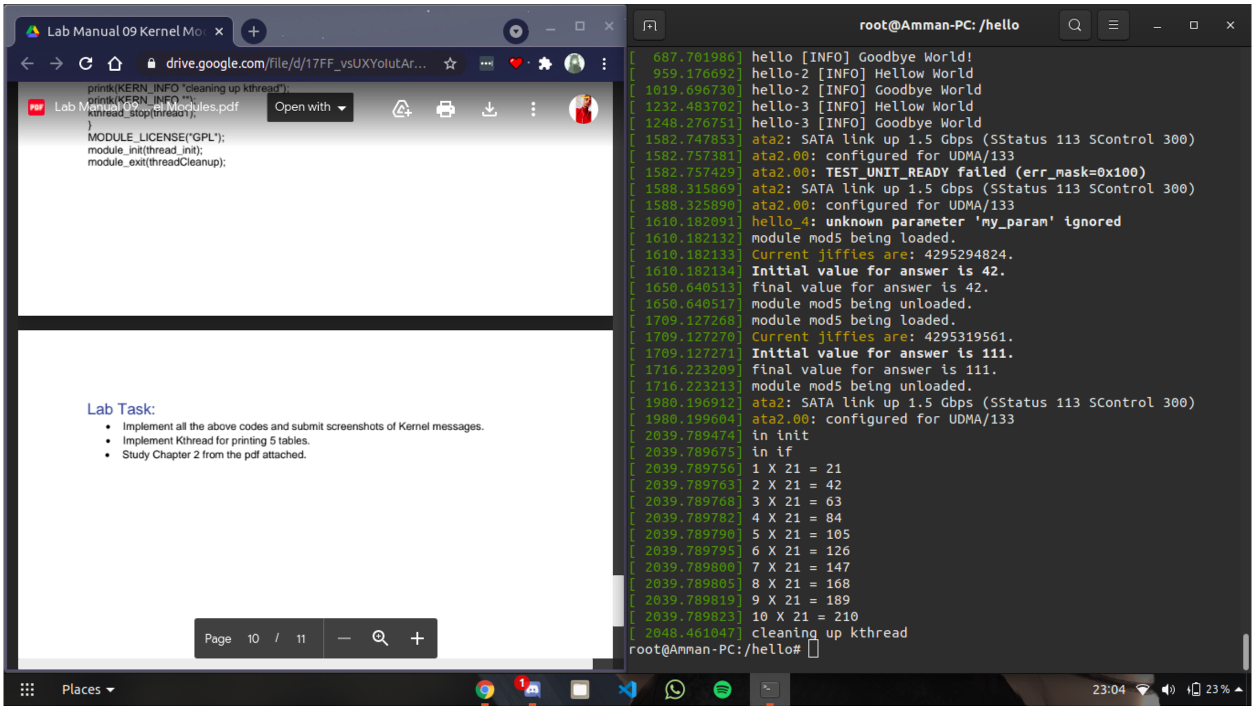
- Implement all the above codes and submit screenshots of Kernel messages.
- Implement Kthread for printing 5 tables.
- Study Chapter 2 from the pdf attached.

root@Amman-PC: /hello

```
hello-2.o       hello-3.o       hello-4.o       hello.o
root@Amman-PC:/hello# touch hello-5.c
root@Amman-PC:/hello# gedit hello-5.c

(gedit:8826): Tepl-WARNING **: 23:02:56.457: GVfs metadata is not supporte
d. Fallback to TeplMetadataManager. Either GVfs is not correctly installed
 or GVfs metadata are not supported on this platform. In the latter case,
you should configure Tepl with --disable-gvfs-metadata.
root@Amman-PC:/hello# gedit Makefile

(gedit:8841): Tepl-WARNING **: 23:03:19.469: GVfs metadata is not supporte
d. Fallback to TeplMetadataManager. Either GVfs is not correctly installed
 or GVfs metadata are not supported on this platform. In the latter case,
you should configure Tepl with --disable-gvfs-metadata.
root@Amman-PC:/hello# make
make -C /lib/modules/5.8.0-50-generic/build M=/hello modules
make[1]: Entering directory '/usr/src/linux-headers-5.8.0-50-generic'
  CC [M]  /hello/hello-5.o
  MODPOST /hello/Module.symvers
WARNING: modpost: missing MODULE_LICENSE() in /hello/hello.o
WARNING: modpost: missing MODULE_LICENSE() in /hello/hello-2.o
  CC [M]  /hello/hello-5.mod.o
  LD [M]  /hello/hello-5.ko
make[1]: Leaving directory '/usr/src/linux-headers-5.8.0-50-generic'
root@Amman-PC:/hello# ls
hello-2.c       hello-3.mod.c   hello-5.c       hello.mod.c
hello-2.ko      hello-3.mod.o   hello-5.ko      hello.mod.o
hello-2.mod     hello-3.o       hello-5.mod     hello.o
hello-2.mod.c   hello-4.c       hello-5.mod.c   Makefile
hello-2.mod.o   hello-4.ko      hello-5.mod.o   modules.order
hello-2.o       hello-4.mod     hello-5.o       Module.symvers
hello-3.c       hello-4.mod.c   hello.c
hello-3.ko      hello-4.mod.o   hello.ko
hello-3.mod     hello-4.o       hello.mod
root@Amman-PC:/hello# insmod ./hello-5.ko
root@Amman-PC:/hello# rmmod hello-5
root@Amman-PC:/hello# dmesg
```



Lab Manual 09 Kernel Mo
drive.google.com/file/d/17FF_vsUXYolutAr...
Lab Manual 09 ... el Modules.pdf
Open with
Page 10 / 11
Places
23:04
23 %

printk(KERN_INFO "cleaning up kthread");
printk(KERN_INFO "");
kthread_stop(thread1);
}
MODULE_LICENSE("GPL");
module_init(thread_init);
module_exit(threadCleanup);

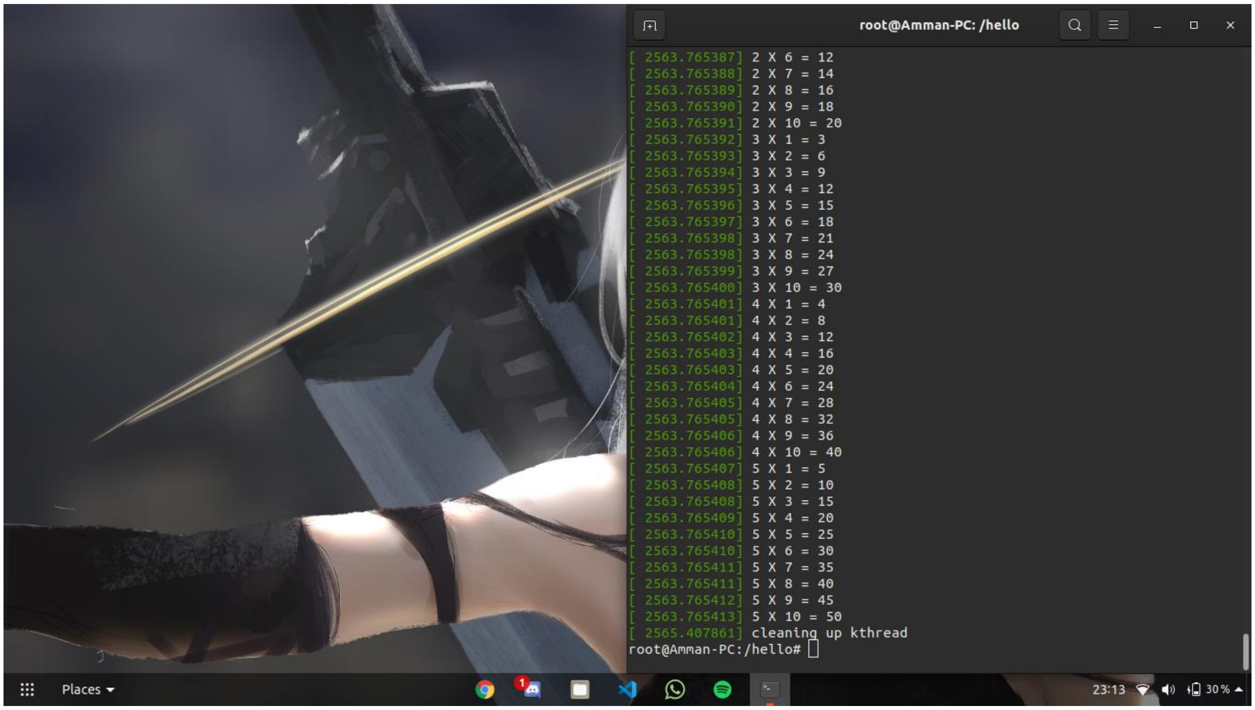
Lab Task:

- Implement all the above codes and submit screenshots of Kernel messages.
- Implement Kthread for printing 5 tables.
- Study Chapter 2 from the pdf attached.

root@Amman-PC: /hello

```
[  687.701986] hello [INFO] Goodbye World!
[  959.176692] hello-2 [INFO] Hellow World
[ 1019.696730] hello-2 [INFO] Goodbye World
[ 1232.483702] hello-3 [INFO] Hellow World
[ 1248.276751] hello-3 [INFO] Goodbye World
[ 1582.747853] ata2: SATA link up 1.5 Gbps (SStatus 113 SControl 300)
[ 1582.757381] ata2.00: configured for UDMA/133
[ 1582.757429] ata2.00: TEST_UNIT_READY failed (err_mask=0x100)
[ 1588.315869] ata2: SATA link up 1.5 Gbps (SStatus 113 SControl 300)
[ 1588.325890] ata2.00: configured for UDMA/133
[ 1610.182091] hello_4: unknown parameter 'my_param' ignored
[ 1610.182132] module mod5 being loaded.
[ 1610.182133] Current jiffies are: 4295294824.
[ 1610.182134] Initial value for answer is 42.
[ 1650.640513] final value for answer is 42.
[ 1650.640517] module mod5 being unloaded.
[ 1709.127268] module mod5 being loaded.
[ 1709.127270] Current jiffies are: 4295319561.
[ 1709.127271] Initial value for answer is 111.
[ 1716.223209] final value for answer is 111.
[ 1716.223213] module mod5 being unloaded.
[ 1980.196912] ata2: SATA link up 1.5 Gbps (SStatus 113 SControl 300)
[ 1980.199604] ata2.00: configured for UDMA/133
[ 2039.789474] in init
[ 2039.789675] in if
[ 2039.789756] 1 X 21 = 21
[ 2039.789763] 2 X 21 = 42
[ 2039.789768] 3 X 21 = 63
[ 2039.789782] 4 X 21 = 84
[ 2039.789790] 5 X 21 = 105
[ 2039.789795] 6 X 21 = 126
[ 2039.789800] 7 X 21 = 147
[ 2039.789805] 8 X 21 = 168
[ 2039.789819] 9 X 21 = 189
[ 2039.789823] 10 X 21 = 210
[ 2048.461047] cleaning up kthread
root@Amman-PC:/hello# 
```

QUESTION NO 2 – PRINTING 5 TABLES



```
[  2563.765387] 2 X 6 = 12
[  2563.765388] 2 X 7 = 14
[  2563.765389] 2 X 8 = 16
[  2563.765390] 2 X 9 = 18
[  2563.765391] 2 X 10 = 20
[  2563.765392] 3 X 1 = 3
[  2563.765393] 3 X 2 = 6
[  2563.765394] 3 X 3 = 9
[  2563.765395] 3 X 4 = 12
[  2563.765396] 3 X 5 = 15
[  2563.765397] 3 X 6 = 18
[  2563.765398] 3 X 7 = 21
[  2563.765398] 3 X 8 = 24
[  2563.765399] 3 X 9 = 27
[  2563.765400] 3 X 10 = 30
[  2563.765401] 4 X 1 = 4
[  2563.765401] 4 X 2 = 8
[  2563.765402] 4 X 3 = 12
[  2563.765403] 4 X 4 = 16
[  2563.765403] 4 X 5 = 20
[  2563.765404] 4 X 6 = 24
[  2563.765405] 4 X 7 = 28
[  2563.765405] 4 X 8 = 32
[  2563.765406] 4 X 9 = 36
[  2563.765406] 4 X 10 = 40
[  2563.765407] 5 X 1 = 5
[  2563.765408] 5 X 2 = 10
[  2563.765408] 5 X 3 = 15
[  2563.765409] 5 X 4 = 20
[  2563.765410] 5 X 5 = 25
[  2563.765410] 5 X 6 = 30
[  2563.765411] 5 X 7 = 35
[  2563.765411] 5 X 8 = 40
[  2563.765412] 5 X 9 = 45
[  2563.765413] 5 X 10 = 50
[  2565.407861] cleaning up kthread
root@Amman-PC:/hello# 
```

CODE

```
#include<linux/module.h>

#include<linux/kernel.h>

#include<linux/kthread.h>

#include<linux/sched.h>

#include<linux/time.h>

#include<linux/timer.h>

static struct task_struct * thread1;

int a = 1;

int threadFnc(void * t) {

    int i = 1;
    int j = 1;
   // int x = * (int * ) t;
    for (i = 1; i <= 5; i++) {
                    for(j = 1; j<= 10; j++)
                    {
        printk(KERN_INFO "%d X %d = %d\n", i, j, i * j);
        printk(KERN_INFO "");
        }
    }
    set_current_state(TASK_INTERRUPTIBLE);
    while (!kthread_should_stop()) {
        schedule();
        set_current_state(TASK_INTERRUPTIBLE);
    }
    return 0;
}
int thread_init(void) {
    char our_thread[8] = "thread1";
    printk(KERN_INFO "in init");
    thread1 = kthread_create(threadFnc, & a, our_thread);
    if ((thread1)) {
        printk(KERN_INFO "in if");
        wake_up_process(thread1);

    }

    return 0;

}

void threadCleanup(void) {

    printk(KERN_INFO "cleaning up kthread");

    printk(KERN_INFO "");

    kthread_stop(thread1);

}

MODULE_LICENSE("GPL");

module_init(thread_init);

module_exit(threadCleanup);
```