운영체제

assignment1

김태석 교수님

2019202103

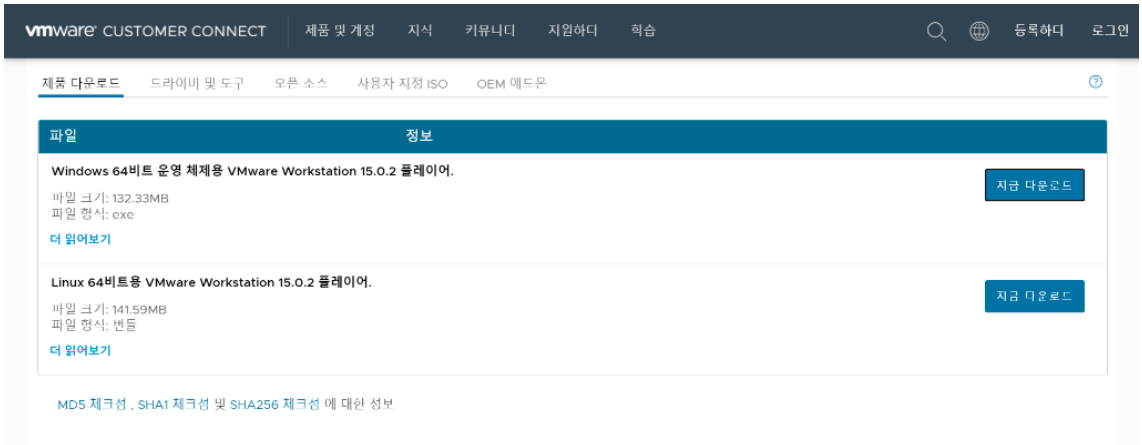
이은비

Assignment #1-1

<Introduction>

앞으로의 수업의 실습 및 과제를 위한 linux운영체제를 이용하기 위해 우분투와 VMWARE를 통해 리눅스를 설치하고 그에 대한 과정을 캡쳐한 사진과 짧은 설명을 보고서에 작성합니다. 또한 설치과정에서 os학번으로 설정하고 이후에 사진을 첨부 합니다.

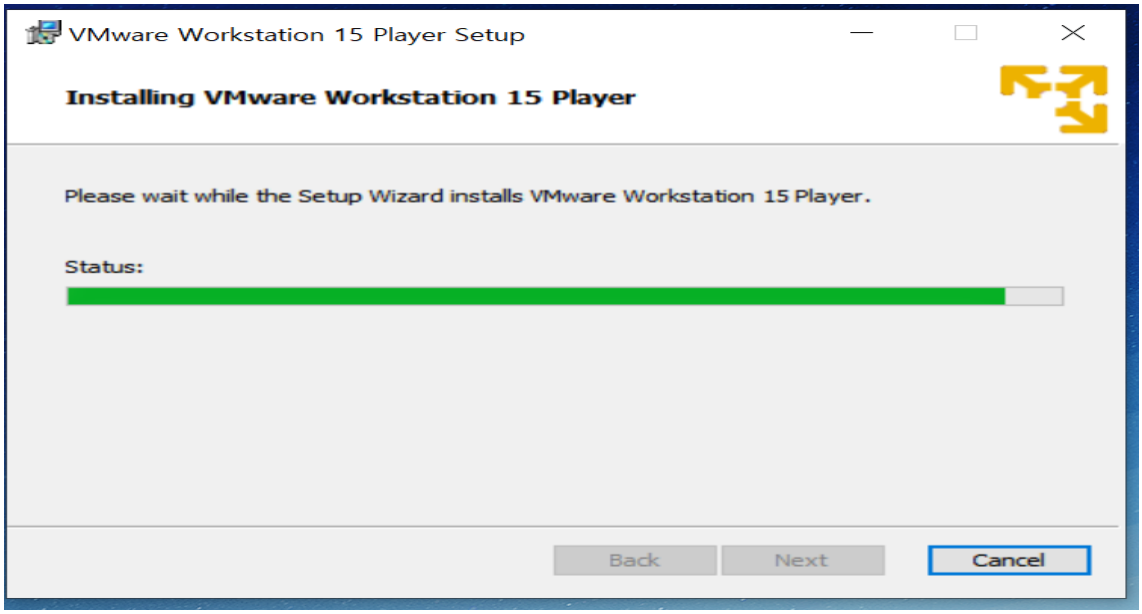
<Conclusion>

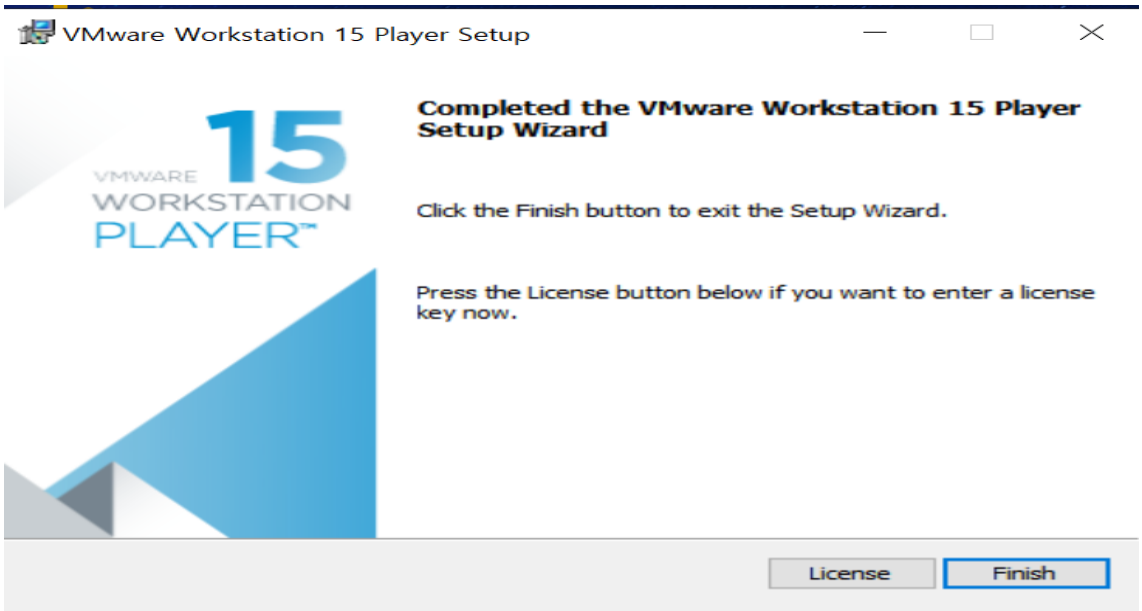


실습과제 강의자료 중 VMWARE설치 URL을 통해 들어간 화면에서 현재 쓰고있는 노트북의 운영체제에 맞춰 Window64에 해당하는 vmware 플레이어를 설치합니다.

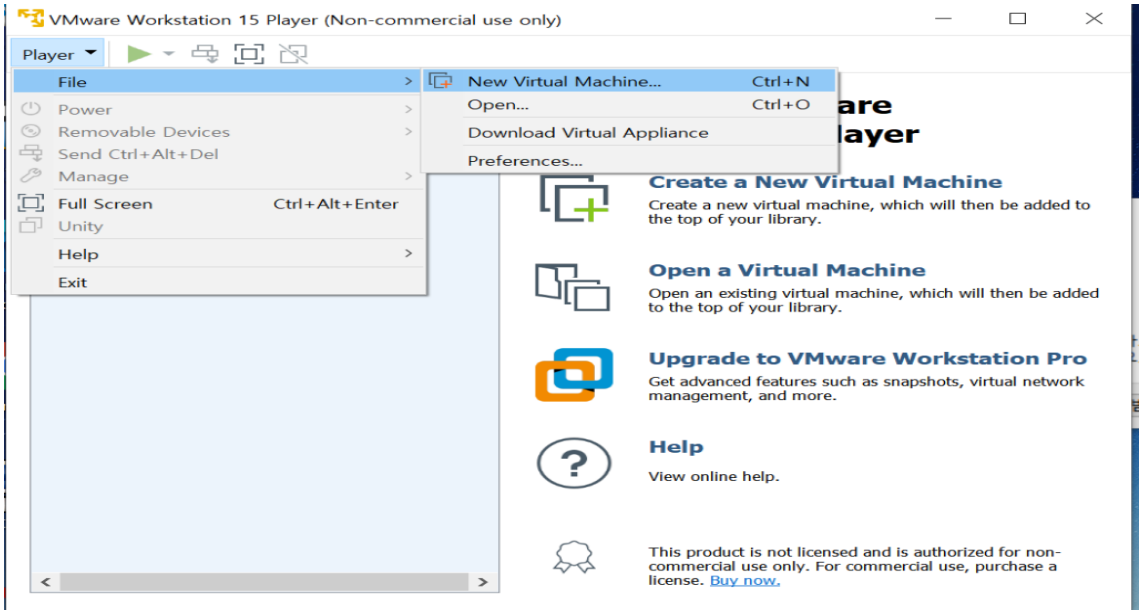
우분투는 자료의 URL을 클릭하면 바로 다운로드가 가능합니다.

두개가 모두 다운로드가 되어있는것을 확인한후 VMware-player를 실행합니다.

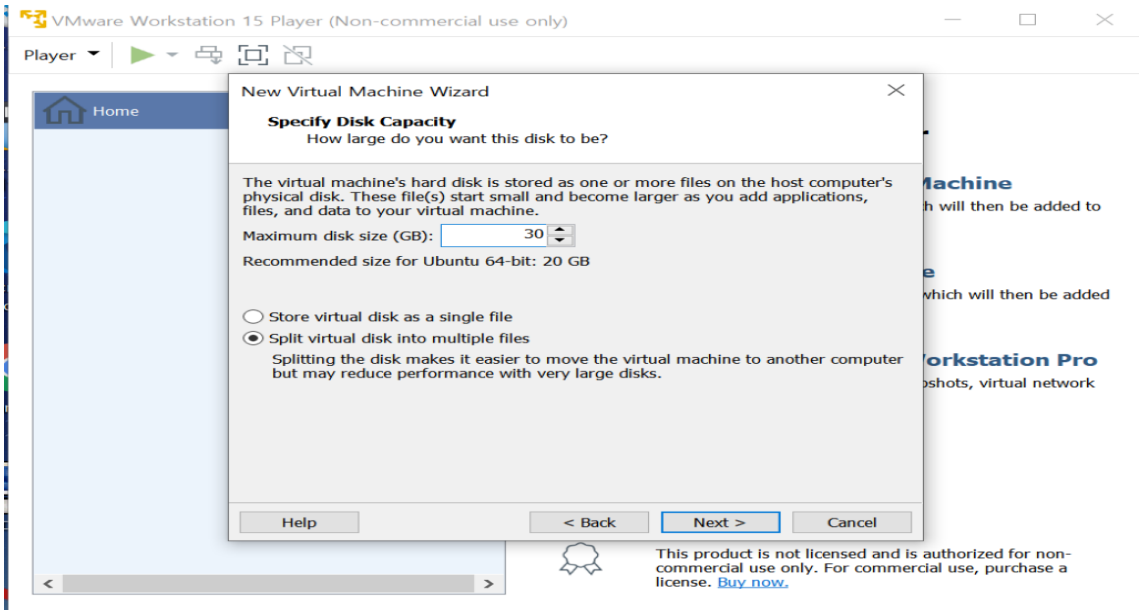




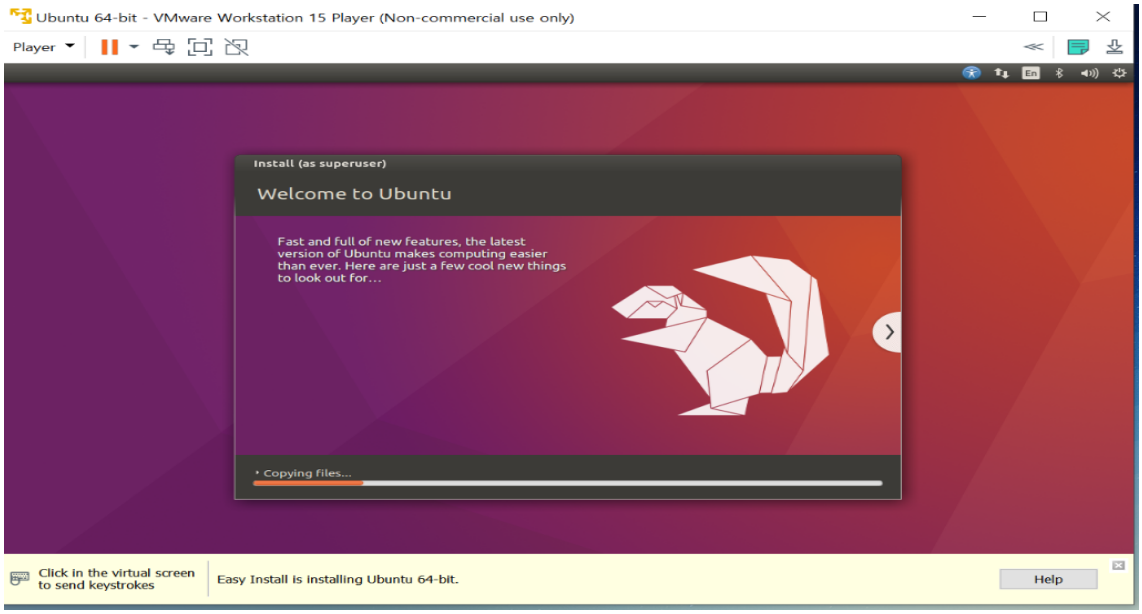
기본으로 체크되어 있는 옵션을 유지한 채 설치를 계속해서 진행하고 완료되었을때 Finish버튼을 눌러 마무리합니다.



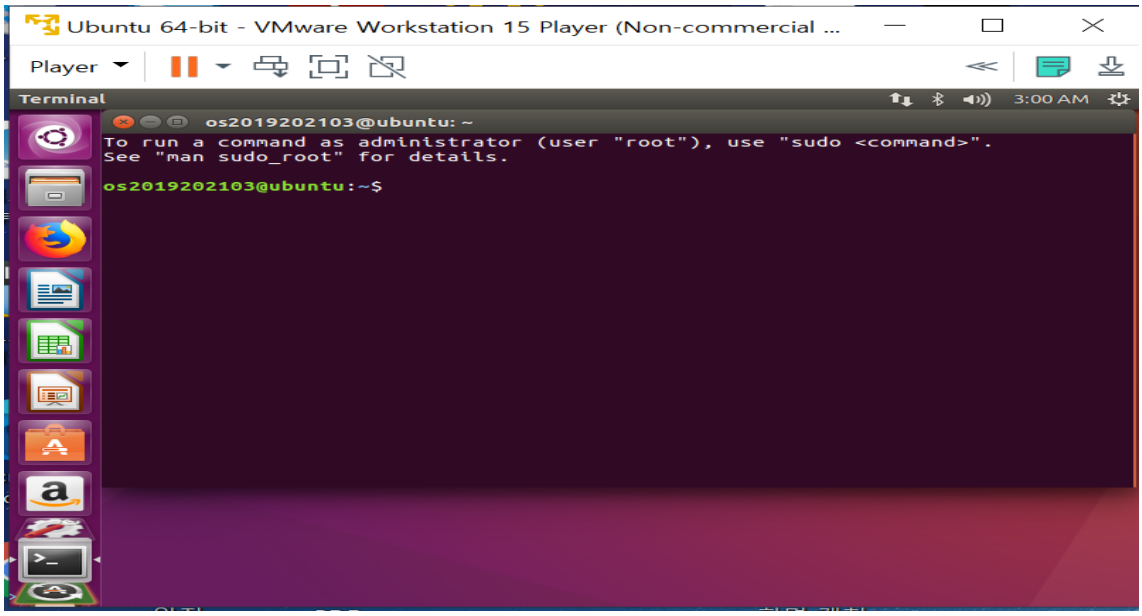
이후 설치된 vmware를 실행한 후 player->file->New Visual Machine을 클릭합니다.
이후 installer disc image file(iso):
밑에 설치된 ubuntu 파일을 browse하여 찾은 후 next합니다.
계정이름(User name)을 os2019202103으로 설정, 그외의 항목도 채우고 next합니다.



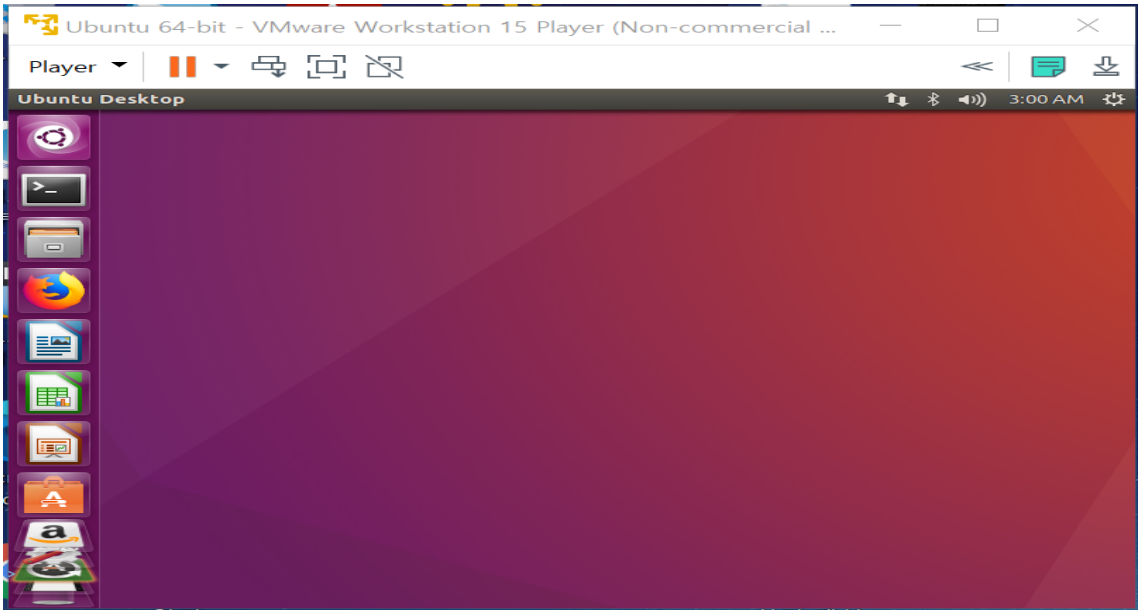
disksize는 30으로 설정하고 memory도 컴퓨터 사양에 맞게 설정합니다.



설치과 완료되면 계정이름을 설정했을때 설정했던 password를 치고 들어옵니다.
이후 terminal로 들어옵니다.



terminal에서 os2019202103@ubuntu:~$ 에 계정이름이 올바르게 되어있는지 확인합니다.



이후 dock에 terminal을 고정합니다.

<Reference>

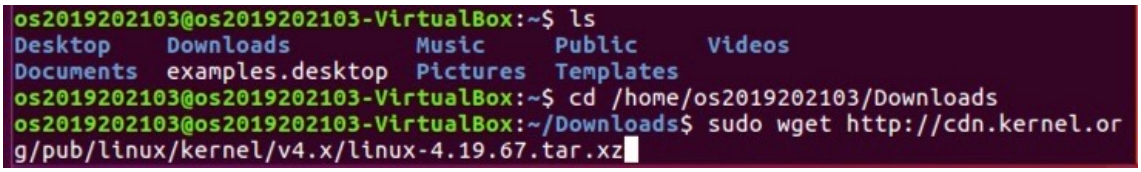
강의자료만 참고하였습니다.

Assignment #1-2

<Introduction>

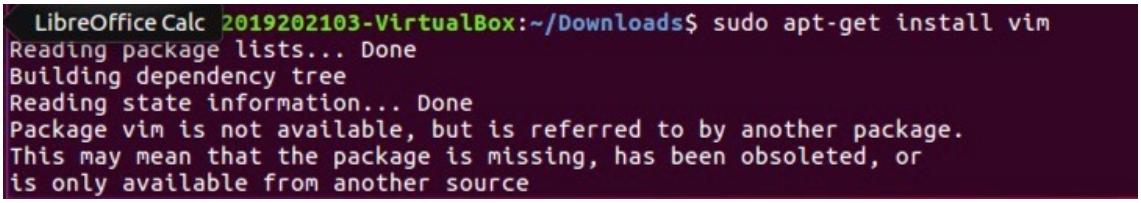
우분투 내에서 Kernel(4.19.67ver)을 다운로드 하고, 컴파일의 모든 과정의 terminal에 사용한 명령어와 vi의 수정상황등을 캡쳐하고 각각의 명령어가 어떠한 기능을 하였는지 설명을 덧붙여 보고서에 작성합니다. 또한 이후의 "uname-r"의 결과가 "4.19.67-본인학번(2019202103)"이 나오도록 하여 그 결과 또한 캡쳐하여 보고서에 추가합니다.

<Conclusion>



$ls하여 전체 디렉터리와 파일을 확인한후 그 안에 Downloads로 $cd /home/계정이름/Downloads(디렉터리)로 이동합니다.

이후 $sudo wget linux-4.19.67.tar.xz를 다운로드 하는 주소를 입력하여 다운로드 합니다.



$sudo apt-get install vim를 이용하여서 vi/vim에 대한것들을 설치하여 편집등을 할 수 있게 합니다.

```
os2019202103@os2019202103-VirtualBox: ~/Downloads
linux-4.19.67/drivers/scsi/hisi_sas/hisi_sas_v2_hw.c
linux-4.19.67/drivers/scsi/hisi_sas/hisi_sas_v3_hw.c
linux-4.19.67/drivers/scsi/hosts.c
linux-4.19.67/drivers/scsi/hpsa.c
linux-4.19.67/drivers/scsi/hpsa.h
linux-4.19.67/drivers/scsi/hpsa_cmd.h
linux-4.19.67/drivers/scsi/hptiop.c
linux-4.19.67/drivers/scsi/hptiop.h
linux-4.19.67/drivers/scsi/ibmvscsi/
linux-4.19.67/drivers/scsi/ibmvscsi/Makefile
linux-4.19.67/drivers/scsi/ibmvscsi/ibmvfc.c
linux-4.19.67/drivers/scsi/ibmvscsi/ibmvfc.h
linux-4.19.67/drivers/scsi/ibmvscsi/ibmvscsi.c
linux-4.19.67/drivers/scsi/ibmvscsi/ibmvscsi.h
linux-4.19.67/drivers/scsi/ibmvscsi_tgt/
linux-4.19.67/drivers/scsi/ibmvscsi_tgt/Makefile
linux-4.19.67/drivers/scsi/ibmvscsi_tgt/ibmvscsi_tgt.c
linux-4.19.67/drivers/scsi/ibmvscsi_tgt/ibmvscsi_tgt.h
linux-4.19.67/drivers/scsi/ibmvscsi_tgt/libsrp.c
linux-4.19.67/drivers/scsi/ibmvscsi_tgt/libsrp.h
linux-4.19.67/drivers/scsi/imm.c
linux-4.19.67/drivers/scsi/imm.h
linux-4.19.67/drivers/scsi/initio.c
```

$tar -jxvf linux-4.19.67.tar.xz가 error가 나서 $tar -xvf linux-4.19.67.tar.xz로 하여서 다운받은 kernel의 tar.xz로 압축된 kernelsource를 압축해제 합니다.그러면 이후 압축해제 되어 linux-4.19.67이라는 디렉터리가 생성됩니다.

```
os2019202103@os2019202103-VirtualBox:~/Downloads$ cd linuc-4.19.67
bash: cd: linuc-4.19.67: No such file or directory
os2019202103@os2019202103-VirtualBox:~/Downloads$ cd linux-4.19.67
os2019202103@os2019202103-VirtualBox:~/Downloads/linux-4.19.67$
```

이후 $cd linux-4.19.67이라는 command를 통해 현재 렉터리에서 linux-4.19.67디렉터리로 이동합니다.

```
os2019202103@os2019202103-VirtualBox:~/Downloads/linux-4.19.67$ vi Makefile
os2019202103@os2019202103-VirtualBox:~/Downloads/linux-4.19.67$
```

```
os2019202103@os2019202103-VirtualBox: ~/Downloads/linux-4.19.67
# SPDX-License-Identifier: GPL-2.0
VERSION = 4
PATCHLEVEL = 19
SUBLEVEL = 67
EXTRAVERSION = 2019202103
NAME = "People's Front"

# *DOCUMENTATION*
# To see a list of typical targets execute "make help"
# More info can be located in ./README
# Comments in this file are targeted only to the developer, do not
# expect to learn how to build the kernel reading this file.

# That's our default target when none is given on the command line
PHONY := _all
_all:

# o Do not use make's built-in rules and variables
#   (this increases performance and avoids hard-to-debug behaviour);
# o Look for make include files relative to root of kernel src
MAKEFLAGS += -rR --include-dir=$(CURDIR)

# Avoid funny character set dependencies
"Makefile" 1736 lines, 60020 characters
```

$vi Makefile을 하여 Makefile의 내용을 vi 명령어를 이용하여 강의자료에 맞게 본인학번으로 수정합니다.

```
os2019202103@os2019202103-VirtualBox:~/Downloads/linux-4.19.67$ sudo apt install
 build-essential libncurses5-dev bison flex libssl-dev libelf-dev
Reading package lists... Done
Building dependency tree
Reading state information... Done
build-essential is already the newest version (12.1ubuntu2).
The following additional packages will be installed:
  libbison-dev libelf1 libfl-dev libsigsegv2 libssl-doc libssl1.0.0
  libtinfo-dev m4 zlib1g zlib1g-dev
Suggested packages:
  bison-doc ncurses-doc
The following NEW packages will be installed:
```

```
os2019202103@os2019202103-VirtualBox:~/Downloads/linux-4.19.67$ sudo make menuco
nfig
  UPD     scripts/kconfig/.mconf-cfg
  HOSTCC  scripts/kconfig/mconf.o
  YACC    scripts/kconfig/zconf.tab.c
  LEX     scripts/kconfig/zconf.lex.c
  HOSTCC  scripts/kconfig/zconf.tab.o
  HOSTCC  scripts/kconfig/lxdialog/checklist.o
  HOSTCC  scripts/kconfig/lxdialog/inputbox.o
  HOSTCC  scripts/kconfig/lxdialog/menubox.o
  HOSTCC  scripts/kconfig/lxdialog/textbox.o
  HOSTCC  scripts/kconfig/lxdialog/util.o
```

```
os2019202103@os2019202103-VirtualBox: ~/Downloads/linux-4.19.67
.config - Linux/x86 4.19.67 Kernel Configuration
                Linux/x86 4.19.67 Kernel Configuration
  Arrow keys navigate the menu.  <Enter> selects submenus ---> (or empty
  submenus ----).  Highlighted letters are hotkeys.  Pressing <Y>
  includes, <N> excludes, <M> modularizes features.  Press <Esc><Esc> to
  exit, <?> for Help, </> for Search.  Legend: [*] built-in  [ ]

      *** Compiler: gcc (Ubuntu 5.4.0-6ubuntu1~16.04.10) 5.4.0 2016
      General setup  --->
  [*] 64-bit kernel
      Processor type and features  --->
      Power management and ACPI options  --->
      Bus options (PCI etc.)  --->
      Binary Emulations  --->
      Firmware Drivers  --->
  [*] Virtualization  --->
      General architecture-dependent options  --->
      (+)

        <Select>    < Exit >    < Help >    < Save >    < Load >
```

compile을 하기 이전 $sudo apt install build-essential libcurses5-dev bison flex libssl-dev libelf-dev명령어를 이용하여 다음 명령어를 수행할 수있는 환경을 만듭니다.이후 $ sudo make menuconfig명령어를 통해 다음 kernel 환경설정화면을 뜨게 합니다. 이후 강의 자료에 맞게 수정할 사항을 수행 한 다음 저장 후 exit합니다.

```
os2019202103@os2019202103-VirtualBox:~/Downloads/linux-4.19.67$ make -j6
scripts/kconfig/conf  --syncconfig Kconfig
  UPD     include/config/kernel.release
  DESCEND  objtool
  UPD     include/generated/utsrelease.h
  CALL    scripts/checksyscalls.sh
  CHK     include/generated/compile.h
  CC      kernel/sys.o
  CC      init/version.o
  AR      init/built-in.a
```

가상머신에 할당한 cpu core수 4의 1.5배인 6을 make-jn에서 n에 6에 할당하여 $make -j6명령어를 입력합니다.

```
os2019202103@os2019202103-VirtualBox:~/Downloads/linux-4.19.67$ make install
sh ./arch/x86/boot/install.sh 4.19.672019202103 arch/x86/boot/bzImage \
        System.map "/boot"
/sbin/installkernel: 42: /sbin/installkernel: cannot create /boot/vmlinuz-4.19.6
72019202103: Permission denied
arch/x86/boot/Makefile:155: recipe for target 'install' failed
make[1]: *** [install] Error 2
arch/x86/Makefile:300: recipe for target 'install' failed
make: *** [install] Error 2
os2019202103@os2019202103-VirtualBox:~/Downloads/linux-4.19.67$ sudo make instal
l
sh ./arch/x86/boot/install.sh 4.19.672019202103 arch/x86/boot/bzImage \
        System.map "/boot"
run-parts: executing /etc/kernel/postinst.d/apt-auto-removal 4.19.672019202103 /
boot/vmlinuz-4.19.672019202103
run-parts: executing /etc/kernel/postinst.d/initramfs-tools 4.19.672019202103 /b
oot/vmlinuz-4.19.672019202103
update-initramfs: Generating /boot/initrd.img-4.19.672019202103
```

$sudo make modules_install을 이용하여 module을 install합니다.

```
  INSTALL sound/usb/line6/snd-usb-pod.ko
  INSTALL sound/usb/line6/snd-usb-podhd.ko
  INSTALL sound/usb/line6/snd-usb-toneport.ko
  INSTALL sound/usb/line6/snd-usb-variax.ko
  INSTALL sound/usb/misc/snd-ua101.ko
  INSTALL sound/usb/snd-usb-audio.ko
  INSTALL sound/usb/snd-usbmidi-lib.ko
  INSTALL sound/usb/usx2y/snd-usb-us122l.ko
  INSTALL sound/usb/usx2y/snd-usb-usx2y.ko
  INSTALL sound/x86/snd-hdmi-lpe-audio.ko
  INSTALL virt/lib/irqbypass.ko
  DEPMOD  4.19.672019202103
os2019202103@os2019202103-VirtualBox:~/Downloads/linux-4.19.67$ make install
```

이후 compile된 kernel을 Boot Loader에 등록 하기 위해 $make install이라는 명령어를 사용합니다. 이때 명령어는 자동으로 kernel image를 /boot로 복사,system Map을 /boot로 복사 하고 Grub 부트 로더에 자동 등록 합니다.

```
Found linux image: /boot/vmlinuz-4.19.672019202103
Found initrd image: /boot/initrd.img-4.19.672019202103
Found linux image: /boot/vmlinuz-4.15.0-29-generic
Found initrd image: /boot/initrd.img-4.15.0-29-generic
Found memtest86+ image: /boot/memtest86+.elf
Found memtest86+ image: /boot/memtest86+.bin
done
os2019202103@os2019202103-VirtualBox:~/Downloads/linux-4.19.67$ vi /etc/default/
grub
```

이후 $sudo vi/etc/default/grub 하여서 자료에 맞게 수정한후 $reboot명령어를 통해 reboot합니다.

```
os2019202103@os2019202103-VirtualBox:~$ uname -r
4.19.672019202103
os2019202103@os2019202103-VirtualBox:~$
```

이후 terminal에서 커널버전을 확인합니다.

<Reference>

vi편집기 참고자료

https://macinjune.com/all-posts/mac/terminal/%EB%A7%A5-%ED%84%B0%EB%AF%B8%EB%84%90-vi-editor%EC%82%AC%EC%9A%A9%EB%B2%95-1%ED%8E%B8/
E212오류 수정상황-iamrealizer.tistory.com/47
그외는 3주차 강의자료 참고하였습니다.

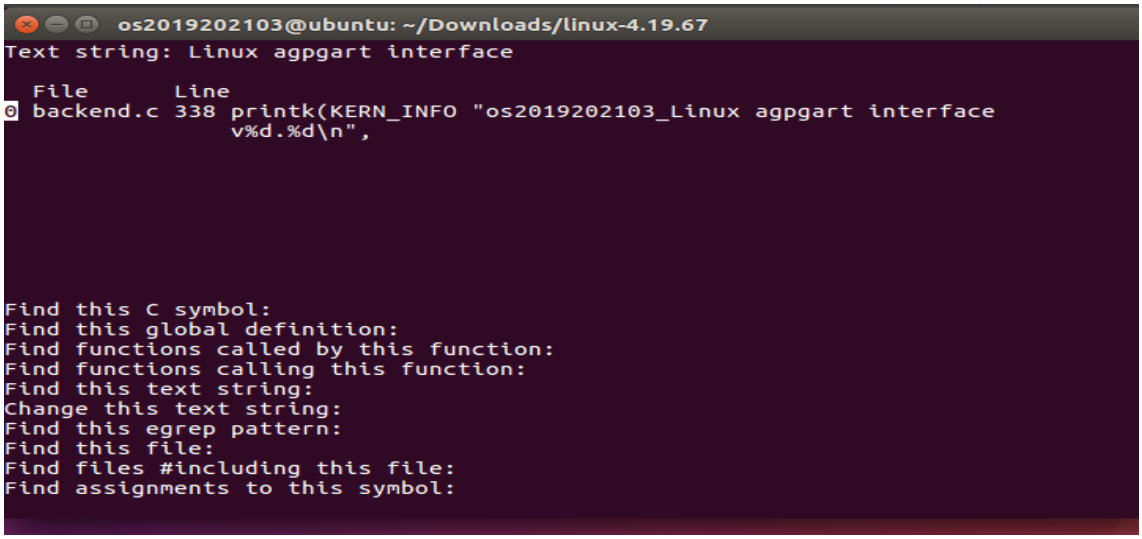
Assignment #1-3

<Introduction>

ctags와 cscope 를 통해 커널 코드 내에 prink()함수를 사용한 코드를 추가하는것이 실습 및 과제 였습니다. 수정할 코드의 내용은 Linux agpgart interface...가 실행되는 지점에 커널 메세지가 os본인학번_Linux agpgart interface v0.103와 위의 Linux agpgart interface가 실행하게 하는 함수의 함수명과 argument또한 수정하여서 값을 출력하도록 커널 코드를 수정하는 고 결과적으로 수정한 결과가 dmesg | grep "os학번" -n 의 결과로 메세지가 출력하게 하였습니다.

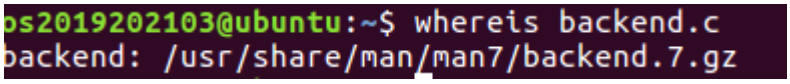
<Conclusion>

먼저 아래 실습을 진행하기위해 vim패키지 설정과 ctags,DB생성,cscope설치를 우선적으로 하였습니다. 이후 cscope -R을 통해 아래 화면으로 들어갑니다.



이후 아래 검색어 입력부분에서 Find this text string: 에 Linux agpgart interface를 입력하면 위와 같은 검색된 결과가 뜹니다. 그러면 tab을 이용하여 검색어 결과 부분에서 index 0, File name은 backend.c,line 338로 enter하여 들어갑니다.

수정한 소스코드의 path는 이후에 where is 를 한 결과 /usr/share/man/man7/backend.7.gz임을 알수 있었습니다.



```
os2019202103@ubuntu: ~/Downloads/linux-4.19.67
EXPORT_SYMBOL_GPL(agp_remove_bridge);

int agp_off;
int agp_try_unsupported_boot;
EXPORT_SYMBOL(agp_off);
EXPORT_SYMBOL(agp_try_unsupported_boot);

static int __init agp_init(void)
{
        if (!agp_off)
                printk(KERN_INFO "os2019202103_Linux agpgart interface v%d.%d\n"
,
                        AGPGART_VERSION_MAJOR, AGPGART_VERSION_MINOR);
                printk(KERN_INFO "os2019202103_arg in __init agp_init()\n");

        return 0;
}

static void __exit agp_exit(void)
{
}

#ifndef MODULE
:wq
```

원래 printk(KERN_INFO"Linux agpgart interface v%d.%d\n....);에서 printk(KERN_INFO"os2019202103_Linux agpgart interface v%d.%d\n....);으로 수정하고 바로 아랫줄에 이와 같은 interface가 시작하는 함수와 argument인 __init agp_init(void)를 확인하고 아래에 printk(KERN_INFO "os2019202103_arg in __init agp_init()\n");을 추가합니다. 이후 vi 화면을 :wq를 이용하여 저장하고 현재 화면에서 이전화면으로 돌아간 뒤 ctrl+d를 이용하여 원래의 커널 명령어 화면으로 돌아옵니다. 이후에 #make,#sudo make modules_install #make install #reboot합니다.

```
os2019202103@ubuntu:~$ dmesg | grep os2019202103 -n
1:[    0.000000] Linux version 4.19.672019202103 (os2019202103@u
ion 5.4.0 20160609 (Ubuntu 5.4.0-6ubuntu1~16.04.10)) #3 SMP Mon
PDT 2021
1264:[    9.524130] os2019202103_Linux agpgart interface v0.103
1265:[    9.524130] os2019202103_arg in __init agp_init()
os2019202103@ubuntu:~$
```

새로 시작한 termial에서 $ dmesg | grep os2019202103 -n을 입력합니다.
기본적인 정보를 제외하고 바로 아래 2개의 입력했던 코드 내용을 확인합니다.

<References>

linux 명령어 관련-https://webdir.tistory.com/158
cscope 관련-https://wonjaek.tistory.com/65
vims+ctags+cscope 관련 -http://csl.skku.edu/uploads/SSE3044F12/vim_ctags_cscope.pdf
그외 강의자료 참고