운영체제 보고서

실습과제 #03

강좌 명: 운영체제

교수님: 정준호 교수님

학과: 컴퓨터공학과

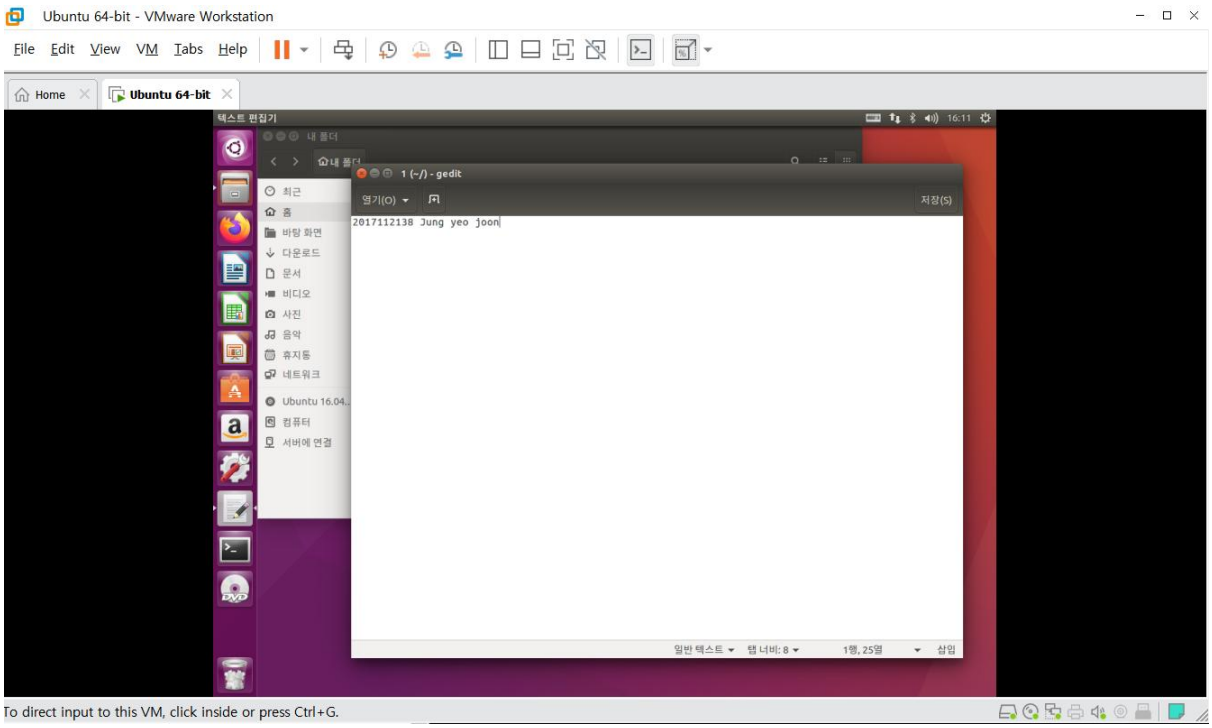
학년: 3학년

학번: 2017112138

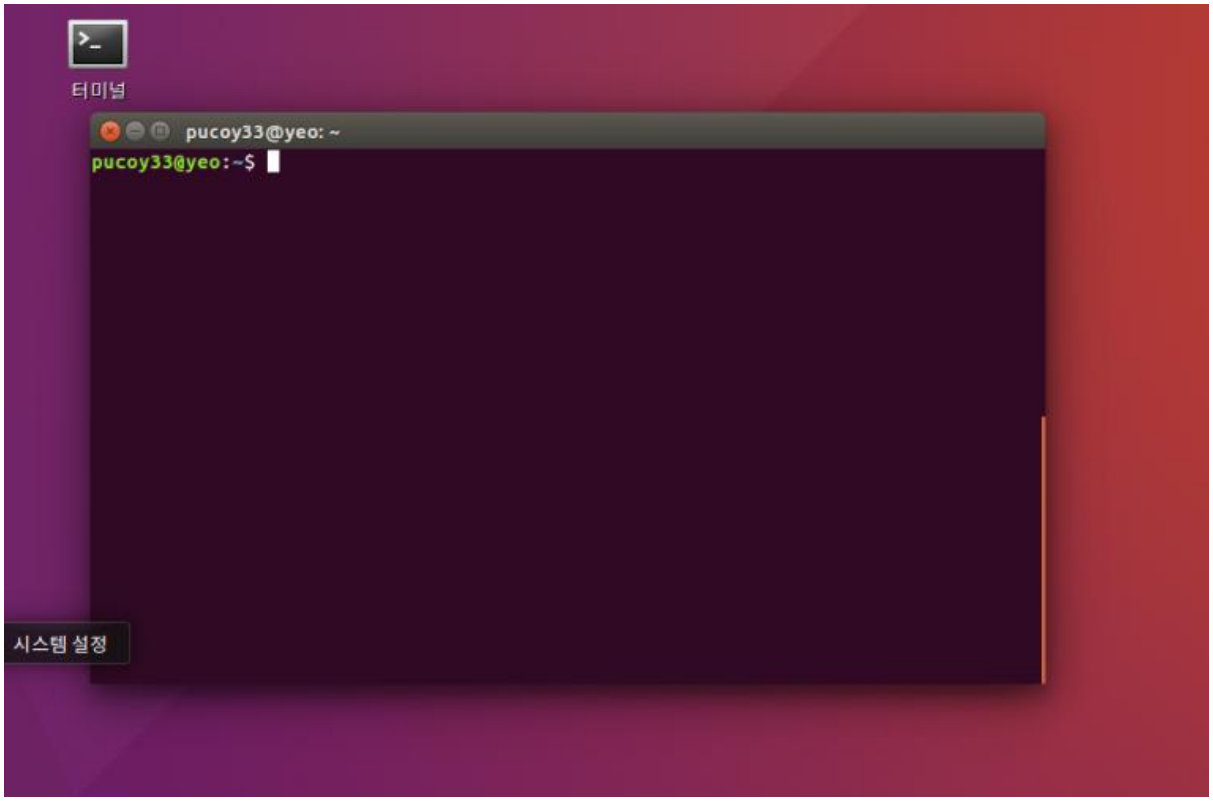
이름: 정여준

교재 문제 2.15 프로그래밍 연습문제

1.txt 작성 및 저장



2. 터미널 시작



3. vim 명령어를 통해서 copy.c파일을 다음과 같이 생성한다.

```
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
#include<fcntl.h>
#include<errno.h>
#include<sys/types.h>
#include<unistd.h>
#include<iostream>
using namespace std;

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

string inputFile, outputFile;
int inputFD,outputFD;
ssize_t bytesRd, bytesWr;
int bufferSize = 32786;
char buffer[bufferSize];
char filenameOut[256], filenameIn[256];

cout<<"First we need a file to copy from (ex: input.txt)"<<endl;
cout<<"Enter the name of the input file"<<endl;
cin>>inputFile;

cout<<"Then we need another file as destination of copied data (ex: output.txt)"<<endl;
cout<<"Enter the name of the input file"<<endl;
cin>>outputFile;

strcpy(filenameIn, inputFile.c_str());
strcpy(filenameOut, outputFile.c_str());

inputFD = open(filenameIn, O_RDONLY);
if(inputFD==-1){
perror("Error while opening Input File");
return 2;
}

outputFD = open(filenameOut, O_WRONLY | O_CREAT,0644);
if(outputFD == -1){
perror("Error while opening Output File");
return 3;
}

while((bytesRd = read(inputFD, &buffer, bufferSize))>0){
bytesWr = write(outputFD, &buffer, (ssize_t) bytesRd);
if(bytesWr != bytesRd){
perror("Error in writing");
return 4;
}
else{
cout<<"File copied correctly";
}
}
```

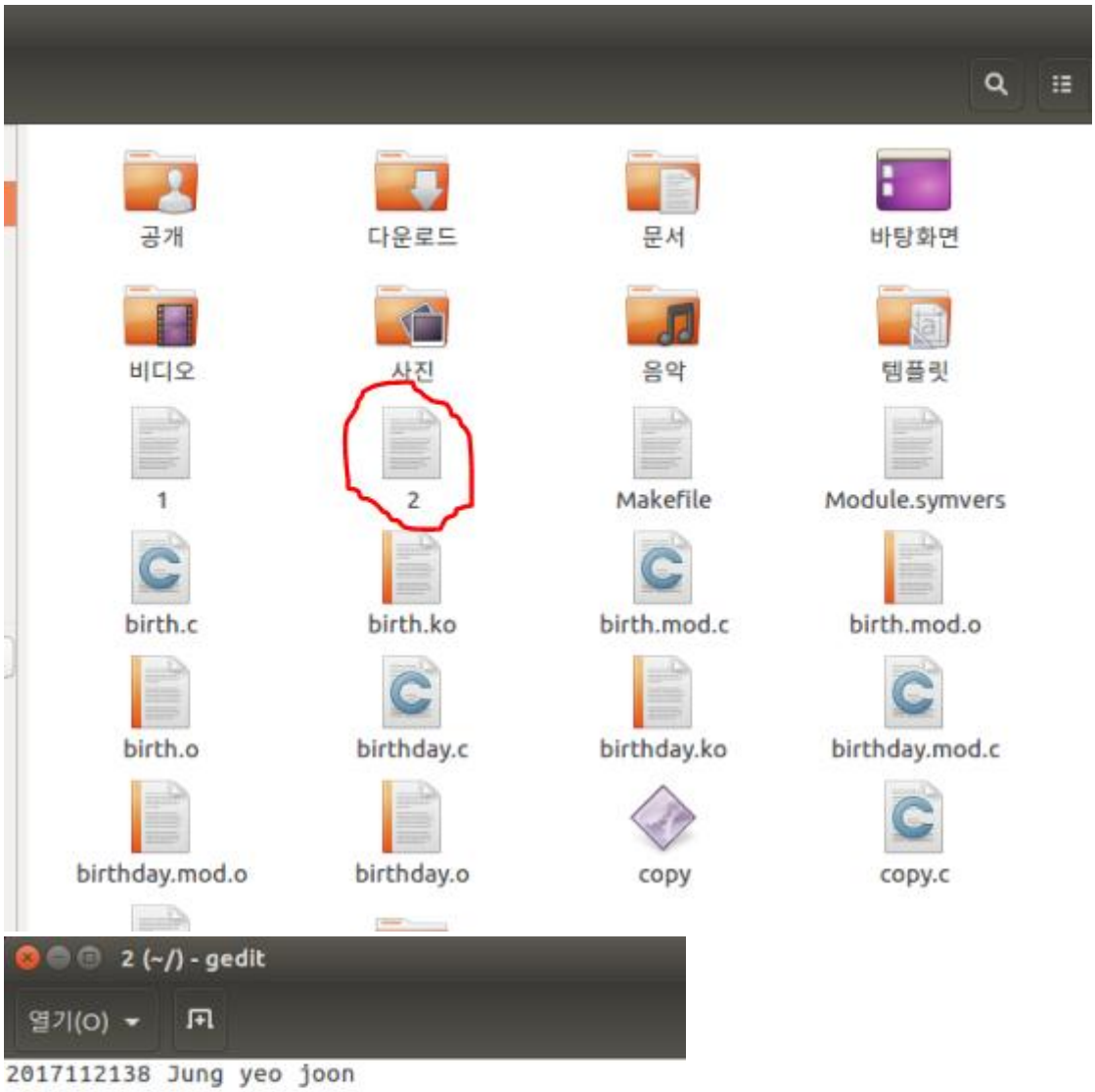
```
close(inputFD);
close(outputFD);

return 0;
}
```

```
pucoy33@yeo:~$ vim copy.c
pucoy33@yeo:~$ cat copy.c
#include<stdio.h>
#include<string.h>
#include<stdlib.h>
#include<fcntl.h>
#include<errno.h>
#include<sys/types.h>
#include<unistd.h>
#include<iostream>
using namespace std;

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

string inputFile, outputFile;
int inputFD,outputFD;
ssize_t bytesRd, bytesWr;
int bufferSize = 32786;
char buffer[bufferSize];
char filenameOut[256], filenameIn[256];

cout<<"First we need a file to copy from (ex: input.txt)"<<endl;
cout<<"Enter the name of the input file"<<endl;
cin>>inputFile;

cout<<"Then we need another file as destination of copied data (ex: output.txt)"<<endl;
cout<<"Enter the name of the input file"<<endl;
cin>>outputFile;

strcpy(filenameIn, inputFile.c_str());
strcpy(filenameOut, outputFile.c_str());

inputFD = open(filenameIn, O_RDONLY);
if(inputFD==-1){
        perror("Error while opening Input File");
        return 2;
}

outputFD = open(filenameOut, O_WRONLY | O_CREAT,0644);
if(outputFD == -1){
        perror("Error while opening Output File");
        return 3;
}

while((bytesRd = read(inputFD, &buffer, bufferSize))>0){
        bytesWr = write(outputFD, &buffer, (ssize_t) bytesRd);
        if(bytesWr != bytesRd){
                perror("Error in writing");
                return 4;
```

4. g++ -o copy copy.c 명령어를 통해 실행가능한 파일 copy를 생성한다.

5. ./copy 명령어를 통해서 실행가능한 파일 copy를 실행하고 input file로 생성해둔 1.txt를 output으로 2.txt를 만든다.

```
pucoy33@yeo:~$ g++ -o copy copy.c
pucoy33@yeo:~$ ./copy
First we need a file to copy from (ex: input.txt)
Enter the name of the input file
1
Then we need another file as destination of copied data (ex: output.txt)
Enter the name of the input file
2
File copied correctlypucoy33@yeo:~$
```



파일이 복사된 것을 확인할 수 있다.

교재 2장의 프로그래밍 프로젝트 2부과제(Part 2 Assignment) 수행

1. vim birth.c 명령어를 통해 다음과 같은 코드를 생성한다.

```
#include <linux/init.h>
#include <linux/kernel.h>
#include <linux/module.h>
#include <linux/list.h>
#include <linux/types.h>
#include <linux/slab.h>

struct birthday {
char *name;
int day;
int month;
int year;
struct list_head list;
};

static LIST_HEAD(birthday_list);

int add_birthday(char *n,int y, int m, int d)
{
struct birthday *person;

person = kmalloc(sizeof(*person), GFP_KERNEL);

if (person == NULL)
return -1;
person->name = n;
person->day = d;
person->month = m;
person->year = y;
INIT_LIST_HEAD(&person->list);


list_add_tail(&person->list, &birthday_list);

return 0;
}


int birthday_init(void)
{
struct birthday *ptr;

printk(KERN_INFO "Loading Module₩n");

add_birthday("yeo",1997, 3, 27);
add_birthday("lee",1999, 10, 14);
add_birthday("jo",1997, 8, 26);
add_birthday("oh",1998, 1, 6);
add_birthday("kim",1997, 10, 11);
```

```
list_for_each_entry(ptr, &birthday_list, list) {

printk(KERN_INFO "%s's Birthday %d-%d-%d₩n",ptr->name,ptr->year,ptr->month,ptr->day);
}
return 0;
}


void birthday_exit(void)
{
struct birthday *ptr, *next;

printk(KERN_INFO "Removing Module₩n");

list_for_each_entry_safe(ptr, next, &birthday_list, list) {
printk(KERN_INFO "%s's Birthdays deleted %d-%d-%d₩n",ptr->name,ptr->year,ptr->month,ptr->day);
list_del(&ptr->list);
kfree(ptr);
}
      printk(KERN_INFO "Memory Freed₩n");
}


module_init(birthday_init);
module_exit(birthday_exit);

MODULE_LICENSE("yeo");
MODULE_DESCRIPTION("Friends Birthday");
MODULE_AUTHOR("Jung yeo joon");
```

```
pucoy33@yeo:~$ vim birth.c
pucoy33@yeo:~$ cat birth.c
#include <linux/init.h>
#include <linux/kernel.h>
#include <linux/module.h>
#include <linux/list.h>
#include <linux/types.h>
#include <linux/slab.h>

struct birthday {
        char *name;
        int day;
        int month;
        int year;
        struct list_head list;
};

static LIST_HEAD(birthday_list);

int add_birthday(char *n,int y, int m, int d)
{
        struct birthday *person;

        person = kmalloc(sizeof(*person), GFP_KERNEL);

        if (person == NULL)
                return -1;
        person->name = n;
        person->day = d;
        person->month = m;
        person->year = y;
        INIT_LIST_HEAD(&person->list);


        list_add_tail(&person->list, &birthday_list);

        return 0;
}


int birthday_init(void)
{
        struct birthday *ptr;

        printk(KERN_INFO "Loading Module\n");

        add_birthday("yeo",1997, 3, 27);
        add_birthday("lee",1999, 10, 14);
```

```
        add_birthday("yeo",1997, 3, 27);
        add_birthday("lee",1999, 10, 14);
        add_birthday("jo",1997, 8, 26);
        add_birthday("oh",1998, 1, 6);
        add_birthday("kim",1997, 10, 11);


        list_for_each_entry(ptr, &birthday_list, list) {

                printk(KERN_INFO "%s's Birthday %d-%d-%d\n",ptr->name,ptr->year,ptr->month,ptr->day);
        }
        return 0;
}


void birthday_exit(void)
{
        struct birthday *ptr, *next;

        printk(KERN_INFO "Removing Module\n");

        list_for_each_entry_safe(ptr, next, &birthday_list, list) {
                printk(KERN_INFO "%s's Birthdays deleted %d-%d-%d\n",ptr->name,ptr->year,ptr->month,ptr->day);
                list_del(&ptr->list);
                kfree(ptr);
        }
    printk(KERN_INFO "Memory Freed\n");
}


module_init(birthday_init);
module_exit(birthday_exit);

MODULE_LICENSE("yeo");
MODULE_DESCRIPTION("Friends Birthday");
휴지통          ("Jung yeo joon");
```

2. .ko를 생성하기 위해서 vim Makefile을 통해 Makefile을 생성해준다. Makefile은 다음과 같이 생성해준다.

```
obj-m += birth.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
```

```
pucoy33@yeo:~$ vim Makefile
pucoy33@yeo:~$ cat Makefile
obj-m += birth.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

pucoy33@yeo:~$
```

여기서 obj-m은 생성할 모듈의 이름을 정해주는 것이다. 예를 들어 test.ko를 생성하고 싶다면 test.o를 인자로 주면 된다. 그래서 birth.o를 인자로 주었다.

All, clean: 여기의 make -C 옵션 값은 커널 헤더 최상위 경로이거나 커널 소스 최상위 경로이면 된다. 중요한 것은 현재 커널 버전 uname -r과 동일해야 한다는 것이다.

3. make를 통해 birth.ko를 생성해준다.

```
3@yeo: ~
pucoy33@yeo:~$ make
make -C /lib/modules/4.15.0-139-generic/build M=/home/pucoy33 modules
make[1]: 디렉터리 '/usr/src/linux-headers-4.15.0-139-generic' 들어감
  CC [M]  /home/pucoy33/birth.o
  Building modules, stage 2.
  MODPOST 1 modules
  LD [M]  /home/pucoy33/birth.ko
make[1]: 디렉터리 '/usr/src/linux-headers-4.15.0-139-generic' 나감
pucoy33@yeo:~$
```

4. sudo insmod birth.ko 명령어를 통해서 모듈을 추가해준다.

```
pucoy33@yeo:~$ sudo insmod birth.ko
[sudo] password for pucoy33:
pucoy33@yeo:~$
```

5. dmesg 명령어를 통해 커널의 로그를 출력한다.

```
pucoy33@yeo:~$ dmesg
[15586.042332] Loading Module
[15586.042334] yeo's Birthday 1997-3-27
[15586.042335] lee's Birthday 1999-10-14
[15586.042335] jo's Birthday 1997-8-26
[15586.042336] oh's Birthday 1998-1-6
[15586.042336] kim's Birthday 1997-10-11
pucoy33@yeo:~$
```

6. sudo rmmod birth 명령어를 통해서 모듈을 제거한다.

```
pucoy33@yeo:~$ sudo rmmod birth
pucoy33@yeo:~$
```

7. sudo dmesg -c 명령어를 통해서 모듈이 제대로 제거가 완료됐는지 확인한다.

```
pucoy33@yeo:~$ sudo dmesg -c
[15586.042332] Loading Module
[15586.042334] yeo's Birthday 1997-3-27
[15586.042335] lee's Birthday 1999-10-14
[15586.042335] jo's Birthday 1997-8-26
[15586.042336] oh's Birthday 1998-1-6
[15586.042336] kim's Birthday 1997-10-11
[15807.352234] Removing Module
[15807.352236] yeo's Birthdays deleted 1997-3-27
[15807.352237] lee's Birthdays deleted 1999-10-14
[15807.352238] jo's Birthdays deleted 1997-8-26
[15807.352239] oh's Birthdays deleted 1998-1-6
[15807.352239] kim's Birthdays deleted 1997-10-11
[15807.352239] Memory Freed
pucoy33@yeo:~$
```

소감문

이번 과제를 통해서 POSIX API를 사용해서 문제를 해결하는 능력을 키울 수 있게 되었다. Visual studio에서 했을 때는 간단하게 코드만 짜고 구현하면 됐었지만 리눅스에서는 리눅스만의 명령어를 또 알아야하기 때문에 리눅스의 명령어 체계에 대해서 배울 수 있어서 좋았습니다. 또한 모듈을 생성하고 로그를 확인하고 모듈을 제거하고 제거가 완벽히 이루어졌는지 확인하는 방법을 배울 수 있었습니다. 이번 과제를 통해서 저번에 만들었던 가상머신에 설치한 우분투를 더 잘 활용할 수 있게 된 것 같아서 기분이 좋았습니다. 또한 두번째 문제를 해결하면서 오랜만에 구조체를 다루고 리눅스에 있는 헤더파일에 대해서 알 수 있어서 좋았습니다. 또한 포인터를 사용해서 LinkedList를 구현해서 출력하고 리스트를 삭제하는 법에 대해서 상기시킬 수 있어서 좋았습니다. 이 과제를 통해서 모듈에 대한 이해에도 도움을 받아서 굉장히 만족스러웠습니다. 내가 만든 가상머신을 통해서 과제를 해결하는 것 또한 매우 성취감이 들었습니다. 이번 과제를 통해서 많은 부분을 배울 수 있어서 좋았습니다.