2018603020_김희찬

Step 1.

```
mjswindells@ubuntu:~/assignment7$ vi main.c
mjswindells@ubuntu:~/assignment7$ vi main.c
mjswindells@ubuntu:~/assignment7$ vi sosil1.c
mjswindells@ubuntu:~/assignment7$ vi sosil2.c
mjswindells@ubuntu:~/assignment7$ vi sosil3.c
mjswindells@ubuntu:~/assignment7$ vi sosil4.c
mjswindells@ubuntu:~/assignment7$ vi sosil5.c
mjswindells@ubuntu:~/assignment7$ vi sosil1.h
mjswindells@ubuntu:~/assignment7$ vi sosil2.h
mjswindells@ubuntu:~/assignment7$ vi sosil3.h
mjswindells@ubuntu:~/assignment7$ vi sosil4.h
mjswindells@ubuntu:~/assignment7$ vi sosil5.h
mjswindells@ubuntu:~/assignment7$ ls
main.c    sosil1.h  sosil2.h  sosil3.h  sosil4.h  sosil5.h
sosil1.c  sosil2.c  sosil3.c  sosil4.c  sosil5.c
mjswindells@ubuntu:~/assignment7$
```

과제 설명에 맞는 내용으로 소스파일과 헤더파일을 만들었다.

(내용은 따로 캡쳐를 하지 않는게 좋다고 생각하여 작성한 파일만 첨부하였습니다.)

Step 1-3.

```
 1 main: main.o sosil1.o sosil2.o sosil3.o sosil4.o sosil5.o
 2     gcc -o main main.o sosil1.o sosil2.o sosil3.o sosil4.o sosil5.o
 3 sosil1.o: sosil1.c sosil1.h
 4     gcc -c sosil1.c
 5 sosil2.o: sosil2.c sosil2.h
 6     gcc -c sosil2.c
 7 sosil3.o: sosil3.c sosil3.h
 8     gcc -c sosil3.c
 9 sosil4.o: sosil4.c sosil4.h
10     gcc -c sosil4.c
11 sosil5.o: sosil5.c sosil5.h
12     gcc -c sosil5.c
13 clean:
14     rm -f main main.o sosil1.o sosil2.o sosil3.o sosil4.o sosil5.o
```

Makefile은 어플리케이션의 구성방법을 make에 알려주는 텍스트 파일

형식

대상 : 대상에 의존되는 파일1 파일2 ...

[tap]　　명령

Step 2.

```
1 OBJF = main.o sosil1.o sosil2.o sosil3.o sosil4.o sosil5.o
2 main: $(OBJF)
3     gcc -o $@ $^
4 sosil1.o: sosil1.c sosil1.h
5     gcc -c $<
6 sosil2.o: sosil2.c sosil2.h
7     gcc -c $*.c
8 sosil3.o: sosil3.c sosil3.h
9     gcc -c $*.c
10 sosil4.o: sosil4.c sosil4.h
11     gcc -c $*.c
12 sosil5.o: sosil5.c sosil5.h
13     gcc -c $*.c
14 clean:
15     rm -f main $(OBJF)
```

내부 매크로를 이용한 Makefile을 작성.

Line 1 : 매크로를 정의할 때 띄어쓰기로 구분을 해준다.

Line 3 : $@ 현재 대상파일의 이름을 의미함 즉, main

$^ OBJF로 정의된 매크로 값

Line 5 : $< 현재 대상파일 sosil1.o가 의존하는 파일 중 첫번째 파일 이름을 의미 즉, sosil1.c

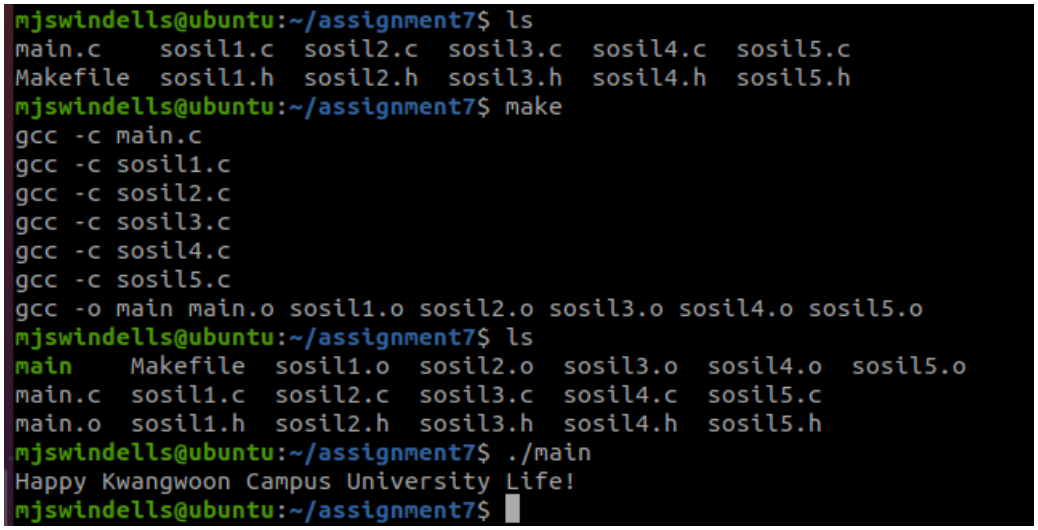
Step 3.

```
1 OBJF = main.o sosil1.o sosil2.o sosil3.o sosil4.o sosil5.o
2 main: $(OBJF)
3     gcc -o $@ $^
4 .c.o:
5     gcc -c $< $(CFLAGS)
6 #sosil1.o: sosil1.c sosil1.h
7 #   gcc -c $<
8 #sosil2.o: sosil2.c sosil2.h
9 #   gcc -c $*.c
10 #sosil3.o: sosil3.c sosil3.h
11 #   gcc -c $*.c
12 #sosil4.o: sosil4.c sosil4.h
13 #   gcc -c $*.c
14 #sosil5.o: sosil5.c sosil5.h
15 #   gcc -c $*.c
16 clean:
17     rm -f main $(OBJF)
```

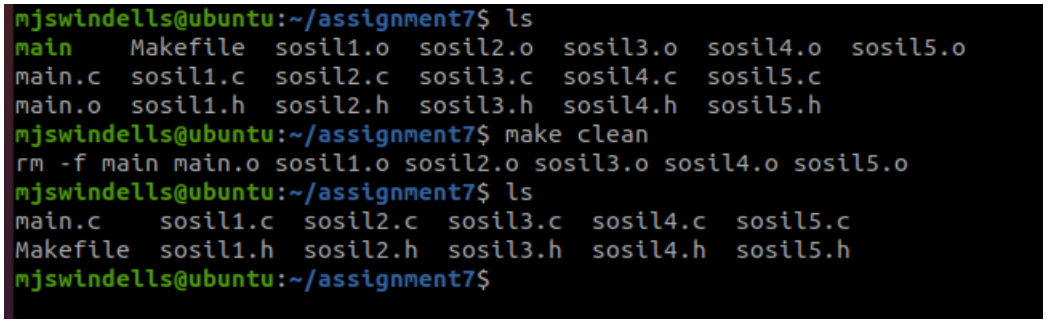
접미사 규칙을 이용한 Makefile 작성

Line 4 : .c.o .c 라는 확장자를 가진 파일을 사용해 .o라는 확장자를 가진 파일을 만들 것임을 make에 알리는 역할을 한다.

Line 5 : $(CFLAGS) C 컴파일러를 위한 플래그를 위해 미리 정의된 변수



make 명령어를 활용하여 목적파일을 만들고 파일을 실행하였다.



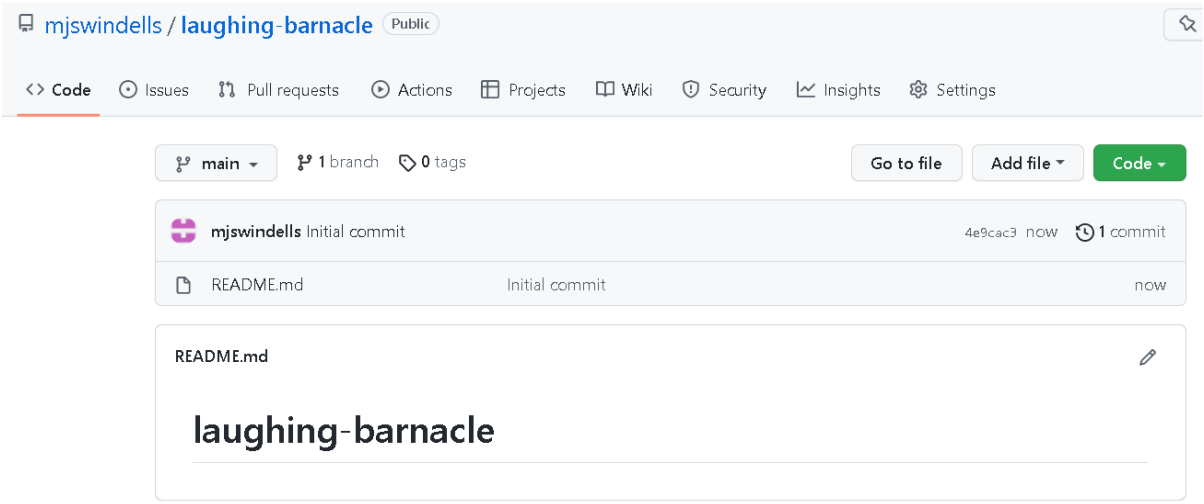
만든 실행파일들을 만들어 놓은 clean을 활용하여 생성된 목적파일을 지웠다.

Step4.

1. 소스의 기밀을 유지할 수 있다. 예를 들어 다른 회사에 A라는 파일을 보내주어야 할 경우 기밀을 위해 소스코드 대신 목적파일을 보낸다
2. 분업화가 가능하다. 서로 다른 소스코드를 만들어 목적파일을 만들고 나중에 목적파일만을 모아 합쳐서 실행파일을 만들면 되므로 분업화가 가능하다.

그 외에 컴파일 시간을 줄일 수도 있고 확장성과 이식성을 높일 수 있다.

Step 5.



Github remote repository 생성

Step 6.

```
mjswindells@ubuntu:~/assignment7$ git remote add origin https://github.com/mjswindells/laughing-
barnacle.git
mjswindells@ubuntu:~/assignment7$ git remote -v
origin  https://github.com/mjswindells/laughing-barnacle.git (fetch)
origin  https://github.com/mjswindells/laughing-barnacle.git (push)
mjswindells@ubuntu:~/assignment7$
```

현재 로컬 디렉토리와 git 저장소 연결

```
mjswindells@ubuntu:~/assignment7$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        Makefile
        main.c
        sosil1.c
        sosil1.h
        sosil2.c
        sosil2.h
        sosil3.c
        sosil3.h
        sosil4.c
        sosil4.h
        sosil5.c
        sosil5.h

nothing added to commit but untracked files present (use "git add" to track)
mjswindells@ubuntu:~/assignment7$ git add .
mjswindells@ubuntu:~/assignment7$ git commit -m "init"
```

git add .

index영역으로 현재 add되지 않은 file을 옮긴다

git commit -m “init”

index 영역에 있는 파일들을 local repository로 옮긴다.

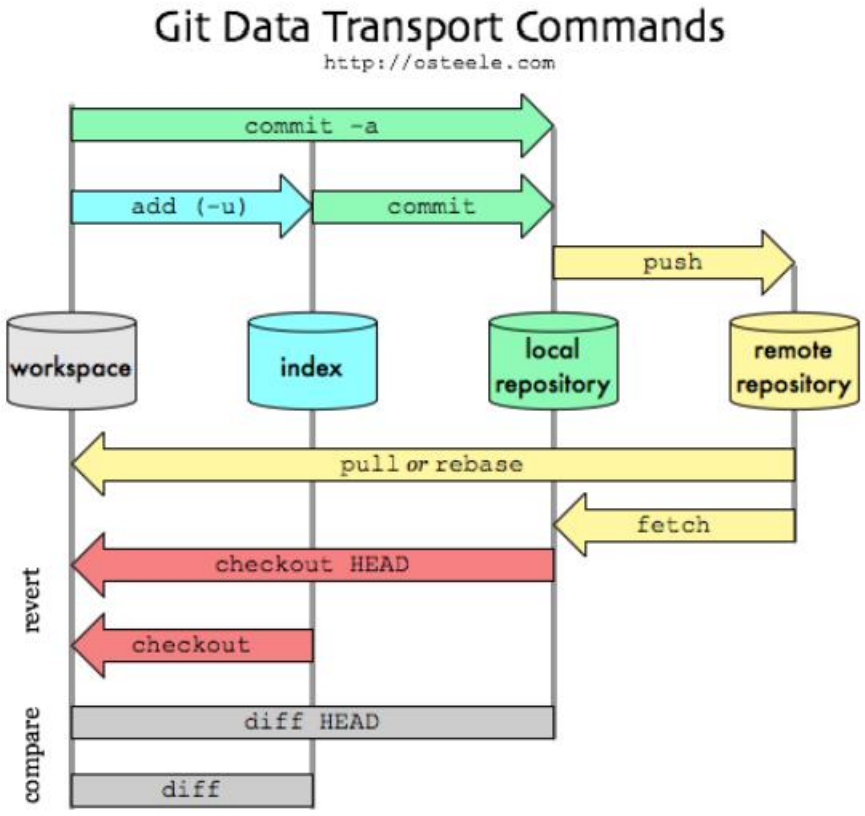
```
mjswindells@ubuntu:~/assignment7$ git push origin master
Username for 'https://github.com': mjswindells
Password for 'https://mjswindells@github.com':
Enumerating objects: 14, done.
Counting objects: 100% (14/14), done.
Delta compression using up to 3 threads
Compressing objects: 100% (9/9), done.
Writing objects: 100% (14/14), 1.31 KiB | 267.00 KiB/s, done.
Total 14 (delta 0), reused 0 (delta 0)
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:      https://github.com/mjswindells/laughing-barnacle/pull/new/master
remote:
To https://github.com/mjswindells/laughing-barnacle.git
 * [new branch]      master -> master
```

Git push origin master

Master branch로 local repository에 있는 내용들을 remote repository로 전달한다.

mjswindells init		c848f79 5 minutes ago	1 commit
Makefile	init		5 minutes ago
main.c	init		5 minutes ago
sosil1.c	init		5 minutes ago
sosil1.h	init		5 minutes ago
sosil2.c	init		5 minutes ago
sosil2.h	init		5 minutes ago
sosil3.c	init		5 minutes ago
sosil3.h	init		5 minutes ago
sosil4.c	init		5 minutes ago
sosil4.h	init		5 minutes ago
sosil5.c	init		5 minutes ago
sosil5.h	init		5 minutes ago

Github 계정의 파일들이 올라가 있는 걸 확인할 수 있다.



Git의 데이터 이동을 보여주는 표.