# 苏州大学实验报告

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 院系 | 计算机学院 | 年级专业 | 21 计科 | 姓名 | 方浩楠 | 学号 | 2127405048 |
| 课程名称 | 操作系统课程实践 | | | | | 成绩 | |
| 指导教师 | 王红玲 | 同组实验者 | 无 | 实验日期 | 2024.2.29 | | |

实验名称　Linux 系统操作及 C 语言编写

## 一. 实验目的

1. 掌握 Linux Shell 常用命令的使用。
2. 掌握 Linux 下 C 程序的编写、编译与运行方法。
3. 掌握 gcc 编译器的编译过程，熟悉编译的各个阶段。
4. 熟悉 Makefile 文件的编写格式和 make 编译工具的使用方法。

## 二. 实验内容

1. 练习常用的 Linux Shell 命令及命令选项，包括文件目录命令、备份压缩命令、重定向及管道命令等。要求熟练掌握下列命令的使用。

（1）改变及显示目录命令：cd、pwd、ls

（2）文件及目录的创建、复制、删除和移动命令：touch、cp、mv、rm、mkdir、rmdir

（3）显示文件内容命令：cat、more、less、head、tail

（4）文件查找命令：find、whereis、grep

（5）文件和目录权限改变命令：chmod

（6）备份和压缩命令：tar、gzip、bzip

2. 练习使用 gcc 编译器编译 C 程序并执行，编写 Makefile 文件，使用 make 工具编译程序并执行。具体要求：

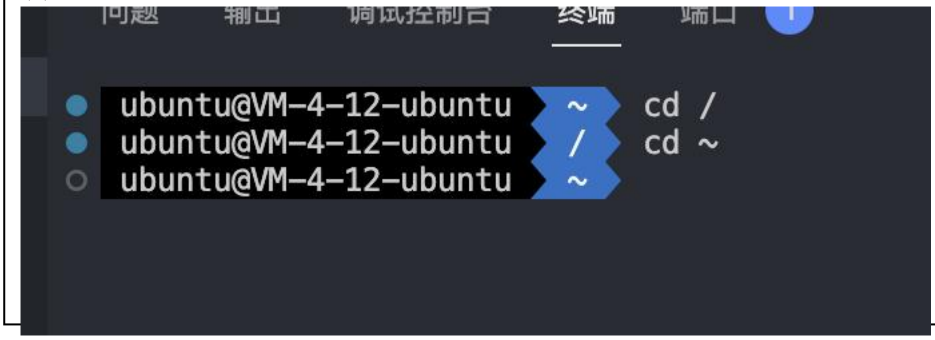
（1）编写简单的 C 程序，功能为在屏幕上输出“Hello gcc! ”。利用该程序练习使用 gcc 编译器的 E、S、c、o、g 选项，观察不同阶段所生成的文件，即*.c、*.i、*.s、*.o 文件和可执行文件。

（2）编写一个由头文件 greeting.h、自定义函数文件 greeting.c、主函数文件 myapp.c 构成的 C 程序，并根据这三个文件的依赖关系编写 Makef ile 文件。

## 三. 实验步骤和结果

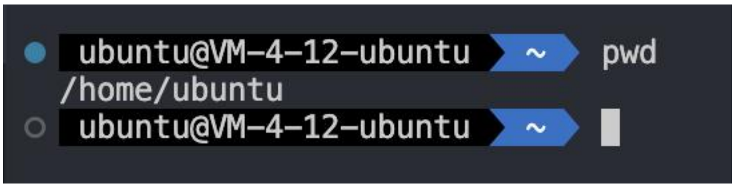
1.

(1) cd:



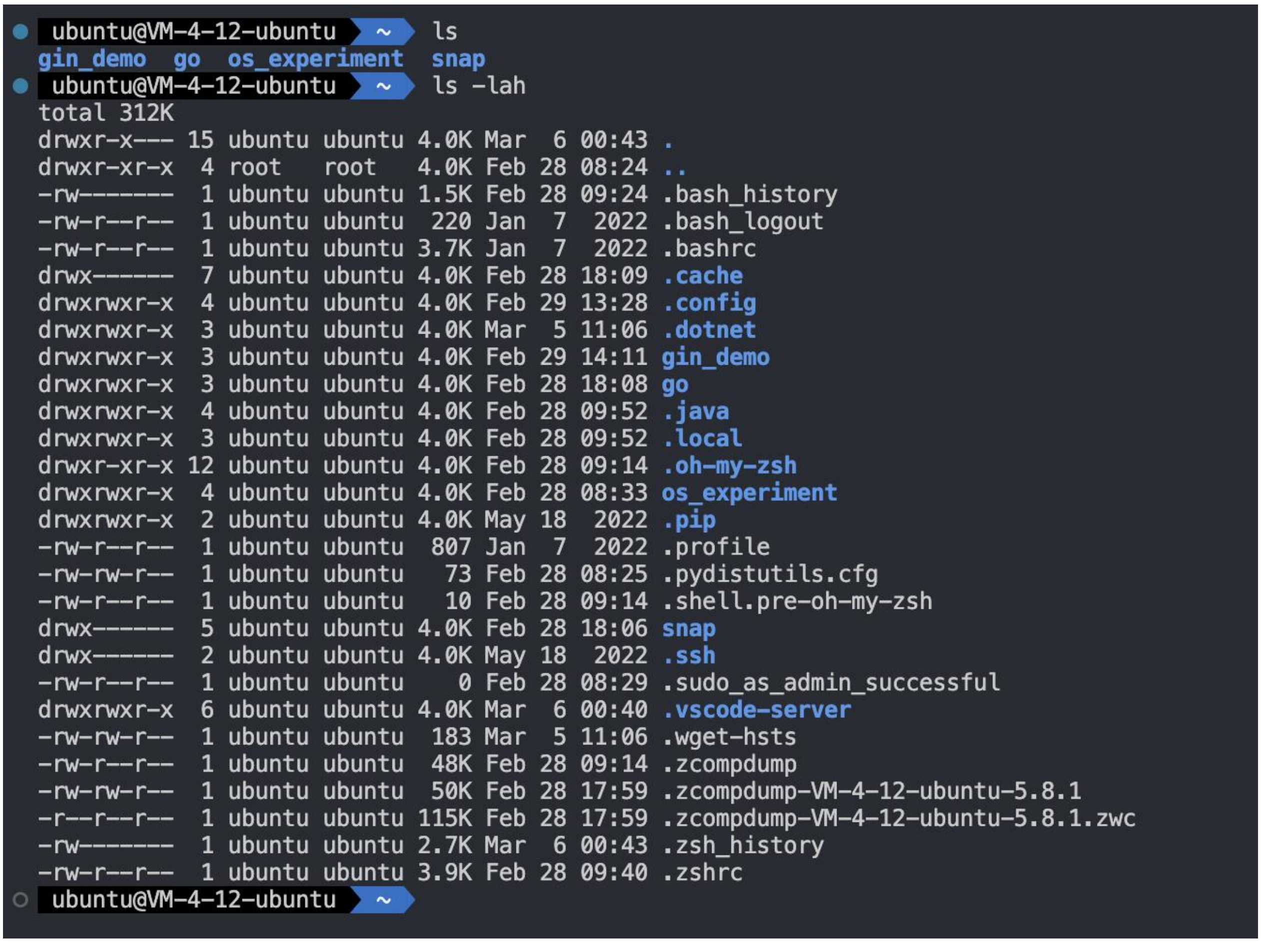
```
ubuntu@VM-4-12-ubuntu ~ cd /
ubuntu@VM-4-12-ubuntu / cd ~
ubuntu@VM-4-12-ubuntu ~
```

(2) pwd



```
● ubuntu@VM-4-12-ubuntu  ~  pwd
/home/ubuntu
○ ubuntu@VM-4-12-ubuntu  ~  █
```

(3) ls

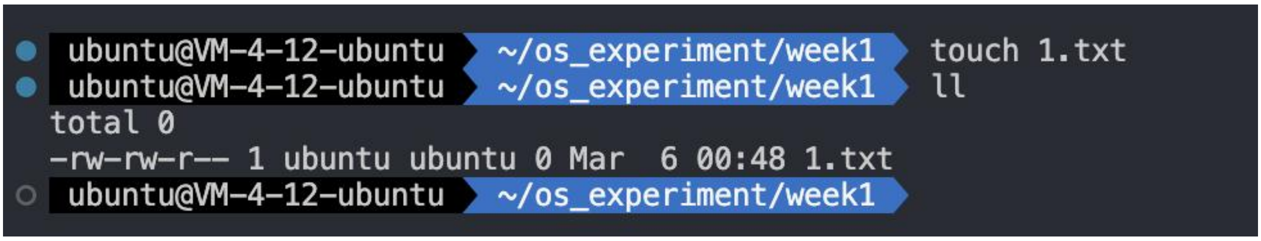


```
● ubuntu@VM-4-12-ubuntu  ~  ls
gin_demo  go  os_experiment  snap
● ubuntu@VM-4-12-ubuntu  ~  ls -lah
total 312K
drwxr-x--- 15 ubuntu ubuntu 4.0K Mar  6 00:43 .
drwxr-xr-x  4 root   root   4.0K Feb 28 08:24 ..
-rw-------  1 ubuntu ubuntu 1.5K Feb 28 09:24 .bash_history
-rw-r--r--  1 ubuntu ubuntu  220 Jan  7  2022 .bash_logout
-rw-r--r--  1 ubuntu ubuntu 3.7K Jan  7  2022 .bashrc
drwx------  7 ubuntu ubuntu 4.0K Feb 28 18:09 .cache
drwxrwxr-x  4 ubuntu ubuntu 4.0K Feb 29 13:28 .config
drwxrwxr-x  3 ubuntu ubuntu 4.0K Mar  5 11:06 .dotnet
drwxrwxr-x  3 ubuntu ubuntu 4.0K Feb 29 14:11 gin_demo
drwxrwxr-x  3 ubuntu ubuntu 4.0K Feb 28 18:08 go
drwxrwxr-x  4 ubuntu ubuntu 4.0K Feb 28 09:52 .java
drwxrwxr-x  3 ubuntu ubuntu 4.0K Feb 28 09:52 .local
drwxr-xr-x 12 ubuntu ubuntu 4.0K Feb 28 09:14 .oh-my-zsh
drwxrwxr-x  4 ubuntu ubuntu 4.0K Feb 28 08:33 os_experiment
drwxrwxr-x  2 ubuntu ubuntu 4.0K May 18  2022 .pip
-rw-r--r--  1 ubuntu ubuntu  807 Jan  7  2022 .profile
-rw-rw-r--  1 ubuntu ubuntu   73 Feb 28 08:25 .pydistutils.cfg
-rw-r--r--  1 ubuntu ubuntu   10 Feb 28 09:14 .shell.pre-oh-my-zsh
drwx------  5 ubuntu ubuntu 4.0K Feb 28 18:06 snap
drwx------  2 ubuntu ubuntu 4.0K May 18  2022 .ssh
-rw-r--r--  1 ubuntu ubuntu    0 Feb 28 08:29 .sudo_as_admin_successful
drwxrwxr-x  6 ubuntu ubuntu 4.0K Mar  6 00:40 .vscode-server
-rw-rw-r--  1 ubuntu ubuntu  183 Mar  5 11:06 .wget-hsts
-rw-r--r--  1 ubuntu ubuntu  48K Feb 28 09:14 .zcompdump
-rw-rw-r--  1 ubuntu ubuntu  50K Feb 28 17:59 .zcompdump-VM-4-12-ubuntu-5.8.1
-r--r--r--  1 ubuntu ubuntu 115K Feb 28 17:59 .zcompdump-VM-4-12-ubuntu-5.8.1.zwc
-rw-------  1 ubuntu ubuntu 2.7K Mar  6 00:43 .zsh_history
-rw-r--r--  1 ubuntu ubuntu 3.9K Feb 28 09:40 .zshrc
○ ubuntu@VM-4-12-ubuntu  ~
```

i

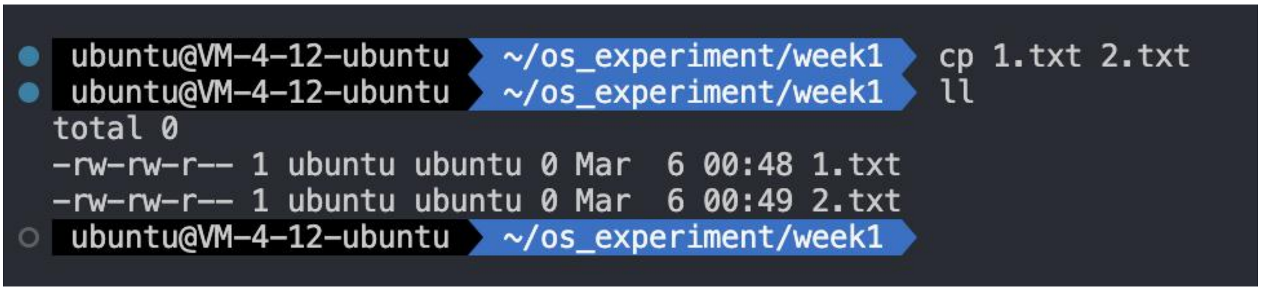
2.

(1) touch



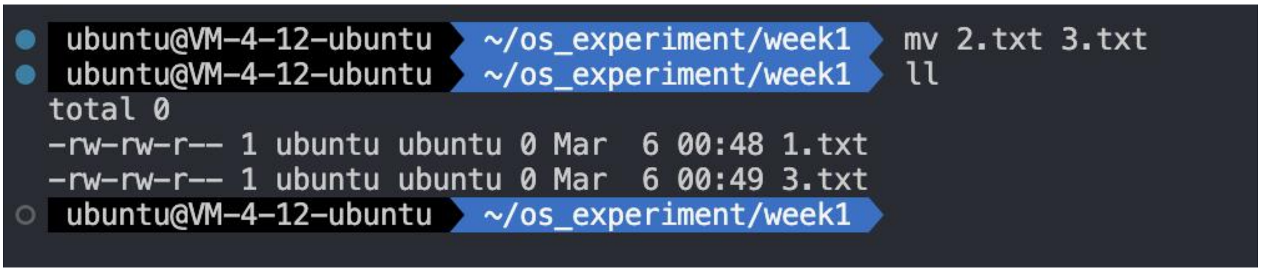
```
● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  touch 1.txt
● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  ll
total 0
-rw-rw-r-- 1 ubuntu ubuntu 0 Mar  6 00:48 1.txt
○ ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1
```

(2) cp



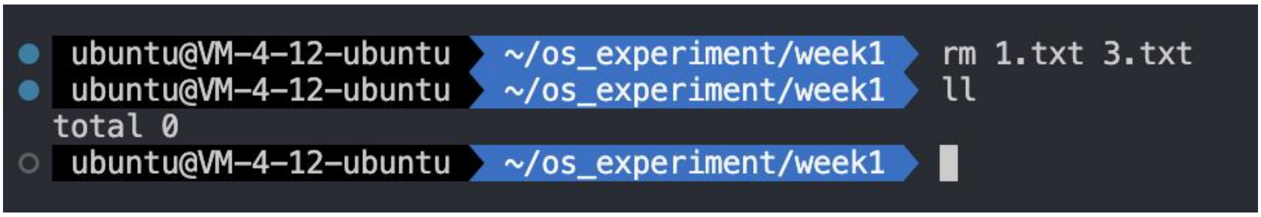
```
● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  cp 1.txt 2.txt
● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  ll
  total 0
  -rw-rw-r-- 1 ubuntu ubuntu 0 Mar  6 00:48 1.txt
  -rw-rw-r-- 1 ubuntu ubuntu 0 Mar  6 00:49 2.txt
○ ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1
```

(3) mv



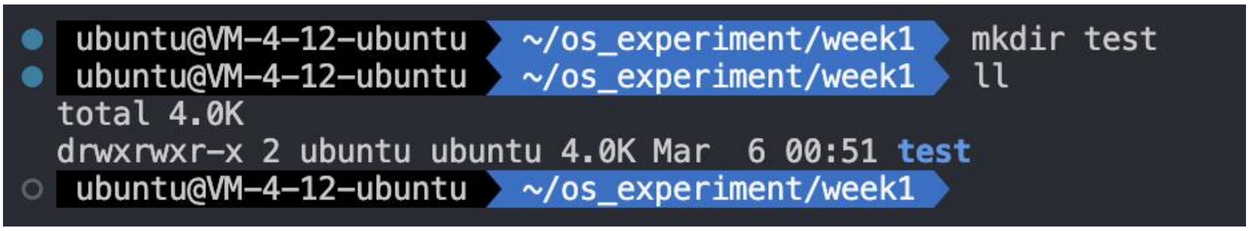
```
● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  mv 2.txt 3.txt
● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  ll
  total 0
  -rw-rw-r-- 1 ubuntu ubuntu 0 Mar  6 00:48 1.txt
  -rw-rw-r-- 1 ubuntu ubuntu 0 Mar  6 00:49 3.txt
○ ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1
```

(4)rm



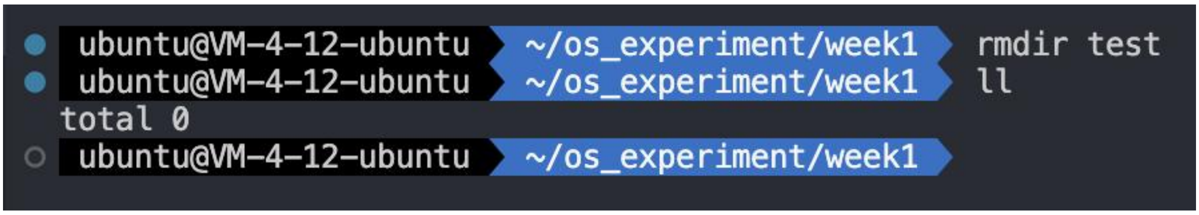
```
● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  rm 1.txt 3.txt
● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  ll
  total 0
○ ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1
```

(5)mkdir



```
● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  mkdir test
● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  ll
  total 4.0K
  drwxrwxr-x 2 ubuntu ubuntu 4.0K Mar  6 00:51 test
○ ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1
```

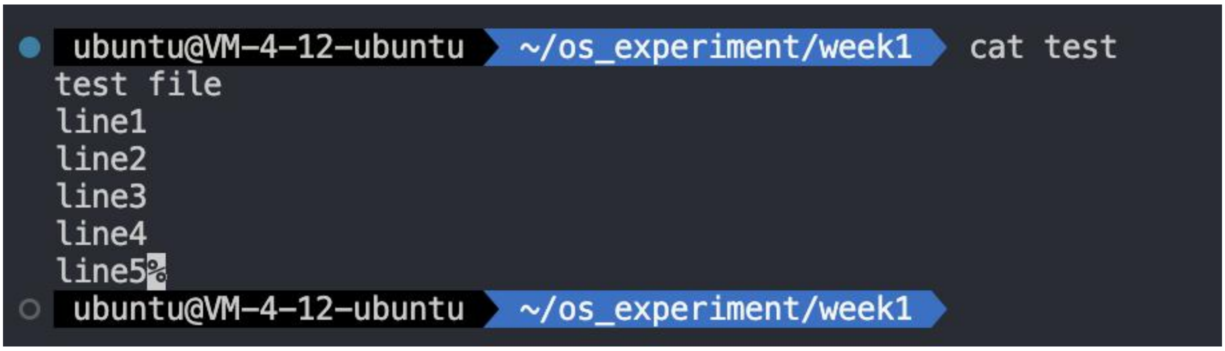
(6) rmdir



```
● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  rmdir test
● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  ll
  total 0
○ ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1
```

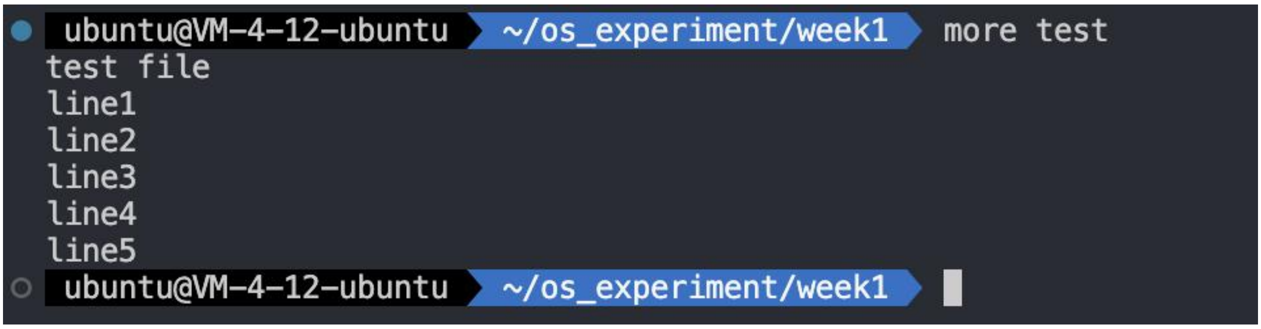
3.

(1) cat



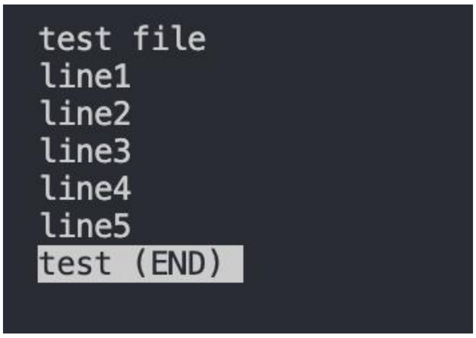
```
● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  cat test
  test file
  line1
  line2
  line3
  line4
  line5%
○ ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1
```

(2) more



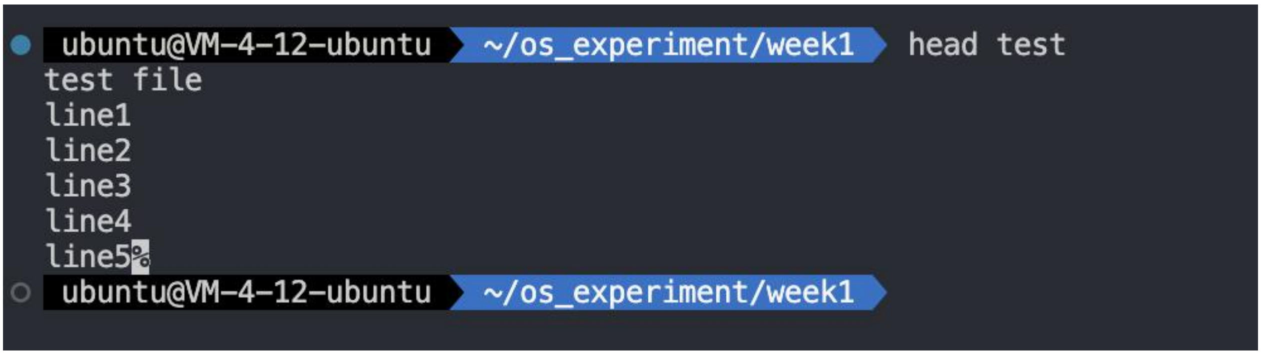
```
● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  more test
test file
line1
line2
line3
line4
line5
○ ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1
```

(3) less



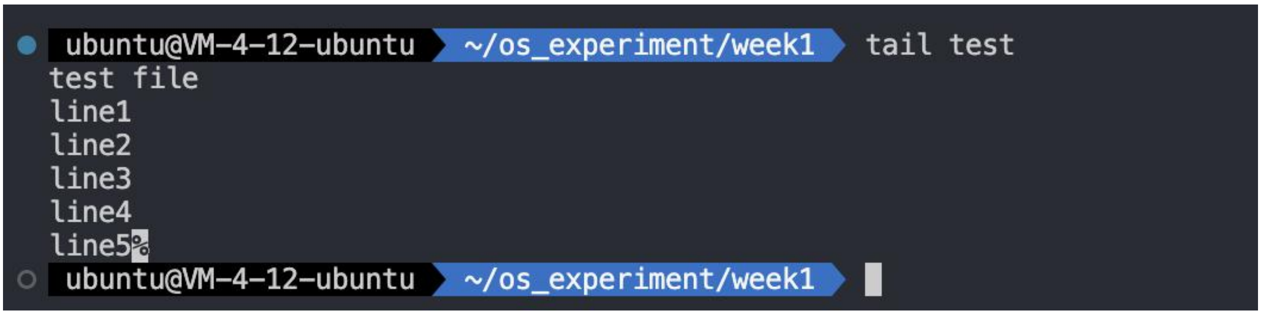
```
test file
line1
line2
line3
line4
line5
test (END)
```

(4) head



```
● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  head test
test file
line1
line2
line3
line4
line5%
○ ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1
```

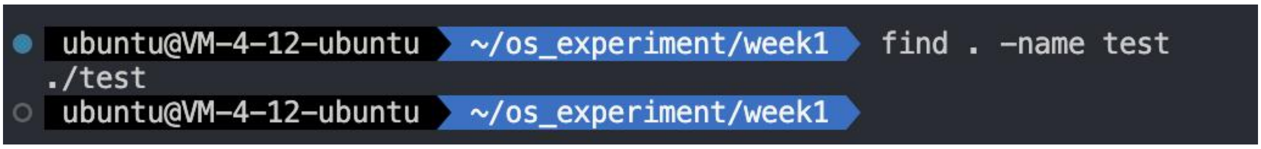
(5) tail



```
● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  tail test
test file
line1
line2
line3
line4
line5%
○ ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1
```

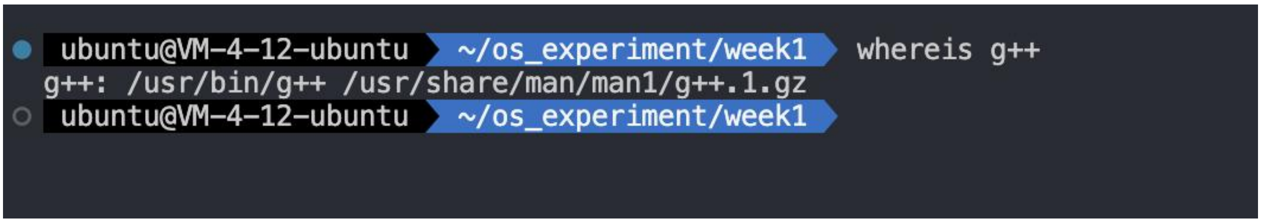
4. 文件查找指令

(1) find



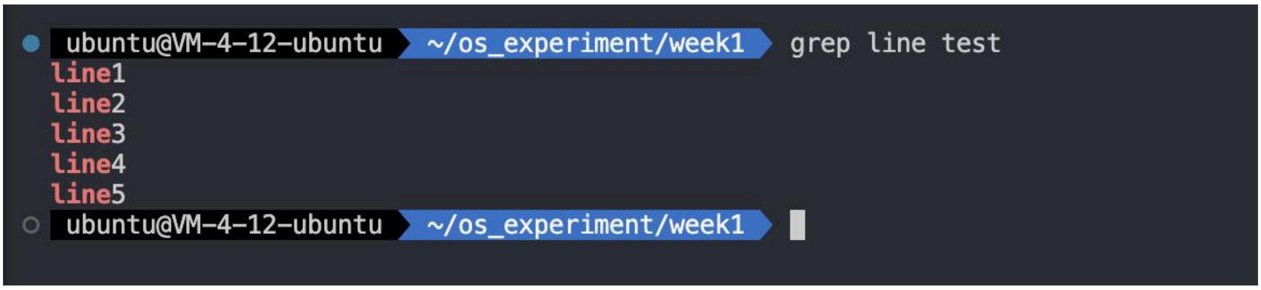
```
● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  find . -name test
./test
○ ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1
```

(2) whereis



```
● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  whereis g++
g++: /usr/bin/g++ /usr/share/man/man1/g++.1.gz
○ ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1
```

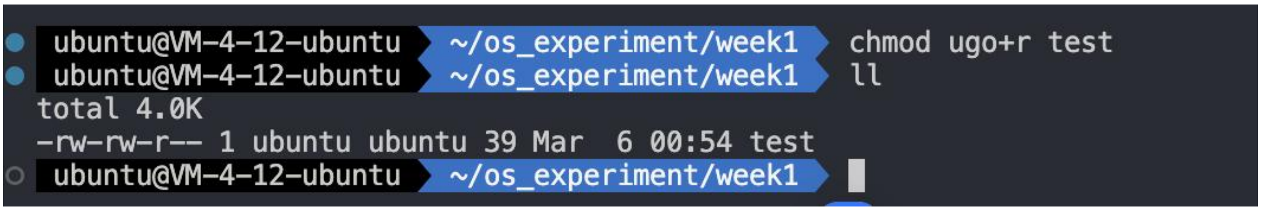
(3) grep



```
● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  grep line test
 line1
 line2
 line3
 line4
 line5
○ ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1
```

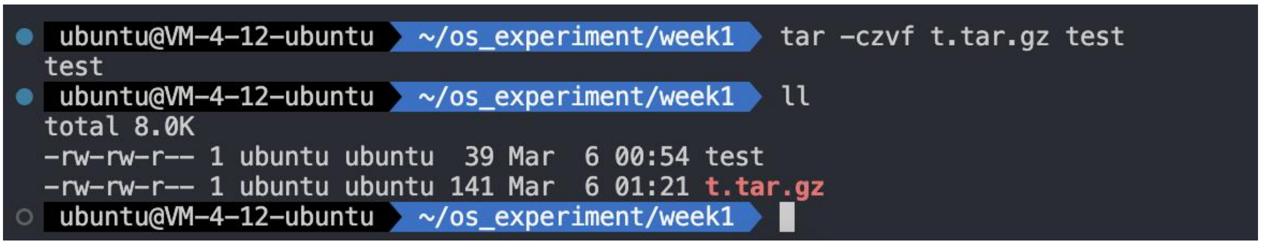
5. 改变权限

chmod

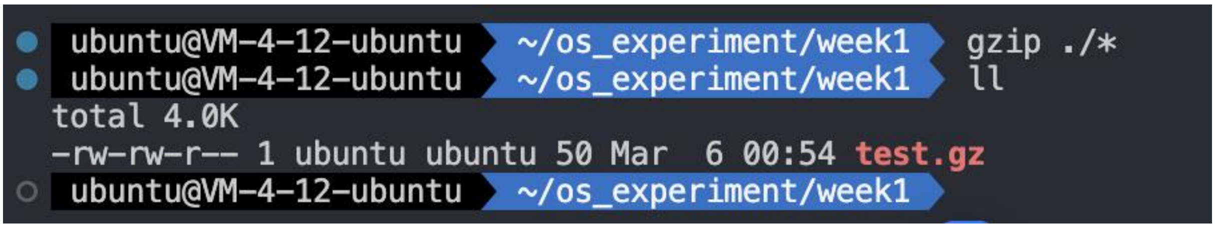


```
● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  chmod ugo+r test
● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  ll
 total 4.0K
 -rw-rw-r-- 1 ubuntu ubuntu 39 Mar  6 00:54 test
○ ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1
```

6. 备份和压缩

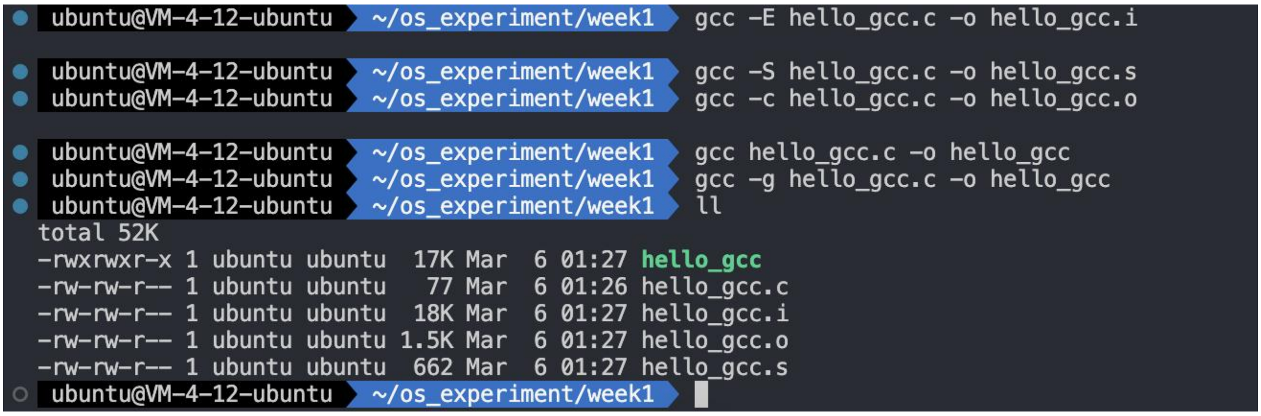


```
● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  tar -czvf t.tar.gz test
 test
● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  ll
 total 8.0K
 -rw-rw-r-- 1 ubuntu ubuntu  39 Mar  6 00:54 test
 -rw-rw-r-- 1 ubuntu ubuntu 141 Mar  6 01:21 t.tar.gz
○ ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1
```



```
● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  gzip ./*
● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  ll
 total 4.0K
 -rw-rw-r-- 1 ubuntu ubuntu 50 Mar  6 00:54 test.gz
○ ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1
```

2.



```
● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  gcc -E hello_gcc.c -o hello_gcc.i

● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  gcc -S hello_gcc.c -o hello_gcc.s
● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  gcc -c hello_gcc.c -o hello_gcc.o

● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  gcc hello_gcc.c -o hello_gcc
● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  gcc -g hello_gcc.c -o hello_gcc
● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  ll
 total 52K
 -rwxrwxr-x 1 ubuntu ubuntu  17K Mar  6 01:27 hello_gcc
 -rw-rw-r-- 1 ubuntu ubuntu   77 Mar  6 01:26 hello_gcc.c
 -rw-rw-r-- 1 ubuntu ubuntu  18K Mar  6 01:27 hello_gcc.i
 -rw-rw-r-- 1 ubuntu ubuntu 1.5K Mar  6 01:27 hello_gcc.o
 -rw-rw-r-- 1 ubuntu ubuntu  662 Mar  6 01:27 hello_gcc.s
○ ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1
```

四. 实验总结

.c 文件：这是源代码文件。

.i 文件：这是经过预处理的源代码，包含了展开的宏定义和包含的头文件内容。

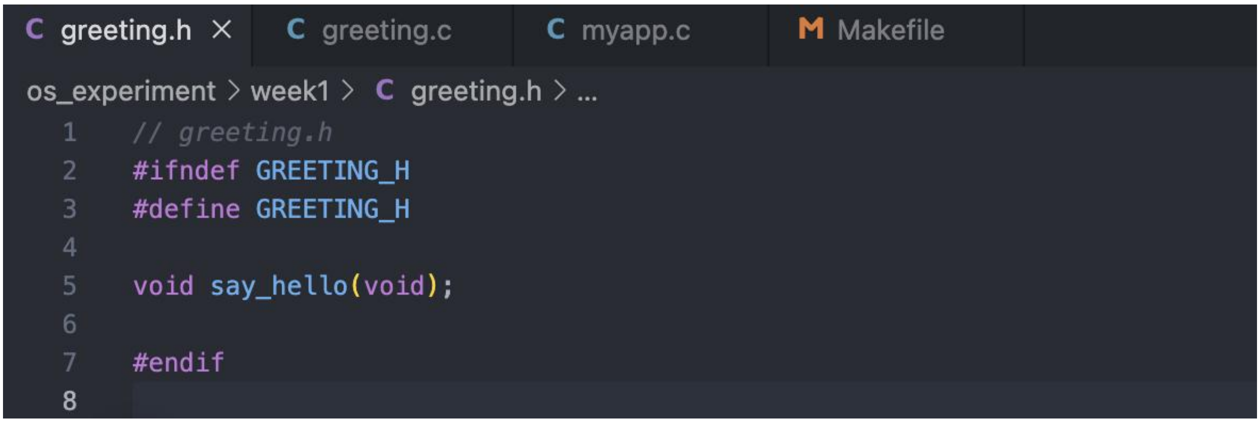
.s 文件：这是汇编语言级别的代码，展示了如何将高级语言转换为更接近机器语言的形式。

.o 文件：这是编译器输出的目标文件，它是机器语言代码，但尚未链接成最终的可执行文件。

可执行文件：经过链接过程，所有的 .o 文件被合并为一个可执行文件，它可以在操作系统上运行。

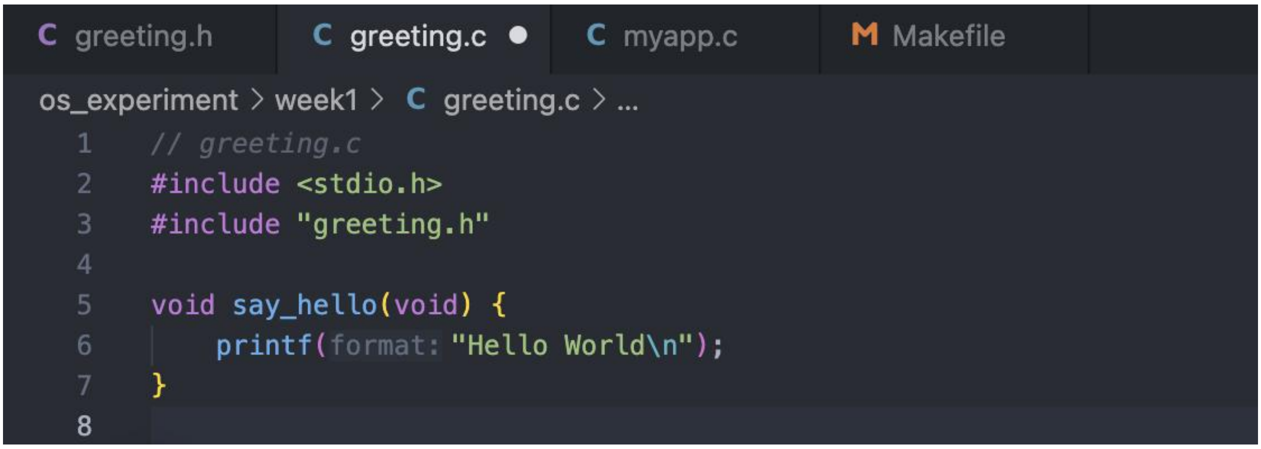
(2)

greeting.h



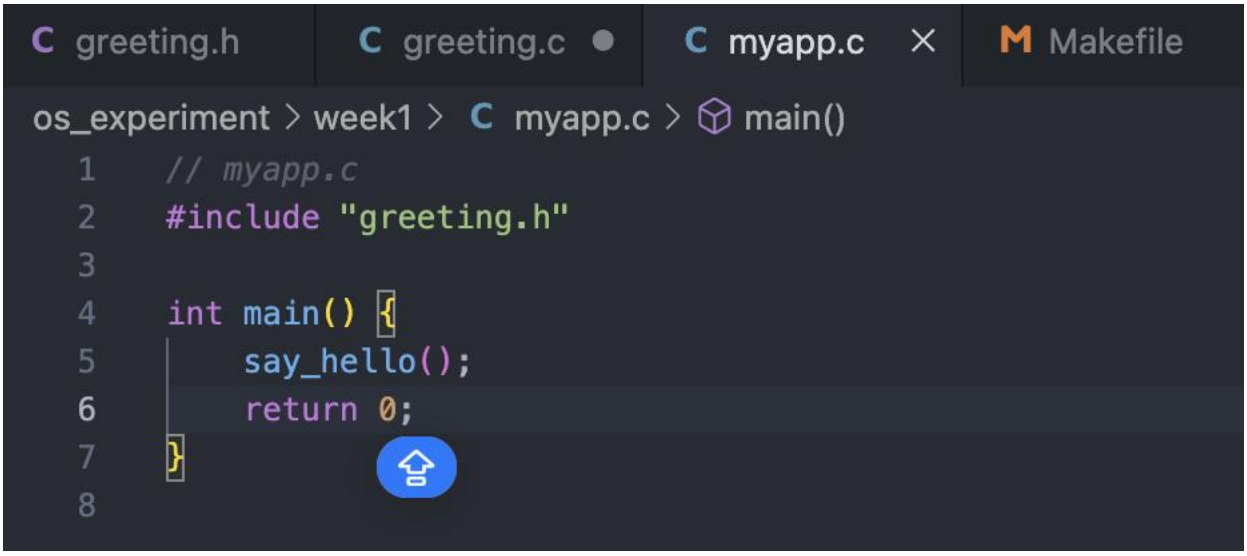
```c
// greeting.h
#ifndef GREETING_H
#define GREETING_H

void say_hello(void);

#endif

```

greeting.c



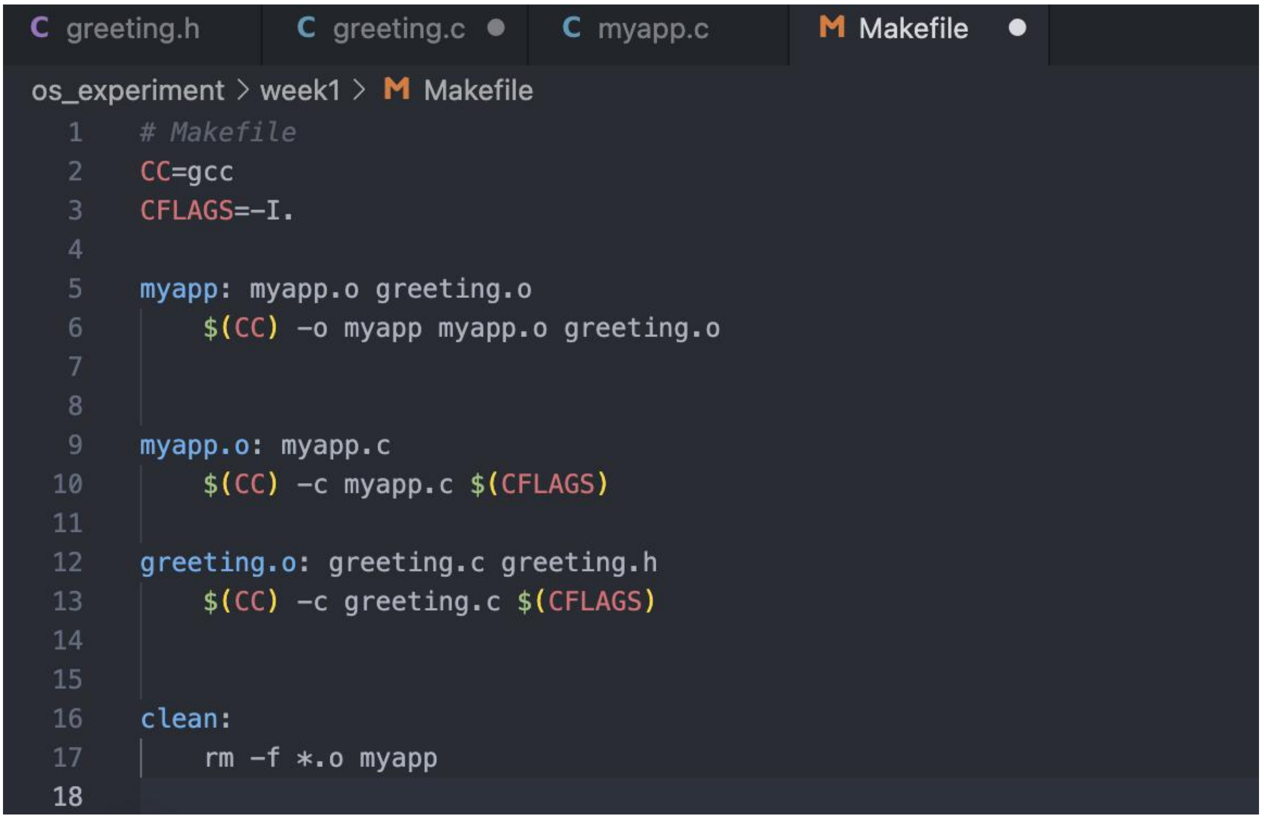
```c
// greeting.c
#include <stdio.h>
#include "greeting.h"

void say_hello(void) {
    printf(format: "Hello World\n");
}

```

myapp.c



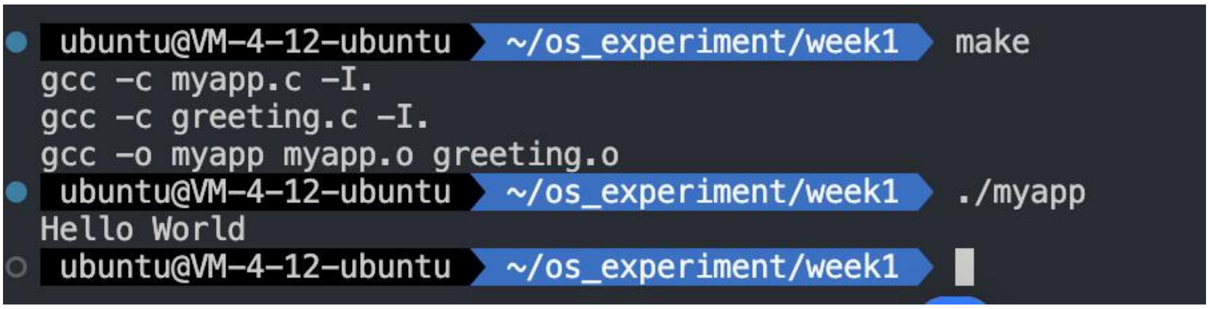
```c
// myapp.c
#include "greeting.h"

int main() {
    say_hello();
    return 0;
}

```

Makefile



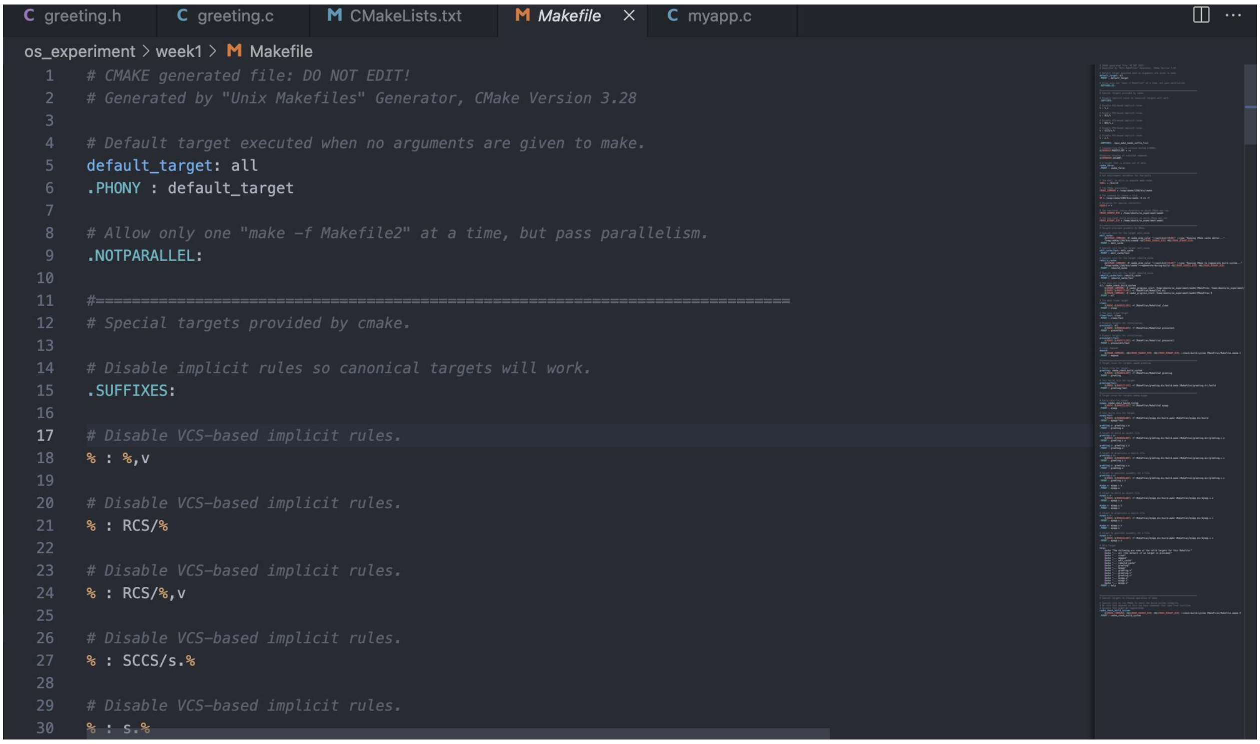
```
# Makefile
CC=gcc
CFLAGS=-I.

myapp: myapp.o greeting.o
	$(CC) -o myapp myapp.o greeting.o


myapp.o: myapp.c
	$(CC) -c myapp.c $(CFLAGS)

greeting.o: greeting.c greeting.h
	$(CC) -c greeting.c $(CFLAGS)


clean:
	rm -f *.o myapp

```

使用 makefile 编译过程



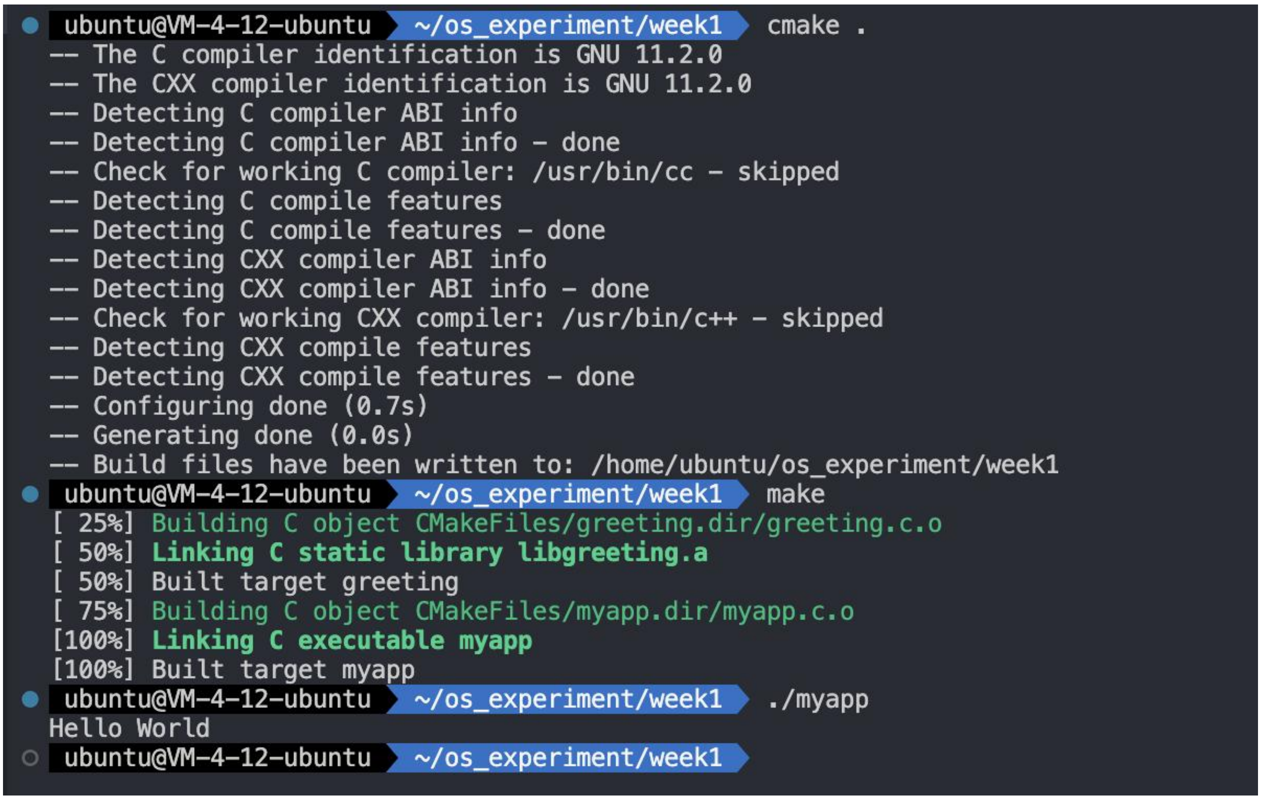
```
● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  make
gcc -c myapp.c -I.
gcc -c greeting.c -I.
gcc -o myapp myapp.o greeting.o
● ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  ./myapp
Hello World
○ ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  
```

使用 cmake 生成的 Makefile:



```
# CMAKE generated file: DO NOT EDIT!
# Generated by "Unix Makefiles" Generator, CMake Version 3.28

# Default target executed when no arguments are given to make.
default_target: all
.PHONY : default_target

# Allow only one "make -f Makefile2" at a time, but pass parallelism.
.NOTPARALLEL:

#=============================================================================
# Special targets provided by cmake.

# Disable implicit rules so canonical targets will work.
.SUFFIXES:

# Disable VCS-based implicit rules.
% : %,v

# Disable VCS-based implicit rules.
% : RCS/%

# Disable VCS-based implicit rules.
% : RCS/%,v

# Disable VCS-based implicit rules.
% : SCCS/s.%

# Disable VCS-based implicit rules.
% : s.%
```

cmake 编译过程:



```
ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  cmake .
-- The C compiler identification is GNU 11.2.0
-- The CXX compiler identification is GNU 11.2.0
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Check for working C compiler: /usr/bin/cc - skipped
-- Detecting C compile features
-- Detecting C compile features - done
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Check for working CXX compiler: /usr/bin/c++ - skipped
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Configuring done (0.7s)
-- Generating done (0.0s)
-- Build files have been written to: /home/ubuntu/os_experiment/week1
ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  make
[ 25%] Building C object CMakeFiles/greeting.dir/greeting.c.o
[ 50%] Linking C static library libgreeting.a
[ 50%] Built target greeting
[ 75%] Building C object CMakeFiles/myapp.dir/myapp.c.o
[100%] Linking C executable myapp
[100%] Built target myapp
ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1  ./myapp
Hello World
ubuntu@VM-4-12-ubuntu  ~/os_experiment/week1 
```

四:实验总结

通过本次实验，我达到了以下目的：

掌握 Linux Shell 常用命令的使用：通过练习，我熟练掌握了文件目录操作、文件内容查看、文件查找和权限改变等基础命令的使用。

掌握 Linux 下 C 程序的编写、编译与运行方法：我学会了如何在 Linux 环境下编写简单的 C 程序，

并使用 gcc 编译器进行编译和运行。

掌握 gcc 编译器的编译过程，熟悉编译的各个阶段：通过实践，我深入了解了 gcc 编译器的预处理、编译、汇编和链接等阶段，以及相应的命令选项。

熟悉 Makefile 文件的编写格式和 make 编译工具的使用方法：我学习了 Makefile 文件的基本编写规则，并利用 make 工具自动化编译了多文件 C 程序。

教务处制