

南京邮电大学

实验报告

(2024/ 2025 学年 第 一 学期)

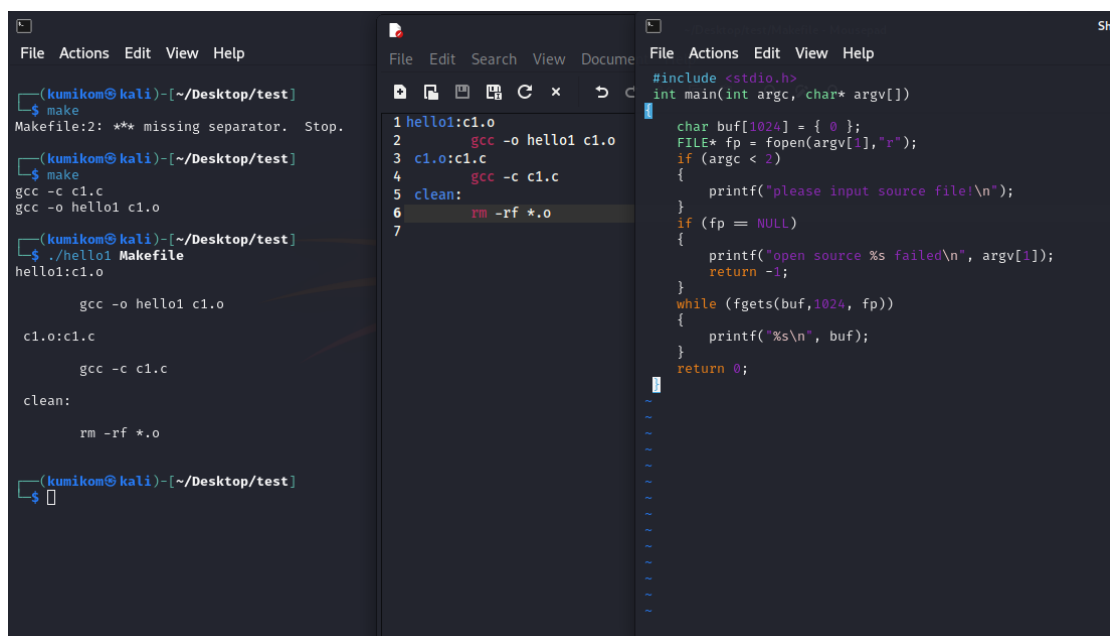
课程名称	GNU/Linux 编程			
实验名称	实验三			
实验时间	2024	年 11	月 15	日
指导单位	计算机学院 网络空间安全系			
指导教师	王磊			

学生姓名	庄元兮	班级学号	B22041024
学院(系)	计算机学院	专 业	信息安全

Task 1

Write a C program that uses standard I/O libraries to display the contents of text files. The program is compiled and linked by the make tool, which requires the generation of the .o file first, and then the generation of the executable file, and the function of deleting the intermediate file (.o) in the makefile file.

Result, Makefile, c1.c



The screenshot displays a development environment with three panels. The left panel is a terminal window showing the execution of a Makefile. The middle panel shows the Makefile content, and the right panel shows the C program code.

```
(kumikom@kali)~/Desktop/test
$ make
Makefile:2: *** missing separator. Stop.

(kumikom@kali)~/Desktop/test
$ make
gcc -c c1.c
gcc -o hello1 c1.o

(kumikom@kali)~/Desktop/test
$ ./hello1 Makefile
hello1:c1.o

gcc -o hello1 c1.o

c1.o:c1.c

gcc -c c1.c

clean:

rm -rf *.o

(kumikom@kali)~/Desktop/test
$
```

```
1 hello1:c1.o
2 gcc -o hello1 c1.o
3 c1.o:c1.c
4 c1.o: gcc -c c1.c
5 clean:
6 rm -rf *.o
7
```

```
#include <stdio.h>
int main(int argc, char* argv[])

char buf[1024] = { 0 };
FILE* fp = fopen(argv[1], "r");
if (argc < 2)
{
    printf("please input source file!\n");
}
if (fp == NULL)
{
    printf("open source %s failed\n", argv[1]);
    return -1;
}
while (fgets(buf, 1024, fp))
{
    printf("%s\n", buf);
}
return 0;
```

Task 2

Write a C program that displays all the file names in the current directory. The program is compiled and linked by the make tool, which requires the generation of the .o file first, and then the generation of the executable file, and the function of deleting the intermediate file (.o) in the makefile file.

Result, Makefile, c2.c

The image shows three terminal windows from a Kali Linux system. The left window shows the initial setup: creating a Makefile, compiling c2.c and c3.c into c2.o and c3.o, and then running the program. The middle window shows the Makefile content, which defines the compilation rules for c2.o, c3.o, and the final executable hello3. The right window shows the source code of the C program, which uses the opendir and readdir functions to traverse the directory.

```
(kumikom@kali)~/Desktop/test
$ make
gcc -c c2.c
gcc -o hello2 c2.o
(kumikom@kali)~/Desktop/test
$ ./hello2
B22041024.txt
c1.c
c2.c
.
i.sh
hello1
hello3
hello2
c2.o
c3.c
Makefile
(kumikom@kali)~/Desktop/test
$ make clean
rm -rf *.o
(kumikom@kali)~/Desktop/test
$ ls
i.sh B22041024.txt Makefile c1.c c2.c c3.c hello1 hello2 hello3
(kumikom@kali)~/Desktop/test
$
```

```
1 hello2:c2.o
2 gcc -o hello2 c2.o
3 c2.o:c2.c
4 gcc -c c2.c
5 clean:
6 rm -rf *.o
7
```

```
1 #include <stdio.h>
2 #include <dirent.h>
3 #include <sys/types.h>
4
5 int main(int argc, char* argv[])
6 {
7     DIR* dirp;
8     struct dirent* direntp;
9     if ((dirp = opendir(argv[1])) == NULL) {
10         printf("error\n");
11         // exit(1);
12     }
13     while ((direntp = readdir(dirp)) != NULL)
14         printf("%s\n", direntp->d_name);
15     closedir(dirp);
16     // exit(0);
17 }
18
```

Task 3

Write a C program that changes the working directory of the current process. The program is compiled and linked by the make tool, which requires the generation of the .o file first, and then the generation of the executable file, and the function of deleting the intermediate file (.o) in the makefile file.

Result, Makefile, c3.c

The image shows three terminal windows from a Kali Linux system. The left window shows the initial setup: creating a Makefile, compiling c3.c into c3.o, and then running the program. The middle window shows the Makefile content, which defines the compilation rules for c3.o and the final executable hello3. The right window shows the source code of the C program, which uses the chdir function to change the working directory.

```
(kumikom@kali)~/Desktop/test
$ make
gcc -c c3.c
gcc -o hello3 c3.o
(kumikom@kali)~/Desktop/test
$ ./hello3
/home/kumikom/Desktop/test
success
/home
(kumikom@kali)~/Desktop/test
$ make clean
rm -rf *.o
(kumikom@kali)~/Desktop/test
$
```

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <unistd.h>
4 int main()
5 {
6     char buf1[1024] = {0};
7     char buf2[1024] = {0};
8     getcwd(buf1, 1024);
9     printf("%s\n", buf1);
10    if(chdir("/home")<0){
11        printf("error\n");
12    }
13    else
14    {
15        printf("success\n");
16    }
17    getcwd(buf2, 1024);
18    printf("%s\n", buf2);
19    return 0;
20 }
```

```
1 hello3:c3.o
2 gcc -o hello3 c3.o
3 c3.o:c3.c
4 gcc -c c3.c
5 clean:
6 rm -rf *.o
7
```