

(1) Task 1

(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.

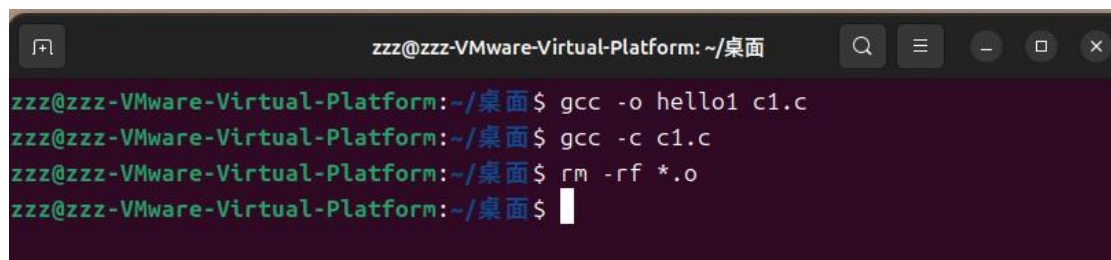
```
#include <stdio.h>
int main(int argc, char* argv[])
{
    char buf[1024] = { 0 };
    FILE* fp = fopen(argv[1], "r");
    if (argc < 2)
    {
        printf("please input source file!\n");
    }
    if (fp == NULL)
    {
        printf("open source %s failed\n", argv[1]);
        return -1;
    }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

Make sure your filename is c1.c

We can use the following makefile.

```
hello1:c1.o
    gcc -o hello1 c1.o
c1.o:c1.c
    gcc -c c1.c
clean:
    rm -rf *.o
```

该代码所实现的功能是读取命令行参数指定的文件内容，并将其输出到标准输出。程序首先检查是否提供了文件名作为命令行参数，如果没有，会提示用户输入文件名。如果提供了文件名，程序会尝试打开这个文件。如果文件打开失败，会打印错误信息并返回-1。如果文件成功打开，程序会使用`fgets`函数逐行读取文件内容，并立即将每一行打印出来。程序继续读取直到到达文件末尾。最后，程序返回 0，表示成功执行完毕，按照实验要求输入指令，可以从桌面上观察到 c1.o 文件在运行第二条命令时被创建，执行第三条命令时被删除。



```
zzy@zzy-VMware-Virtual-Platform: ~/桌面
zzy@zzy-VMware-Virtual-Platform:~/桌面$ gcc -o hello1 c1.c
zzy@zzy-VMware-Virtual-Platform:~/桌面$ gcc -c c1.c
zzy@zzy-VMware-Virtual-Platform:~/桌面$ rm -rf *.o
zzy@zzy-VMware-Virtual-Platform:~/桌面$
```

图 1 输入指令 (1)

首先创建 data 文件，内容为 B21111421，那么运行该程序可得到如下结果:

```
zzz@zzz-VMware-Virtual-Platform:~/桌面$ gcc -o hello1 c1.c
zzz@zzz-VMware-Virtual-Platform:~/桌面$ ./hello1
please input source file!
open source (null) failed
zzz@zzz-VMware-Virtual-Platform:~/桌面$ ./hello1 data
B21111421

zzz@zzz-VMware-Virtual-Platform:~/桌面$
```

图 2 实验结果 (1)

(2) Task 2

(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.

include <stdio.h>

include <dirent.h>

include <sys/types.h>

```
int main(int argc, char* argv[])
{
    DIR* dirp;
    struct dirent* direntp;
    if ((dirp = opendir(argv[1])) == NULL) {
        printf("error\n");
        // exit(1);
    }
    while ((direntp = readdir(dirp)) != NULL)
        printf("%s\n", direntp->d_name);
    closedir(dirp);
    // exit(0);
}
```

Make sure your filename is c2.c

We can use the following makefile.

```
hello2:c2.o
    gcc -o hello1 c2.o
c2.o:c2.c
    gcc -c c2.c
clean:
    rm -rf *.o
```

该代码用于列出指定目录下的所有文件和子目录。程序首先尝试打开命令行参数中给出的目录路径，如果目录打开失败，则打印“error”并退出。如果目录成功打开，程序使用`readdir`函数遍历目录中的每个条目，并将每个条目的名称打印出来。最后，程序关闭目录流并结束执行。简而言之，这个程序的功能是显示指定目录中的文件和子目录列表，首先创建 test 文件，并且在其中创建 c21、c22、c33 三个文件，如下图所示：

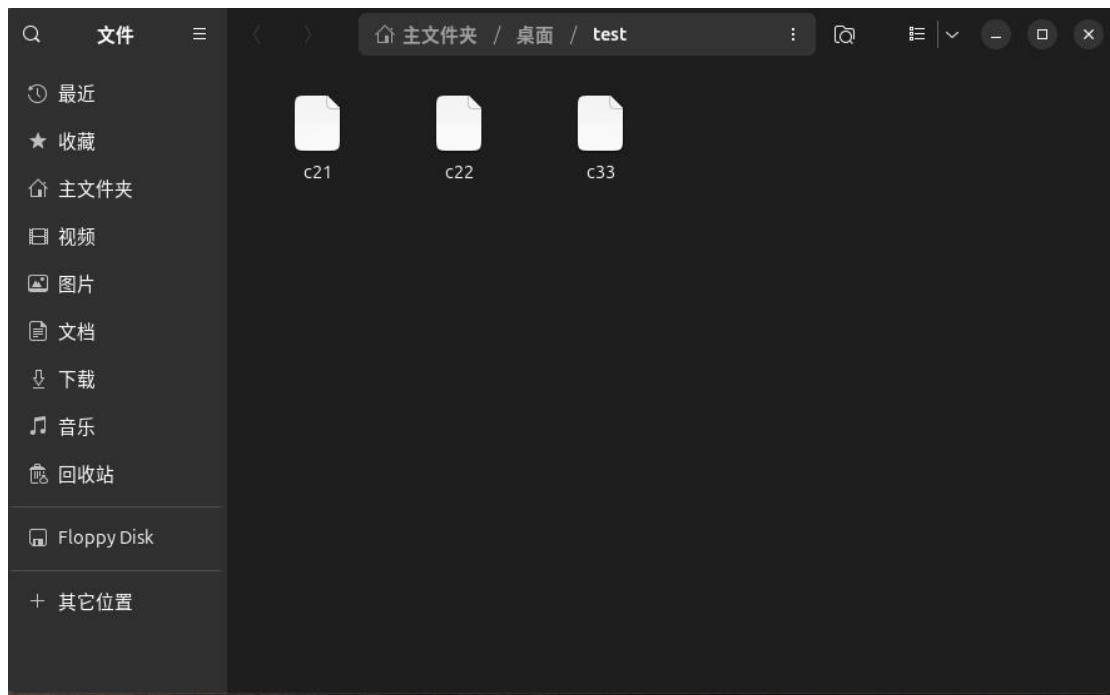


图 3 创建文件

执行`./hello2 test` 命令得到如下结果，可以看出该程序将文件的各个子目录已经打印出来。

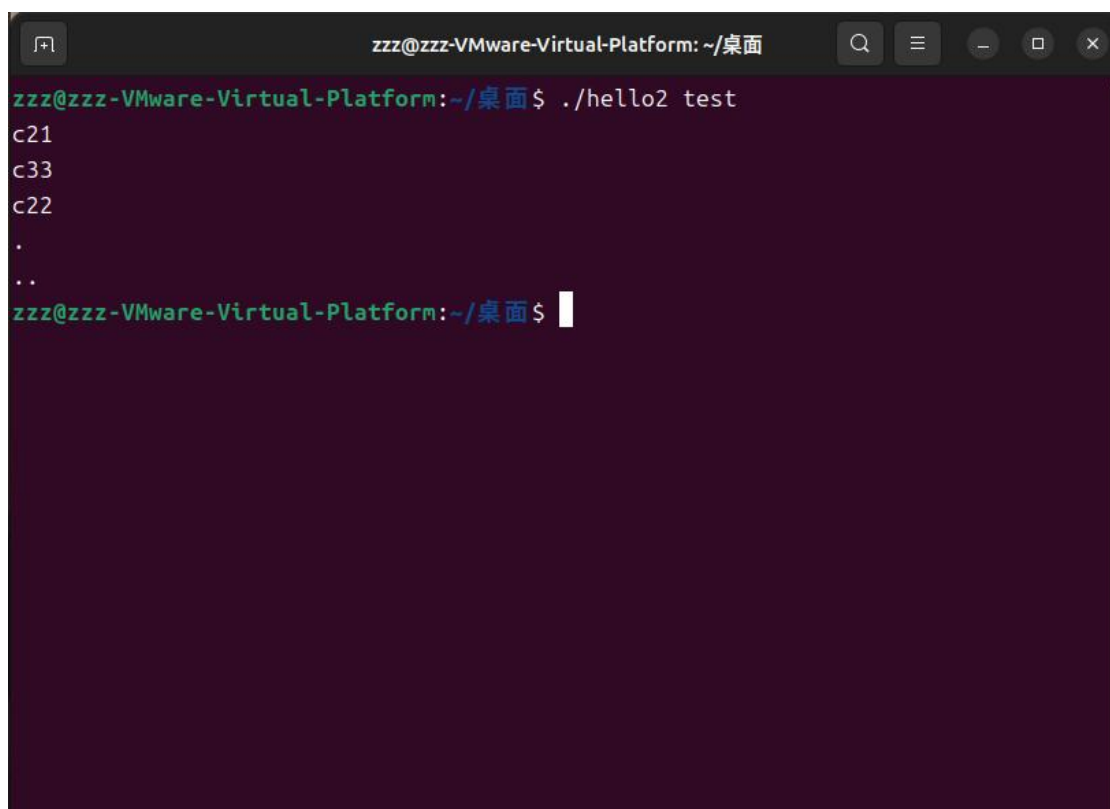


图 4 实验结果（2）

执行`./hello ~`有如下结果:

```
zzz@zzz-VMware-Virtual-Platform:~/桌面$ ./hello2 ~
桌面
.local
.cache
下载
.bash_logout
.bashrc
音乐
.sudo_as_admin_successful
模板
.config
.idlrc
文档
视频
.
..
图片
.bash_history
.ssh
.profile
snap
公共
.wget-hsts
zzz@zzz-VMware-Virtual-Platform:~/桌面$
```

图 5 实验结果（3）

(3) Task 3

(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.

```
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
int main(){
    char buf[1024] = {0};
```

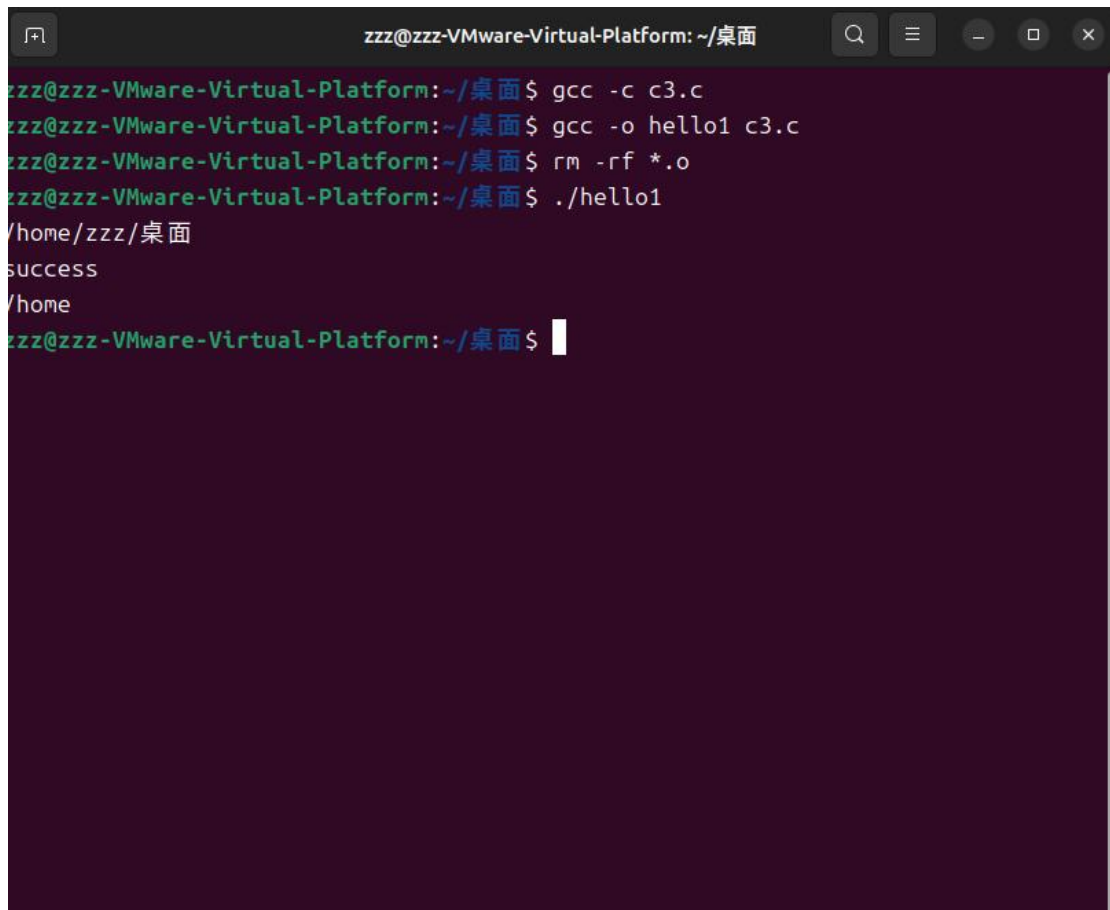
```
    char buf2[1024]={0};
    getcwd(buf, 1024);
    printf("%s\n", buf);
    if(chdir("/home")<0){
        printf("error\n");
    }
    else
    {
        printf("success\n");
    }
    getcwd(buf2,1024);
    printf("%s\n",buf2);
    return 0;
}
```

Make sure your filename is c3.c

We can use the following makefile.

```
hello3:c3.o
    gcc -o hello1 c3.o
c3.o:c3.c
    gcc -c c3.c
clean:
    rm -rf *.o
```

该代码首先使用`getcwd`函数获取当前工作目录的路径，并将其存储在`buf`数组中，然后打印出这个路径。接着，程序尝试使用`chdir`函数改变当前工作目录到`/home`目录。如果改变目录失败，程序会打印“error”，否则打印“success”。最后，程序再次使用`getcwd`函数获取改变后的当前工作目录路径，并将其存储在`buf2`数组中，然后打印出这个新的路径。简而言之，这个程序展示了如何获取和改变当前工作目录，并展示了这些操作前后的目录路径，运行程序有如下结果：



A terminal window titled "zzz@zzz-VMware-Virtual-Platform: ~/桌面" with standard window controls. The terminal shows the following commands and output:

```
zzz@zzz-VMware-Virtual-Platform:~/桌面$ gcc -c c3.c
zzz@zzz-VMware-Virtual-Platform:~/桌面$ gcc -o hello1 c3.c
zzz@zzz-VMware-Virtual-Platform:~/桌面$ rm -rf *.o
zzz@zzz-VMware-Virtual-Platform:~/桌面$ ./hello1
/home/zzz/桌面
success
/home
zzz@zzz-VMware-Virtual-Platform:~/桌面$
```

图 6 实验结果（4）