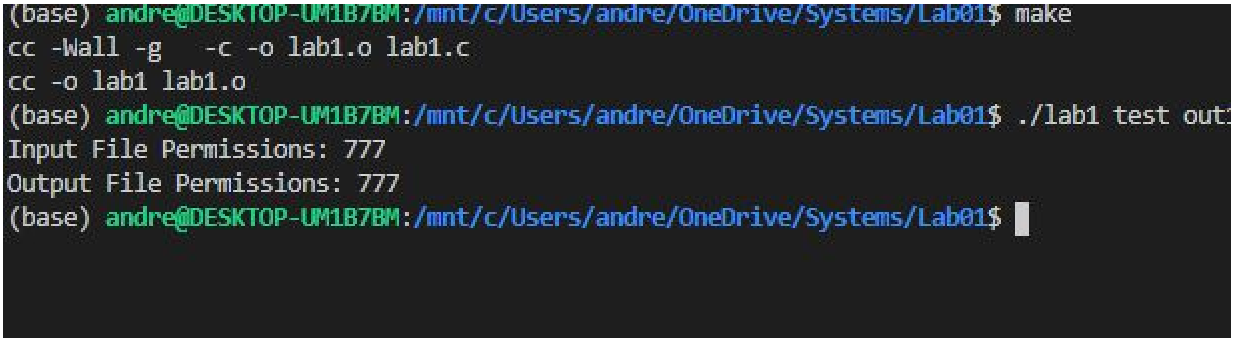
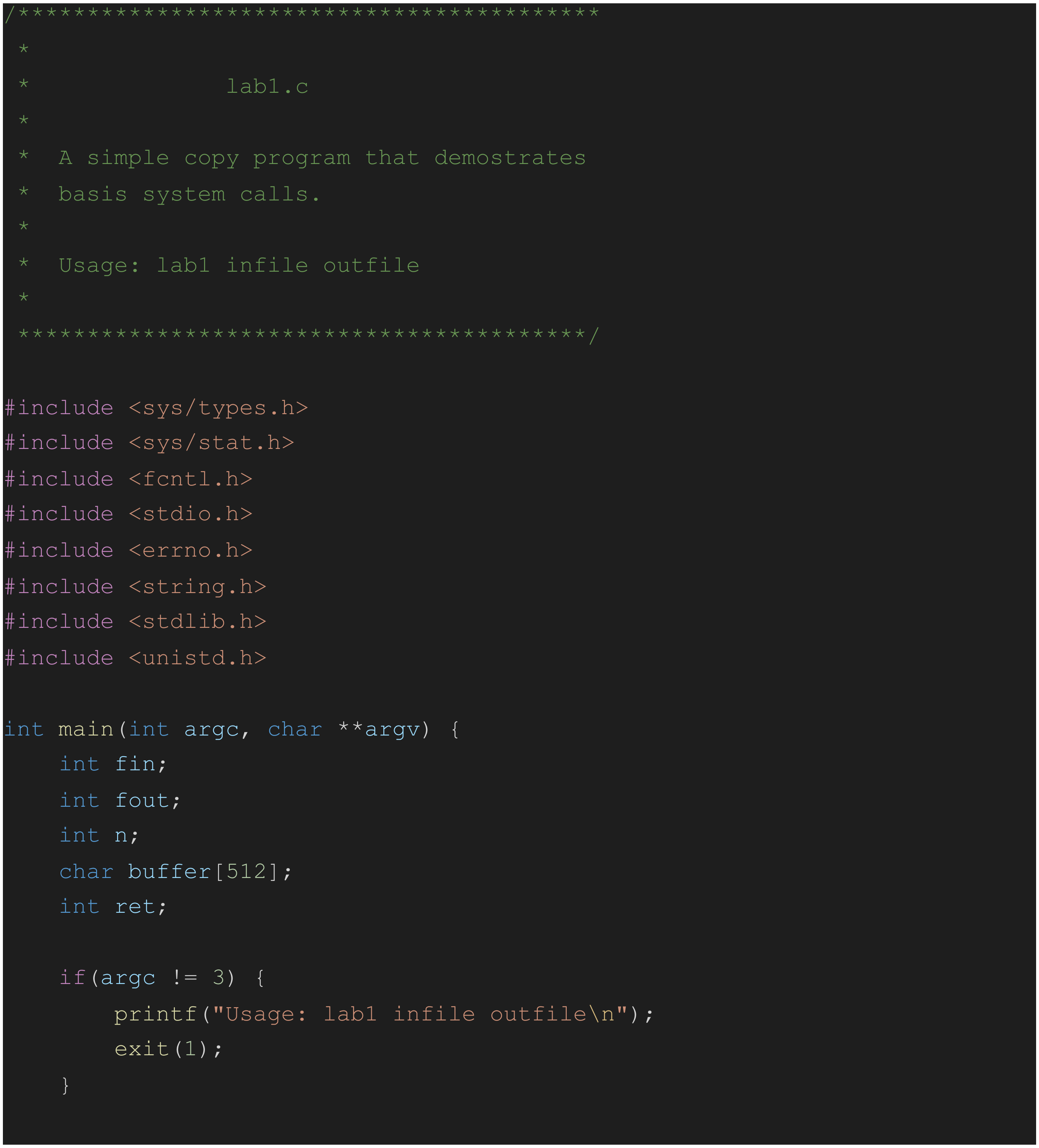


```
(base) andre@DESKTOP-UM1B7BM:/mnt/c/Users/andre/OneDrive/Systems/Lab01$ make
cc -Wall -g   -c -o lab1.o lab1.c
cc -o lab1 lab1.o
(base) andre@DESKTOP-UM1B7BM:/mnt/c/Users/andre/OneDrive/Systems/Lab01$ ./lab1 test out
Input File Permissions: 777
Output File Permissions: 777
(base) andre@DESKTOP-UM1B7BM:/mnt/c/Users/andre/OneDrive/Systems/Lab01$
```

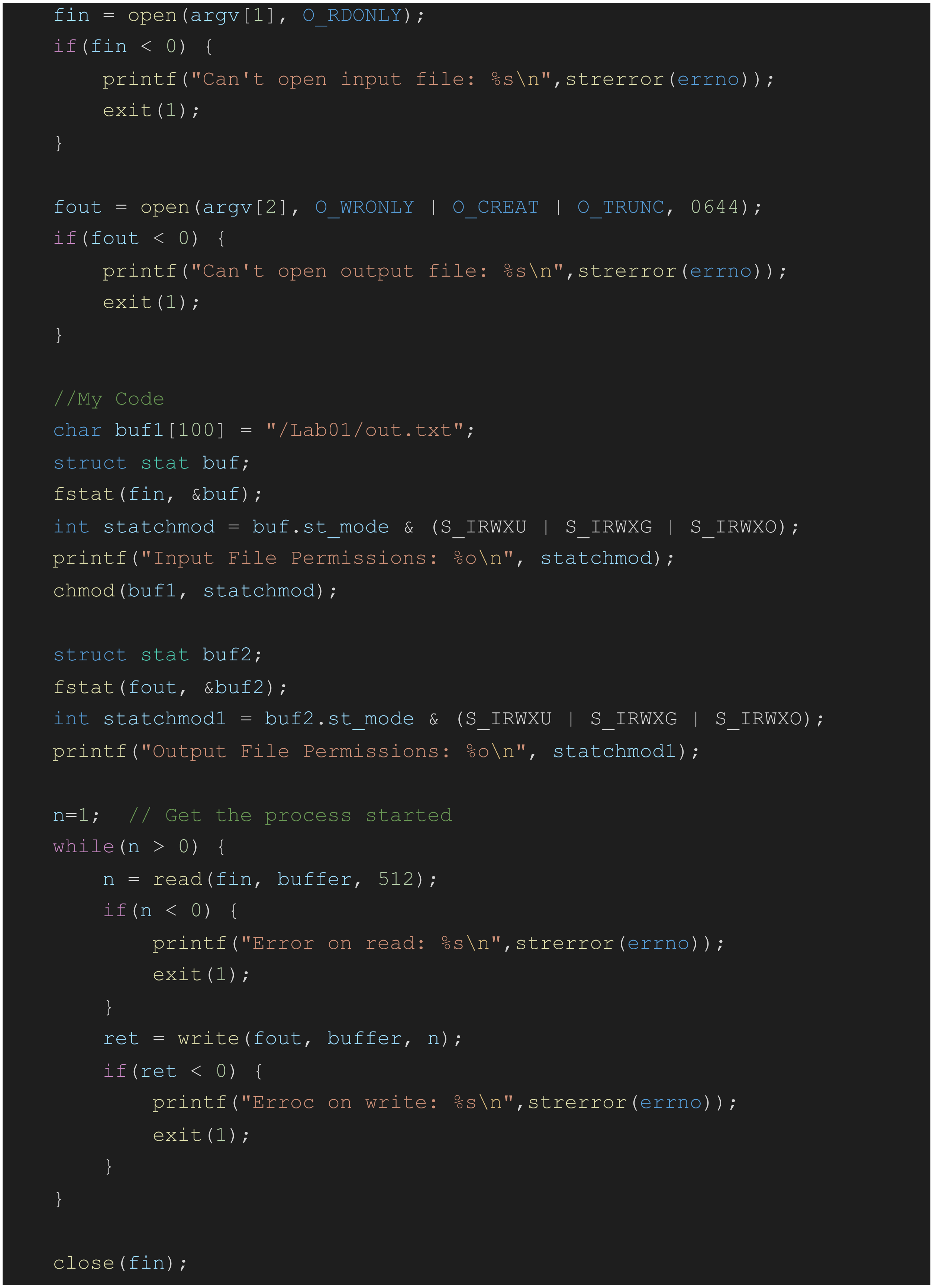


```c
/*****************************************
 *
 *              lab1.c
 *
 *  A simple copy program that demostrates
 *  basis system calls.
 *
 *  Usage: lab1 infile outfile
 *
 *****************************************/

#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <stdio.h>
#include <errno.h>
#include <string.h>
#include <stdlib.h>
#include <unistd.h>

int main(int argc, char **argv) {
    int fin;
    int fout;
    int n;
    char buffer[512];
    int ret;

    if(argc != 3) {
        printf("Usage: lab1 infile outfile\n");
        exit(1);
    }
```



```
    fin = open(argv[1], O_RDONLY);
    if(fin < 0) {
        printf("Can't open input file: %s\n",strerror(errno));
        exit(1);
    }

    fout = open(argv[2], O_WRONLY | O_CREAT | O_TRUNC, 0644);
    if(fout < 0) {
        printf("Can't open output file: %s\n",strerror(errno));
        exit(1);
    }

    //My Code
    char buf1[100] = "/Lab01/out.txt";
    struct stat buf;
    fstat(fin, &buf);
    int statchmod = buf.st_mode & (S_IRWXU | S_IRWXG | S_IRWXO);
    printf("Input File Permissions: %o\n", statchmod);
    chmod(buf1, statchmod);

    struct stat buf2;
    fstat(fout, &buf2);
    int statchmod1 = buf2.st_mode & (S_IRWXU | S_IRWXG | S_IRWXO);
    printf("Output File Permissions: %o\n", statchmod1);

    n=1;  // Get the process started
    while(n > 0) {
        n = read(fin, buffer, 512);
        if(n < 0) {
            printf("Error on read: %s\n",strerror(errno));
            exit(1);
        }
        ret = write(fout, buffer, n);
        if(ret < 0) {
            printf("Erroc on write: %s\n",strerror(errno));
            exit(1);
        }
    }

    close(fin);
    close(fout);
    exit(0);
}
```