

# FILE MANIPULATION USING SYSTEM CALL

## System Call

- **Open:** to create a file and open the file in read write and append mode.

Syntax: `int return = open("file name", O_CREAT|O_RDONLY|O_WRONLY,666);`

- **Read:** to open a file in read mode and read from file or console.  
Syntax: `int return= read(int fd, char buffer, bytes);`

- **Write:** to open a file in write mode and write on file or console.  
Syntax: `int return= write(int fd, char buffer, bytes);`

- **Close:** to close a file.

- **lseek:** to set the pointer inside the file to a position.  
Syntax: `int return= lseek(int fd, int offset, whence);`

### 1. Program to create and open a file using system calls

```
#include<stdio.h>
#include<fcntl.h>
int main()
{
    int n,m;
    n=open("open.txt",O_RDONLY);
    printf("file descriptor is %d\n",n);
    m=open("open1.txt",O_CREAT|O_WRONLY,0777);
    printf("file descriptor is %d\n",m);
    return 0;
}
```

## 2 Program to read from console and write to console

```
#include<stdio.h>
#include<fcntl.h>
#include<unistd.h>
int main()
{
    int n,m;
    char buffer[100];
    n=write(0,"HELLO WORLD",11);
    printf("number of bytes written %d\n",n);
    m=read(1,buffer,12);
    printf("\nnumber of bytes read %d\n",m);
    return 0;
}
```

## 3 Program to append file

```
#include<stdio.h>
#include<fcntl.h>
#include<unistd.h>
int main()
{
    int a,b,c,d;
    char buffer[100];
    a=open("file2.txt",O_WRONLY|O_APPEND, 0777);
    printf("file descriptor is: %d\n",a);
    b=read(0,buffer,10);
    c=write(a,buffer,b);
    printf("value of b :%d\n",b);
    printf("value of b :%d and c: %d\n",b,c);
    return 0;
}
```

#### 4 Program to read and write from and to a file

```
#include<stdio.h>
#include<fcntl.h>
#include<unistd.h>

int main()
{
    int a,b,c,d;
    char buffer[100];
    a=open("file3.txt",O_CREAT|O_WRONLY, 0777);
    printf("file descriptor is: %d\n",a);
    b=read(0,buffer,10);
    c=write(a,buffer,b);
    printf("value of b :%d and c: %d\n",b,c);
    return 0;
}
```

#### Output Of All 4 Programs:

```
chinmaye@chinmaye-virtual-machine:~/new$ ./open
file descriptor is 3
file descriptor is 4
chinmaye@chinmaye-virtual-machine:~/new$ ./write
HELLO WORLDnumber of bytes written 11

number of bytes read 1
chinmaye@chinmaye-virtual-machine:~/new$ ./append
file descriptor is: -1

value of b :1
value of b :1 and c: -1
chinmaye@chinmaye-virtual-machine:~/new$ ./write_to_file
bash: ./write_to_file: No such file or directory
chinmaye@chinmaye-virtual-machine:~/new$ gcc -o write2 write_to_file.c
chinmaye@chinmaye-virtual-machine:~/new$ ./write2
file descriptor is: 3

value of b :1 and c: 1
chinmaye@chinmaye-virtual-machine:~/new$
```

Create a program using system calls to copy either the first half or the second half of a file into a new file.

```
command 'mcat' from deb mtools (4.0.33-1+really4.0.32-1build1)
command 'zcat' from deb gzip (1.10-4ubuntu4)
Try: sudo apt install <deb name>
ubuntu@ubuntu:~/medhansh$ cat 1.c
#include <stdio.h>
#include <unistd.h>
#include <fcntl.h>

void main()
{
    char buf;
    int fd_one, fd_two;

    fd_one = open("first_file", O_RDONLY);

    if (fd_one == -1)
    {
        printf("Error opening first_file\n");
        close(fd_one);
        return;
    }

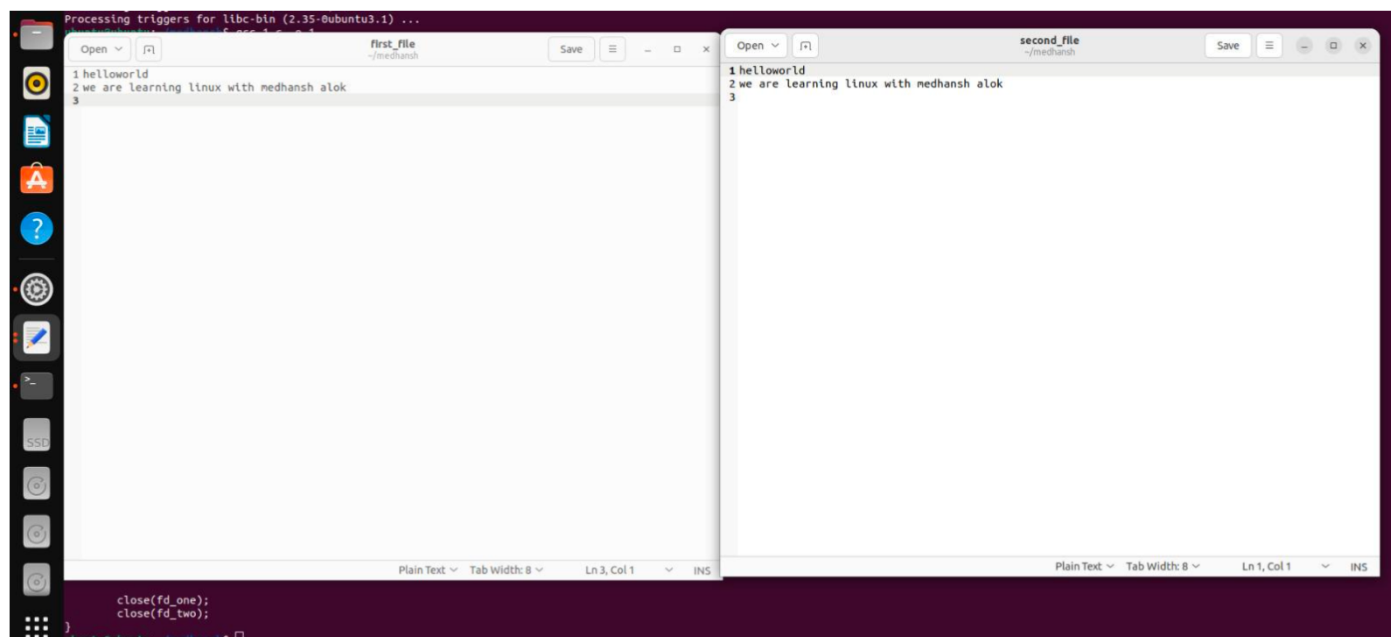
    fd_two = open("second_file",
                  O_WRONLY | O_CREAT,
                  S_IRUSR | S_IWUSR | S_IRGRP | S_IROTH);

    while(read(fd_one, &buf, 1))
    {
        write(fd_two, &buf, 1);
    }

    printf("Successful copy");

    close(fd_one);
    close(fd_two);
}
```

Output:



```
Processing triggers for libc-bin (2.35-0ubuntu3.1) ...
Open ▾ | first_file | Save | - | + | x
1 helloworld
2 we are learning linux with medhansh alok
3
Plain Text ▾ Tab Width: 8 ▾ Ln 3, Col 1 ▾ INS

close(fd_one);
close(fd_two);
ubuntu@ubuntu:~/medhansh$

Open ▾ | second_file | Save | - | + | x
1 helloworld
2 we are learning linux with medhansh alok
3
Plain Text ▾ Tab Width: 8 ▾ Ln 1, Col 1 ▾ INS
```

Develop a program using system calls to read input from the console until the user input '\$', and then save the input into a file.

```
ubuntu@ubuntu:~/medhansh$ nano 2.c
ubuntu@ubuntu:~/medhansh$ gcc 2.c -o 2
ubuntu@ubuntu:~/medhansh$ ./2
Enter input (type 'S' to stop):
hello there we are learning
linux s
$
Input saved to 'user_input.txt'
ubuntu@ubuntu:~/medhansh$ cat 2.c
#include <stdio.h>
#include <stdlib.h>
#include <fcntl.h>
#include <unistd.h>

int main() {
    char input;
    int fd;
    ssize_t bytes_written;

    fd = open("user_input.txt", O_WRONLY | O_CREAT | O_TRUNC, 0644);
    if (fd == -1) {
        perror("Error opening file");
        return 1;
    }

    printf("Enter input (type 'S' to stop):\n");

    while (1) {
        scanf("%c", &input);

        if (input == 'S') {
            break;
        }

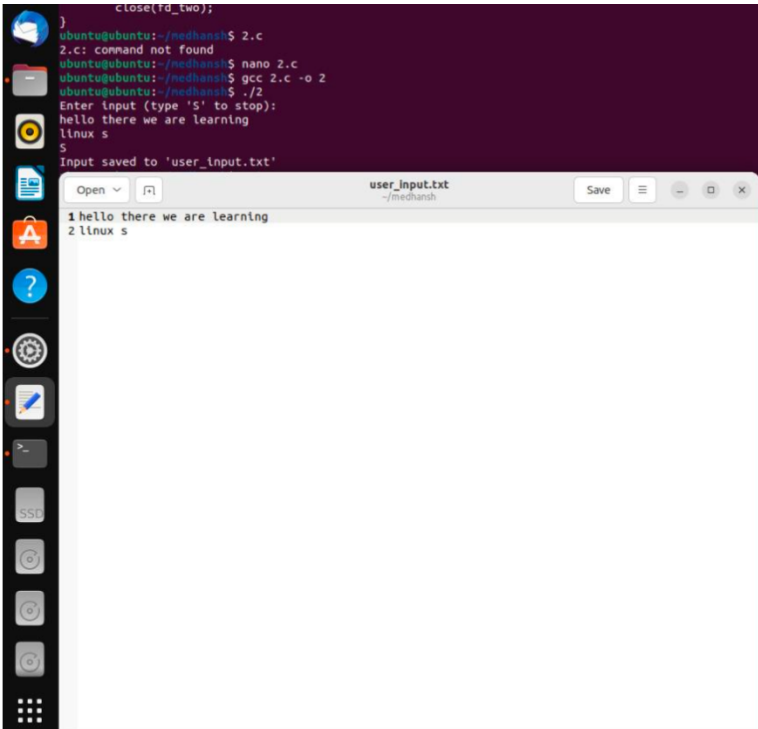
        bytes_written = write(fd, &input, sizeof(input));
        if (bytes_written == -1) {
            perror("Error writing to file");
            close(fd);
            return 1;
        }
    }

    close(fd);
    printf("Input saved to 'user_input.txt'\n");

    return 0;
}
ubuntu@ubuntu:~/medhansh$
```

Output:

```
    close(fd_two);
}
ubuntu@ubuntu:~/medhansh$ 2.c
2.c: command not found
ubuntu@ubuntu:~/medhansh$ nano 2.c
ubuntu@ubuntu:~/medhansh$ gcc 2.c -o 2
ubuntu@ubuntu:~/medhansh$ ./2
Enter input (type 'S' to stop):
hello there we are learning
linux s
$
Input saved to 'user_input.txt'
```



Design a program using system calls to read the contents of a file without using a char array and display the contents directly on the console. Please refrain from using built-in functions like `sizeof()` and `strlen()`.

```
bytes_written = write(fd, &input, sizeof(input));
if (bytes_written == -1) {
    perror("Error writing to file");
    close(fd);
    return 1;
}

close(fd);
printf("Input saved to 'user_input.txt'\n");

return 0;
}

ubuntu@ubuntu:~/medhansh$ nano 3.c
ubuntu@ubuntu:~/medhansh$ gcc 3.c -o 3
ubuntu@ubuntu:~/medhansh$ ./3
Linux is fun and loving!!!
ubuntu@ubuntu:~/medhansh$ cat 3.c
#include <stdio.h>
#include <fcntl.h>
#include <unistd.h>
#include <sys/stat.h>
#include <sys/types.h>

int main() {
    int fd = open("your_file.txt", O_RDONLY);

    if (fd == -1) {
        perror("Error opening file");
        return 1;
    }

    struct stat file_stats;
    if (fstat(fd, &file_stats) == -1) {
        perror("Error getting file size");
        close(fd);
        return 1;
    }

    off_t file_size = file_stats.st_size;
    char placeholder;

    for (off_t i = 0; i < file_size; i++) {
        read(fd, &placeholder, sizeof(char));
        write(STDOUT_FILENO, &placeholder, sizeof(char));
    }

    close(fd);

    return 0;
}
ubuntu@ubuntu:~/medhansh$
```

Output:

```
bytes_written = write(fd, &input, sizeof(input));
if (bytes_written == -1) {
    perror("Error writing to file");
    close(fd);
    return 1;
}

close(fd);
printf("Input saved to 'user_input.txt'\n");

return 0;
}

ubuntu@ubuntu:~/medhansh$ nano 3.c
ubuntu@ubuntu:~/medhansh$ gcc 3.c -o 3
ubuntu@ubuntu:~/medhansh$ ./3
Linux is fun and loving!!!
ubuntu@ubuntu:~/medhansh$ cat 3.c
#include <stdio.h>
#include <fcntl.h>
#include <unistd.h>
#include <sys/stat.h>
#include <sys/types.h>

int main() {
    int fd = open("your_file.txt", O_RDONLY);

    if (fd == -1) {
        perror("Error opening file");
        return 1;
    }

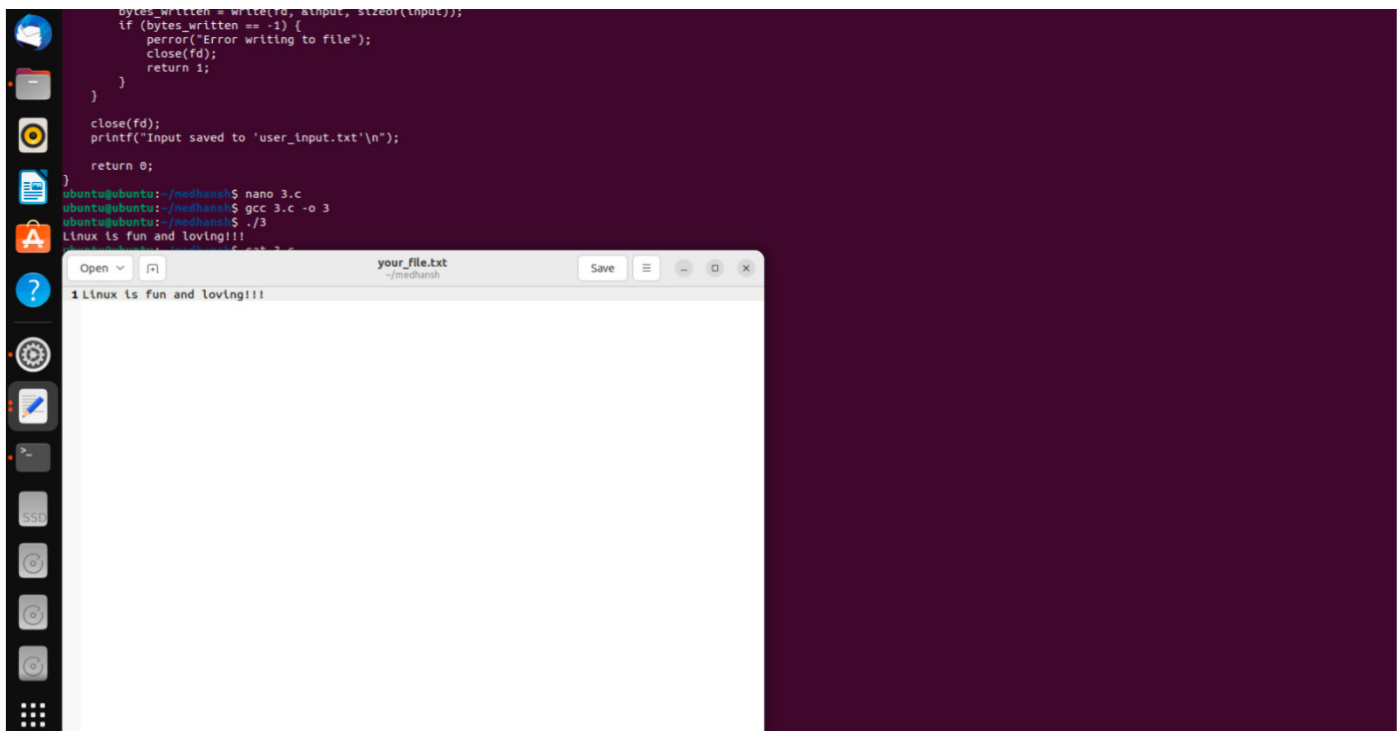
    struct stat file_stats;
    if (fstat(fd, &file_stats) == -1) {
        perror("Error getting file size");
        close(fd);
        return 1;
    }

    off_t file_size = file_stats.st_size;
    char placeholder;

    for (off_t i = 0; i < file_size; i++) {
        read(fd, &placeholder, sizeof(char));
        write(STDOUT_FILENO, &placeholder, sizeof(char));
    }

    close(fd);

    return 0;
}
ubuntu@ubuntu:~/medhansh$
```



Submitted to:  
Ashish Kumar Singh

Submitted by:  
Arvind Singh  
Chinmaye H G  
Daniel  
Manikanta