

# Operating Systems Lab Report

Muhammad Shafeen

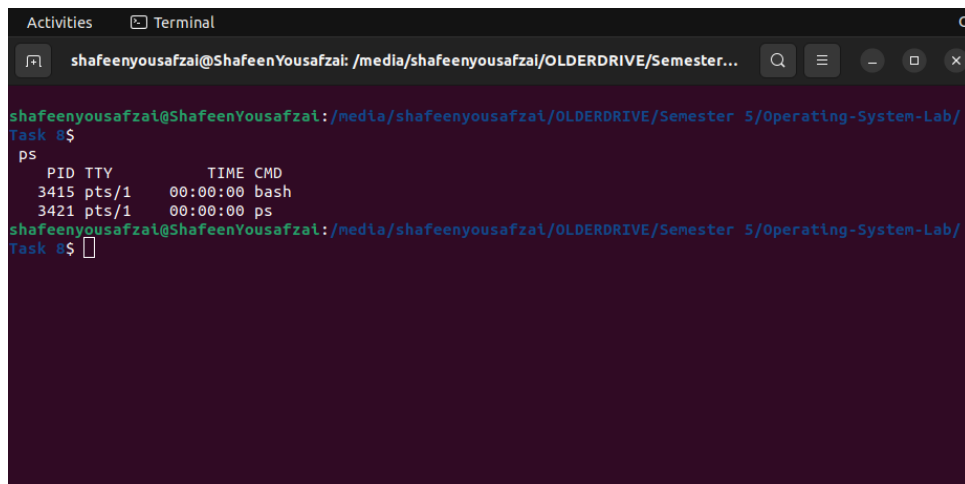
Student ID: 22P-9278

October 11, 2024

## Lab 8: Operating Systems

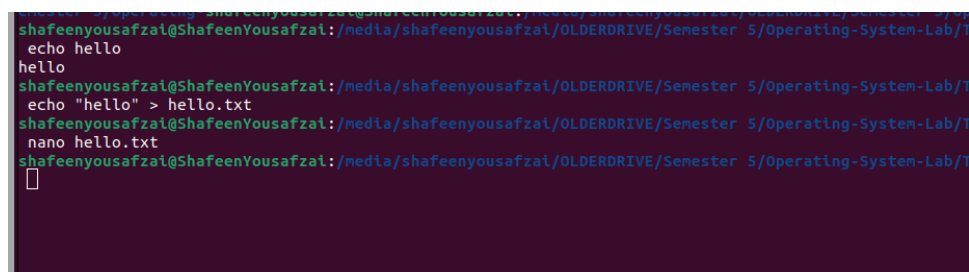
### 0.1 Open 2 Terminals

Opening two terminals checking the process id and then

A terminal window titled 'Terminal' with a search bar and window controls. The prompt is 'shafeenyousafzal@ShafeenYousafzal: /media/shafeenyousafzal/OLDERDRIVE/Semester...'. The user enters 'ps' and the output is displayed as a table with columns PID, TTY, TIME, and CMD. The first row shows PID 3415, TTY pts/1, TIME 00:00:00, and CMD bash. The second row shows PID 3421, TTY pts/1, TIME 00:00:00, and CMD ps.

```
shafeenyousafzal@ShafeenYousafzal: /media/shafeenyousafzal/OLDERDRIVE/Semester...
shafeenyousafzal@ShafeenYousafzal: /media/shafeenyousafzal/OLDERDRIVE/Semester 5/Operating-System-Lab/
Task 8$ ps
ps
  PID TTY          TIME CMD
 3415 pts/1    00:00:00 bash
 3421 pts/1    00:00:00 ps
shafeenyousafzal@ShafeenYousafzal: /media/shafeenyousafzal/OLDERDRIVE/Semester 5/Operating-System-Lab/
Task 8$
```

Figure 1: Checking the process id by using PS

A terminal window showing the user entering 'echo hello', which outputs 'hello'. Then the user enters 'echo "hello" > hello.txt' and 'nano hello.txt'. The prompt is 'shafeenyousafzal@ShafeenYousafzal: /media/shafeenyousafzal/OLDERDRIVE/Semester 5/Operating-System-Lab/T' and the cursor is at the end of the line.

```
shafeenyousafzal@ShafeenYousafzal: /media/shafeenyousafzal/OLDERDRIVE/Semester 5/Operating-System-Lab/T
echo hello
hello
shafeenyousafzal@ShafeenYousafzal: /media/shafeenyousafzal/OLDERDRIVE/Semester 5/Operating-System-Lab/T
echo "hello" > hello.txt
shafeenyousafzal@ShafeenYousafzal: /media/shafeenyousafzal/OLDERDRIVE/Semester 5/Operating-System-Lab/T
nano hello.txt
shafeenyousafzal@ShafeenYousafzal: /media/shafeenyousafzal/OLDERDRIVE/Semester 5/Operating-System-Lab/T

```

Figure 2: Outputting hello on temrinal using cat ;

```

shafeenyousafzal@ShafeenYousafzal:/media/shafeenyousafzal/OLDERDRIVE/Semester 5/Operating-System-Lab/T
echo "hello" > /proc/3415/fd/1
shafeenyousafzal@ShafeenYousafzal:/media/shafeenyousafzal/OLDERDRIVE/Semester 5/Operating-System-Lab/T
nan

```

Figure 3: Outputting hello on the proces id termianl

```

3421 pts/1 00:00:00 ps
shafeenyousafzal@ShafeenYousafzal:/media/shafeenyousafzal/OLDERDRIVE/Semester 5/Operating-S
Task 8$ hello

```

Figure 4: Outputting hello on temrinal using cat ;

```

shafeenyousafzal@ShafeenYousafzal:/media/shafeenyousafzal/OLDERDRIVE/Semester 5/
Operating-System-Lab/Task 8$ ps
PID TTY TIME CMD
3033 pts/1 00:00:00 bash
3185 pts/1 00:00:00 ps
shafeenyousafzal@ShafeenYousafzal:/media/shafeenyousafzal/OLDERDRIVE/Semester 5/
Operating-System-Lab/Task 8$ Muhammad Shafeen
22P-9278
BS-AI

shafeenyousafzal@ShafeenYousafzal:/media/shafeenyousafzal/OLDERDRIVE/Semester 5/
Operating-System-Lab/Task 8$ echo "Muhammad Shafeen"
Muhammad Shafeen
22P-9278
BS-AI
shafeenyousafzal@ShafeenYousafzal:/media/shafeenyousafzal/OLDERDRIVE/Semester 5/
Operating-System-Lab/Task 8$ echo "Muhammad Shafeen"
Muhammad Shafeen
22P-9278
BS-AI" > /proc/3033/fd/1
shafeenyousafzal@ShafeenYousafzal:/media/shafeenyousafzal/OLDERDRIVE/Semester 5/
Operating-System-Lab/Task 8$

```

Figure 5: Outputting Name,Roll,section

## 0.2 Run code and check file size

```

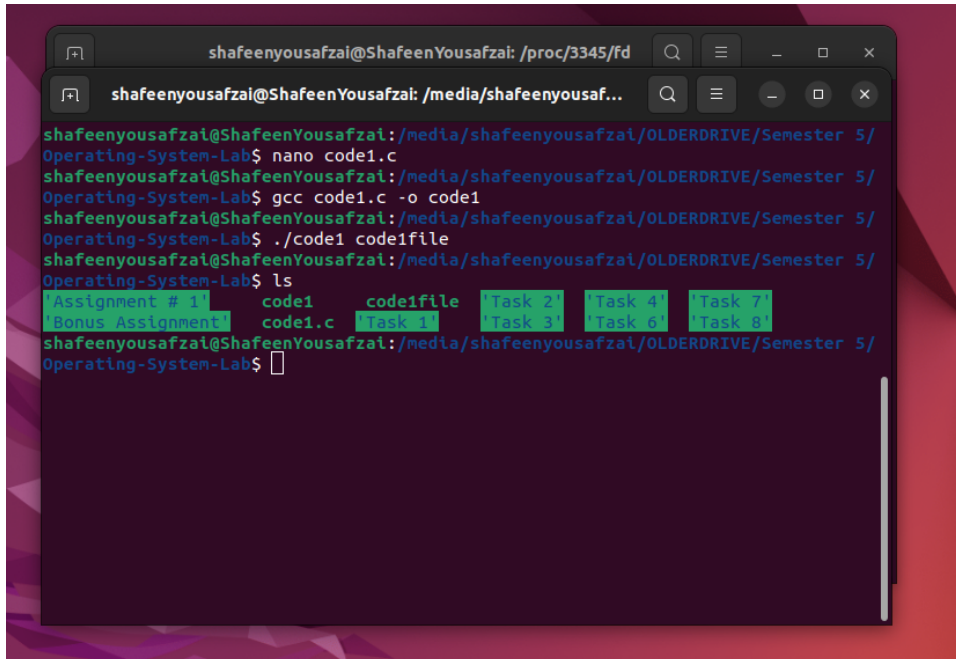
#include <fcntl.h>
#include <stdio.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <unistd.h>
int main(int argc, char* argv[])
{
char *path = argv[1];
int fd = open(path, O_WRONLY | O_EXCL |
O_CREAT);
if (fd == -1)
{
printf("Error: File not Created\n");
return 1;
}

```

```

}
return 0;
}

```



```

shafeenyousafzal@ShafeenYousafzal: /proc/3345/fd
shafeenyousafzal@ShafeenYousafzal: /media/shafeenyousafzal/OLDERDRIVE/Semester 5/
Operating-System-Lab$ nano code1.c
shafeenyousafzal@ShafeenYousafzal: /media/shafeenyousafzal/OLDERDRIVE/Semester 5/
Operating-System-Lab$ gcc code1.c -o code1
shafeenyousafzal@ShafeenYousafzal: /media/shafeenyousafzal/OLDERDRIVE/Semester 5/
Operating-System-Lab$ ./code1 code1file
shafeenyousafzal@ShafeenYousafzal: /media/shafeenyousafzal/OLDERDRIVE/Semester 5/
Operating-System-Lab$ ls
'Assignment # 1'  code1  code1file  'Task 2'  'Task 4'  'Task 7'
'Bonus Assignment'  code1.c  'Task 1'  'Task 3'  'Task 6'  'Task 8'
shafeenyousafzal@ShafeenYousafzal: /media/shafeenyousafzal/OLDERDRIVE/Semester 5/
Operating-System-Lab$

```

Figure 6: Outputting the output of file and creating a file

### 0.3 Question

Question What is the size of the file? Why is it this size?

### 0.4 Answer:

The size of the process is 0 bytes because whenever a new file is created it takes 0 bytes of storage

### 0.5 Question

Check what the `open()` function returns

### 0.6 Answer:

An exit code of 0 indicates success while a non-zero exit code indicates failure. The following failure codes can be returned:

- 1 Error in command line syntax.
- 2 One of the files passed on the command line did not exist.
- 3 A required tool could not be found.
- 4 The action failed.

## 0.7 Close a File

```
#include <fcntl.h>
#include <stdio.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <unistd.h>
int main(int argc, char* argv[])
{
    if (argc != 2)
    {
        printf("Error: Run like this: ");
        printf("%6s name-of-new-file\n", argv[0]);
        return 1;
    }
    char *path = argv[1];
    int i = 0;
    while(i<2)
    {
        int fd = open(path, O_WRONLY | O_CREAT);
        printf("Created! Descriptor is %d\n", fd);
        close(fd);
        i++;
    }
    return 0;
}
```

## 0.8 run without closing file

```
#include <fcntl.h>
#include <stdio.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <unistd.h>
int main(int argc, char* argv[])
{
    if (argc != 2)
    {
        printf("Error: Run like this: ");
        printf("%6s name-of-new-file\n", argv[0]);
        return 1;
    }
    char *path = argv[1];
    int i = 0;
    while(i<2)
    {
        int fd = open(path, O_WRONLY | O_CREAT);
        printf("Created! Descriptor is %d\n", fd);
        i++;
    }
}
```

```

shafeenyousafzai@ShafeenYousafzai: /media/shafeenyousafzai/...
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/
Operating-System-Lab/Task 8$ ./code2 test10
Created! Descriptor is 3
Created! Descriptor is 3
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/
Operating-System-Lab/Task 8$ ls
code1.c      'io-operations-guide(1).pdf'  LabTask8.pdf  test6
code2        'io-operations-guide(2).pdf'  test          test7
code2.c      'io-operations-guide.pdf'    test1         test8
code2file.c  'IPC Message Queue.pdf'     test10        test9
code2file.txt 'IPC - Shared Memory.pdf'   test2         text.c
filedescriptors.pdf Lab-08.pdf                test3
Hello        Lab_8_manual-1.pdf         test4
hello.txt    Lab_8_manual.pdf          test5
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/
Operating-System-Lab/Task 8$ nano code2.c
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/
Operating-System-Lab/Task 8$ gcc code2.c -o code2withoutfd
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/
Operating-System-Lab/Task 8$ ./code2withoutfd filewithoutfd
Created! Descriptor is 3
Created! Descriptor is 4
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/
Operating-System-Lab/Task 8$

```

Figure 7: Running the code without fd(close)

```

}
return 0;
}

```

## 0.9 Question :

Writing to a file is done using the write call. To write, we should obviously open a file first.

## 0.10 Answer :

we are getting 3 because one file is being run and we get a 4 because we did not close the file created before with a return descriptor of 3 .

## 0.11 Writing to a file

```

#include <fcntl.h>
#include <stdio.h>
#include <string.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <time.h>
#include <unistd.h>
char* get_timeStamp()
{
time_t now = time(NULL);
return asctime(localtime(&now));
}

```

```
int main(int argc, char* argv[])
{
    char *filename = argv[1];
    char *timeStamp = get_timeStamp();
    int fd = open(filename, O_WRONLY |
    O_APPEND |
    O_CREAT, 0666);
    size_t length = strlen(timeStamp);
    write(fd, timeStamp, length);
    close(fd);
    return 0;
}
```

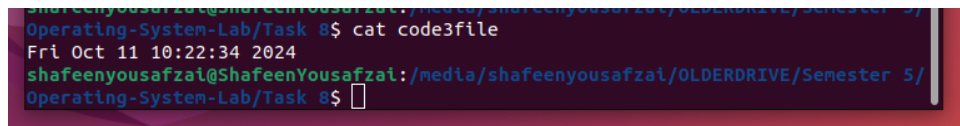


Figure 8: Output of the code. print the output

### 0.12 Question :

What is 0666 that is specified in the open() call? What does it mean?

### 0.13 Answer :

0666 is the file permission mode in octal notation. It means the file will be created with read and write permissions for the owner, group, and others.

### 0.14 Question :

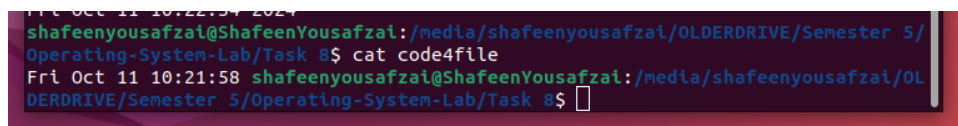
What is O\_APPEND doing in the same call? Run the program again and check its output.

### 0.15 Answer :

O\_APPEND flag ensures that the data is appended to the end of the file. If you run the program multiple times, you'll see multiple timestamps in the file, one after another.

### 0.16 Question :

Modify the following line in the code and then compile and run the program and check its output. What has happened? From: `size_t length = strlen(timeStamp);` To: `size_t length = strlen(timeStamp)-5`



```

Fri Oct 11 10:22:57 2024
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/
Operating-System-Lab/Task 8$ cat code4file
Fri Oct 11 10:21:58 shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OL
DERDRIVE/Semester 5/Operating-System-Lab/Task 8$ 

```

Figure 9: Out of the file with -5 timestamp

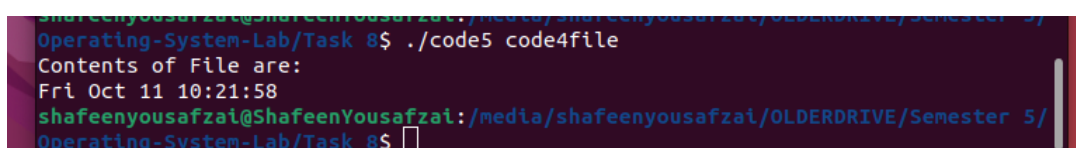
### 0.17 Answer :

By reducing the length by 5, you're truncating the last 5 characters of the times- tamp. This will likely cut off the year and newline character from the timestamp in the file.

```

#include <stdio.h>
#include <sys/stat.h>
#include <sys/types.h>
#include <unistd.h>
int main(int argc, char* argv[])
{
    if (argc != 2)
    {
        printf("Error: Run like this: ");
        printf("%6s name-of-existing-file\n",
            argv[0]);
        return 1;
    }
    char *path = argv[1];
    int fd = open(path, O_RDONLY);
    if (fd == -1)
    {
        printf("File does not exist\n");
        return 1;
    }
    char buffer[200];
    read(fd, buffer, sizeof(buffer)-1);
    printf("Contents of File are:\n");
    printf("%s\n", buffer);
    close(fd);
    return 0;
}

```



```

shafeenyousafzai@shafeenyousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/
Operating-System-Lab/Task 8$ ./code5 code4file
Contents of File are:
Fri Oct 11 10:21:58
shafeenyousafzai@ShafeenYousafzai:/media/shafeenyousafzai/OLDERDRIVE/Semester 5/
Operating-System-Lab/Task 8$ 

```

Figure 10: Output of the last code file