

Linux Programming

Lab 6 | 19-02-2020

Name: j.sanjana reddy

Reg:17mis1012

System Calls For File Operations:

1.System calls for file open,read,write

Code:

open():

```
#include<stdio.h>
#include<fcntl.h>
#include<errno.h>
extern int errno;
int main()
{
    int fd = open("file1.txt", O_RDONLY);
    printf("fd = %d\n", fd);
    if (fd == -1)
    {
        printf("Error Number % d\n", errno);
        perror("Program");
    }
    return 0;
}
```

Output:

```
linux@sanjana:~$ gcc sanjana.c
linux@sanjana:~$ ./a.out
fd=3
```


Read & Write:

Code:

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

int main()
{
    int fd;
    fd=open("/home/sanjana/lab6/file1.txt",O_RDWR);
    char word[50];
    read(fd,word,sizeof(word));
    const char *buf="Okay Read from file1 and written to file2";
    ssize_t nr;
    nr=write(fd,buf,strlen(buf));
}
```

Output:



```
linux@sanjana:~$ cat sanjana.txt
hello world.....
```

2. Manage EINTR while accessing file using system calls.

Code:

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

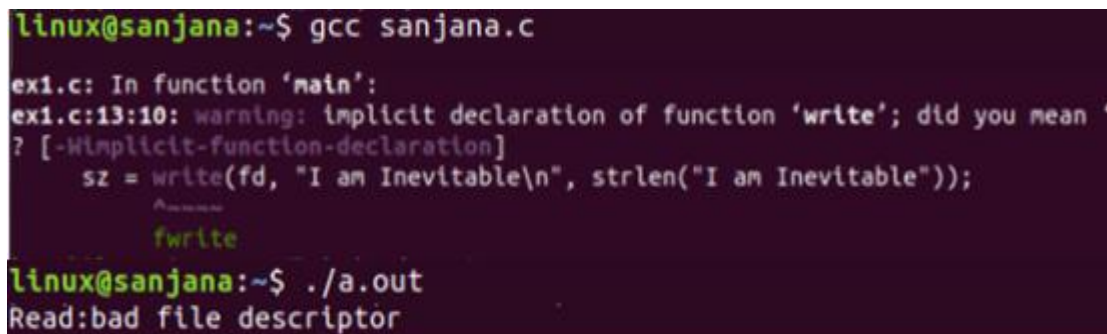
```

#include<fcntl.h>s
#include<errno.h>
#include<stdlib.h>
#include<string.h>
int main()
{

int fd = open("file1.txt", O_RDONLY );
int sz;
sz = write(fd, "I am Inevitable\n", strlen("I am Inevitable"));
if (sz == -1 && errno != EINTR)
{
perror("Read");
exit(EXIT_FAILURE);
}
return 0;
}

```

Output:



```

linux@sanjana:~$ gcc sanjana.c
ex1.c: In function 'main':
ex1.c:13:10: warning: implicit declaration of function 'write'; did you mean 'fwrite' [-Wimplicit-function-declaration]
    sz = write(fd, "I am Inevitable\n", strlen("I am Inevitable"));
           ^~~~~~
           fwrite
linux@sanjana:~$ ./a.out
Read:bad file descriptor

```

3. Do Non-Block read and write using system calls.

Code:

```

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

int main()
{
int fd,ret;
fd=open("/home/sanjana17mis1012/file1.txt",O_RDWR);
ssize_t nr;
char buf[BUFSIZ];
start:
nr=read(fd,buf,BUFSIZ);
while(BUFSIZ!=0 && (ret = read(fd,buf,BUFSIZ))!=0)

```

```

{
    if(nr==-1)
    {
        if(errno == EINTR)
        {
            goto start;
        }
        if(errno == EAGAIN)
        {
            continue;
        }
        else
        {
            perror("Read");
            break;
        }
    }
}
}

```

Output:

```

linux@sanjana:~$ gcc sanjana2.c
linux@sanjana:~$ ./a.out
Read:bad file Descriptor

```

File Permissions:

4. Disable Write permissions to user for all the files in specific folder.

Code:

So let's create a directory called "acldemo" and create two files namely file1.txt,file2.txt in it.

Changing permissions to r-x i.e, disabling write permissions.

```

linux@sanjana:~$ ls
sanjana1.txt  sanjana2.txt

```

```

linux@sanjana:~$ setfacl -mu:sanjana:r-x /home/sanjana/desktop
linux@sanjana:~$ getfacl /home/sanjana/desktop/*
linux@sanjana:~$ #owner:sanjana
linux@sanjana:~$ user::rw-
linux@sanjana:~$ #group:sanjana
linux@sanjana:~$ other::r--

```

5. Set write permission for only one user on a file

```
linux@sanjana:~$ setfacl -mu:sanjana:r-x /home/sanjana/desktop
linux@sanjana:~$ getfacl /home/sanjana/desktop/*
linux@sanjana:~$ setfacl -m u:sanjana:-w-file3.txt
linux@sanjana:~$ #owner:sanjana
linux@sanjana:~$ user::rw-
linux@sanjana:~$ #group:sanjana
linux@sanjana:~$ other::r--
```