# **ASG 06**

**Name:** Yatharth Thakare

**Roll No:** 51

**PRN:** 12111403

**Subject:** OS

**Title:**

1. **Process system call**

**Code:**

#include<stdio.h>

#include<stdlib.h>

#include<unistd.h>

#include<fcntl.h>

#include<string.h>

#include<sys/wait.h>

void processSystemCall(){

pid\_t childPid;

int status;

childPid = fork();

if(childPid<0){

perror("Child Process not created. Error!");

}

else if(childPid == 0)

{

printf("Child Process PID: %d\n",getpid());

exit(0);

}

else{

printf("Parent Process: PID - %d\n",getppid());

}

}

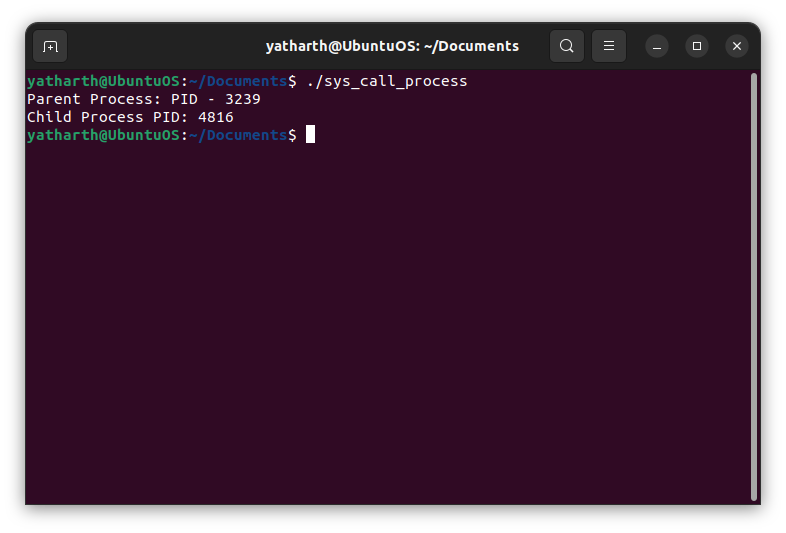
int main(){

processSystemCall();

return 0;

}

**Output:**

****

1. **File related system call**

**Code:**

#include<stdio.h>

#include<stdlib.h>

#include<unistd.h>

#include<fcntl.h>

#include<string.h>

#include<sys/wait.h>

void fileSysCall(){

int fd;

char buffer[100];

ssize\_t bytesRead,bytesWritten;

// open call

fd = open("hello.txt",O\_CREAT|O\_RDWR,S\_IRUSR,S\_IWUSR);

if(fd<0){

perror("can't open");

return;

}

//write call

bytesWritten = write(fd,"Hello World!",strlen("Hello World"));

if(bytesWritten < 0){

perror("can't write");

return;

}

//read call

lseek(fd,0,SEEK\_SET);

bytesRead = read(fd,buffer,sizeof(buffer));

if(bytesRead<0){

perror("can't read");

return;

}

buffer[bytesRead] = '\0';

printf("Read from file: %s\n",buffer);

//close call

close(fd);

//unlink

unlink("hello.txt");

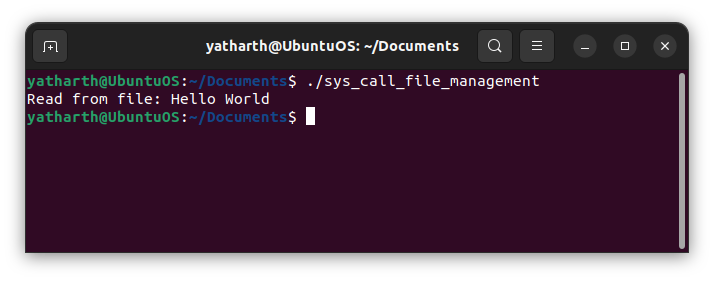
}

int main(){

fileSysCall();

}

**Output:**

****

1. **File permission related system call**

**Code:**

#include<stdio.h>

#include<stdlib.h>

#include<unistd.h>

#include<fcntl.h>

#include<string.h>

#include<sys/wait.h>

#include<sys/stat.h>

void fileSecurity(){

const char \*filename = "example.txt";

umask(0);

int fd = open(filename,O\_CREAT|O\_RDWR, S\_IRUSR|S\_IWUSR);

if(fd==-1){

perror("open");

return;

}

struct stat file\_stat;

if(stat(filename,&file\_stat)==-1){

perror("stat");

close(fd);

return;

}

printf("File permission before chmod %o\n",file\_stat.st\_mode & 0777);

int ch = chmod(filename,S\_IRUSR|S\_IWUSR|S\_IRGRP|S\_IROTH);

if(ch == -1){

perror("chmod");

close(fd);

return;

}

struct stat file\_stat2;

if(fstat(fd,&file\_stat2)==-1){

perror("fstat");

close(fd);

return;

}

printf("FILE permission after chmod: %o\n",file\_stat2.st\_mode & 0444);

close(fd);

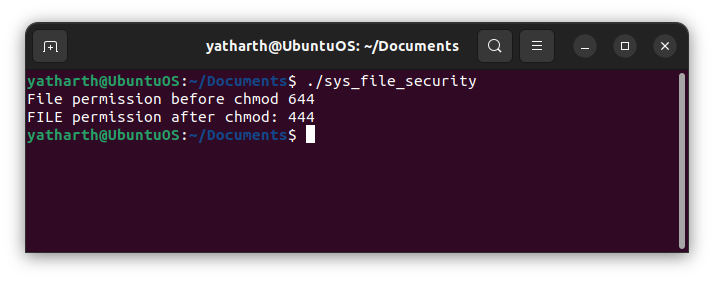
}

int main(){

fileSecurity();

}

**Output:**

****