S.No: 1	Exp. Name: ***Execute UNIX system call for File management.***	Date:2025-04-09

ID: 2300331530046

Aim:

Aim: To Execute UNIX system call for File management.

Description:

The file structure related system calls available in the UNIX system let you create, open, and close files, read and write files, randomly access files, alias and remove files, get information about files, check the accessibility of files, change protections, owner, and group of files, and control devices. These operations either use a character string that defines the absolute or relative path name of a file, or a small integer called a file descriptor that identifies the I/O channel.

There are four system calls for file management:

1. open ()
2. read ()
3. write ()
4. close ()
5.

open ():

open() system call is used to know the file descriptor of user-created files. Since read and write use file descriptor as their 1st parameter so to know the file descriptor open() system call is used.

Syntax:

fd = open (file_name, mode, permission);.

Example:

fd = open ("file", O_CREAT | O_RDWR, 0777);

Here,

- file_name is the name to the file to open.
- mode is used to define the file opening modes such as create, read, write modes.
- permission is used to define the file permissions.

Return value: Function returns the file descriptor.

read ():

read() system call is used to read the content from the file. It can also be used to read the input from the keyboard by specifying the 0 as file descriptor (see in the program given below).

Syntax:

length = read(file_descriptor , buffer, max_len);

Example:

n = read(0, buff, 50);

Here,

- file_descriptor is the file descriptor of the file.
- buffer is the name of the buffer where data is to be stored.
- max_len is the number specifying the maximum amount of that data can be read

Return value: If successful read returns the number of bytes actually read.

write ():

write() system call is used to write the content to the file.

Syntax:
length = write(file_descriptor , buffer, len);

Example:
n = write(fd, "Hello world!", 12);

Here,

- file descriptor is the file descriptor of the file.
- buffer is the name of the buffer to be stored.
- len is the length of the data to be written.
-

Return value: If successful write() returns the number of bytes actually written.

close ():
close() system call is used to close the opened file, it tells the operating system that you are done with the file and close the file.

Syntax:
int close(int fd);
Here,

- fd is the file descriptor of the file to be closed.

Return value: If file closed successfully it returns 0, else it returns -1.

Source Code:

fileManagement.c

```
#include<unistd.h>
#include<fcntl.h>
#include<sys/stat.h>
#include<stdio.h>
int main()
{
   int n,fd;
   char buff[50];
   printf("Enter text to write in the file:\n");
   n=read(0,buff,50);
   fd=open("file",O_CREAT|O_RDWR,0777);
   write(fd,buff,n);
   write(1,buff,n);
   int close(int fd);
   return 0;
}
```

Execution Results - All test cases have succeeded!

Test Case - 1
User Output
Enter text to write in the file: Hello world, welcome @ IncludeHelp
Hello world, welcome @ IncludeHelp

ID: 2300331530046
2023-27-CSE-AIML-A
Raj Kumar Goel Institute Of Technology

Test Case - 2
User Output
Enter text to write in the file: Welcome to the operating system lab
Welcome to the operating system lab