MOUNT CARMEL COLLEGE AUTONOMOUS

(Affiliated to Bengaluru City University)

Palace Road, Bengaluru - 560 052

ASSIGNMENT

On

UNIX PROGRAMMING

Submitted by

AYUSHI GUPTA MS194412

SUMITHRA M MS194440

I V SEMESTER BCA : 2020-2021

Under the Guidance of

SUSHMA MARGARET A

Lecturer

[Dept. of CS]

TABLE OF CONTENTS

TOPIC 1

UNIX SYSTEM CALLS

1.1 Understanding UNIX System Calls

A system call is just what its name implies - a request for the operating system to do something on behalf of the user's program. The system calls are functions used in the kernel itself. To the programmer, the system call appears as a normal C function call. However since a system call executes code in the kernel, there must be a mechanism to change the mode of a process from user mode to kernel mode. The C compiler uses a predefined library of functions (the C library) that have the names of the system calls. The library functions typically invoke an instruction that changes the process execution mode to kernel mode and causes the kernel to start executing code for system calls. The instruction that causes the mode change is often referred to as an "operating system trap" which is a software generated interrupt. In simple words interface between a process and an operating system is provided by system calls. In general, system calls are available as assembly language instructions. They are also included in the manuals used by the assembly level programmers.

System calls in UNIX are used for **file system control, process control, interprocess communication** etc. Access to the UNIX kernel is only available through these system calls. System calls are the only entry points into the kernel system. Generally, system calls are similar to function calls, the only difference is that they remove the control from the user process.

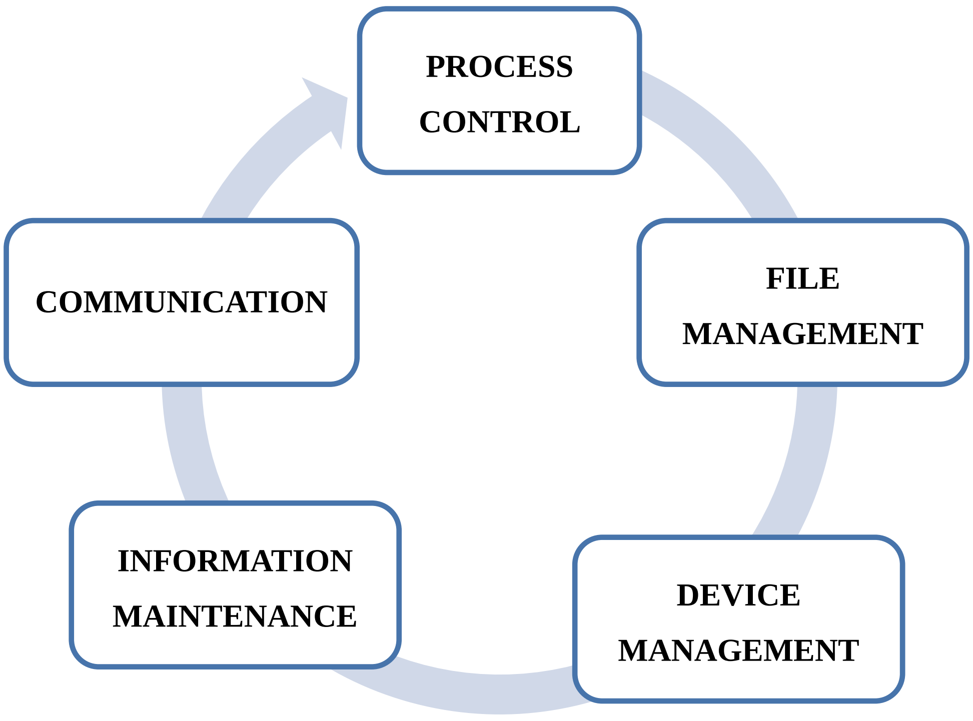
The UNIX system interface consists of about **80 system calls** (as UNIX evolves this number will increase).

SERVICES PROVIDED BY SYSTEM CALLS :

1. Process creation and management
2. Main memory management
3. File Access, Directory and File system management
4. Device handling(I/O)
5. Protection
6. Networking, etc.

1.2 Types of System Calls

There are mainly five types of system calls. These are explained in detail as follows –



1. Process Control

These system calls deal with processes such as process creation, process termination etc.

2. **File Management**

These system calls are responsible for file manipulation such as creating a file, reading a file, writing into a file etc.

3. **Device Management**

These system calls are responsible for device manipulation such as reading from device buffers, writing into device buffers etc.

4. **Information Maintenance**

These system calls handle information and its transfer between the operating system and the user program.

5. **Communication**

These system calls are useful for interprocess communication. They also deal with creating and deleting a communication connection.

1.3 Common and well known System Calls

- access: checks if calling process has file access
- alarm: sets a process's alarm clock
- chdir: changes the working directory
- chmod: changes the mode of a file
- chown: changes the ownership of a file
- chroot: changes the root directory
- close: closes a file descriptor
- dup, dup2: duplicates an open file descriptor
- execl, execv, execle, execve, execlp, execvp: executes a file
- exit: exits a process
- fcntl: controls open files
- fork: creates a new process
- getpid, getpgrp, getppid: gets group and process IDs

- getuid, geteuid, getgid, getegid: gets user and group IDs
- ioctl: controls character devices
- kill: sends a signal to one or more processes
- link: links a new file name to an existing file
- lseek: moves read/write file pointer
- mknod: makes a directory, special or ordinary file
- mount: mounts a filesystem
- msgctl, msgget, msgsnd, msgrcv: message passing support
- nice: changes priority of a process
- open: opens a file for reading or writing
- pause: suspends a process until a signal occurs
- pipe: creates an interprocess pipe
- plock: locks a process in memory
- profil: requests an execution profile
- ptrace: allows a process to trace the execution of another
- read: reads from a file
- semctl, semget, semop: semaphore support
- setpgrp: sets process group ID
- setuid, setgid: sets user and group IDs
- shmctl, shmget, shmop: shared memory support
- signal: control of signal processing
- sleep: suspends execution for an interval
- stat, fstat: gets file status
- stime: sets the time
- sync: updates the super block
- time: number of seconds since 1/1/1970
- times: gets process and child process times
- ulimit: gets and sets user limits
- umask: gets and sets file creation mask

- umount: unmounts a file system
- uname: gets system information
- unlink: removes directory entry
- ustat: gets file system statistics
- utime: sets file access and modification times
- wait: waits for a child process to stop or terminate
- write: writes to a file

1.4 Purpose of some System Calls

open()

The open() system call is used to provide access to a file in a file system. This system call allocates resources to the file and provides a handle that the process uses to refer to the file. A file can be opened by multiple processes at the same time or be restricted to one process. It all depends on the file organization and file system.

read()

The read() system call is used to access data from a file that is stored in the file system. The file to read can be identified by its file descriptor and it should be opened using open() before it can be read. In general, the read() system calls takes three arguments i.e. the file descriptor, the buffer which stores read data and the number of bytes to be read from the file.

write()

The write() system call writes the data from a user buffer into a device such as a file. This system call is one of the ways to output data from a program. In general, the write() system calls takes three arguments i.e. the file descriptor,the pointer to the buffer where data is stored and the number of bytes to write from the buffer.

close()

The close() system call is used to terminate access to a file system. Using this system call means that the file is no longer required by the program and so the buffers are flushed, the file metadata is updated and the file resources are de-allocated.

wait()

In some systems, a process may wait for another process to complete its execution. This happens when a parent process creates a child process and the execution of the parent process is suspended until the child process executes. The suspending of the parent process occurs with a wait() system call. When the child process completes execution, the control is returned back to the parent process.

This system call runs an executable file in the context of an already running process. It replaces the previous executable file. This is known as an overlay. The original process identifier remains since a new process is not created but data, heap, stack etc. of the process are replaced by the new process.

fork()

Processes use the fork() system call to create processes that are a copy of themselves. This is one of the major methods of process creation in operating systems. When a parent process creates a child process and the execution of the parent process is suspended until the child process executes. When the child process completes execution, the control is returned back to the parent process.

exit()

The exit() system call is used by a program to terminate its execution. In a multithreaded environment, this means that the thread execution is complete. The operating system reclaims resources that were used by the process after the exit() system call.

kill()

The kill() system call is used by the operating system to send a termination signal to a process that urges the process to exit. However, kill() system call does not necessarily mean killing the process and can have various meanings.

1.5 Errors during System Calls

If an error occurs during a system call the function may return an error value.

When a system call discovers and error, it returns -1 and stores the reason the called failed in an external variable named "errno". The "/usr/include/errno.h" file maps these error numbers to manifest constants, and it these constants that you should use in your programs.

When a system call returns successfully, it returns something other than -1, but it does not clear "errno". "errno" only has meaning directly after a system call that returns an error.

When you use system calls in your programs, you should check the value returned by those system calls. Furthermore, when a system call discovers an error, you should use the "perror()" subroutine to print a diagnostic message on the standard error file that describes why the system call failed. The syntax for "perror()" is:

```
void perror(string)
char string;
```

****

TOPIC 2

SHELL SCRIPT PROGRAMMING

2.1 Shell Script to count vowels, symbols, lines, blank spaces and characters in file.

Problem Statement : Write a Shell Script that, given a filename as the argument will count vowels, blank spaces, characters, number of lines and symbols.

Working : This shell script program accepts a filename which is given as passed as an argument and performs the operation that counts the number of vowels, characters, blank spaces, lines and symbols.

Commands Used : In this program in order to achieve the solution **grep, wc, cat** commands are used.

- grep : It will globally search a pattern and prints it.
- wc : It stands for word count, and also it has several options with it.
- cat file_name : It displays the content of the file.

Source Code :

```
vi Program1
------------------------------------------------------------------------------------------
 if [ $# -ne 1 ]
 then
    echo "Pass only one file"
    exit
```

```
fi


if [ -f $1 ]
then
  echo "------------------------------------------------------------------"
  echo "      The Contents of the file $1 is :  "
  echo "------------------------------------------------------------------"
  cat $1
else
  echo "File does not Exists"
  exit
fi


char=`cat $1 | grep -o "[aA-zZ]" | wc -l`
line=`cat $1 | wc -l`
vow=`cat $1 | grep -o "[aAeEiIoOuU]" | wc -l`
space=`cat $1 | grep -o " " | wc -l`
symbol=`cat $1 | grep -o "[!?:;'.,]" | wc -l`


echo "=========================================================="
echo "Vowels       : $vow"
echo "Blank Spaces  : $space"
echo "Characters    : $char"
echo "Lines        : $line"
```

```
 echo "Symbols       : $symbol"

echo "============================================================="
```

Sample Input/Output 01 :

```
localhost:~$ sh Program1 simple
------------------------------------------------------------------
       The Contents of the file simple is :
------------------------------------------------------------------
apple :
orange ,
.
==================================================================
Vowels         : 5
Blank Spaces   : 2
Characters     : 11
Lines          : 3
Symbols        : 3
==================================================================
```

Sample Input/Output 02 :

```
localhost:~$ sh Program1 europe
File does not Exists
```

```
===========================================
```

Sample Input/Output 03 :

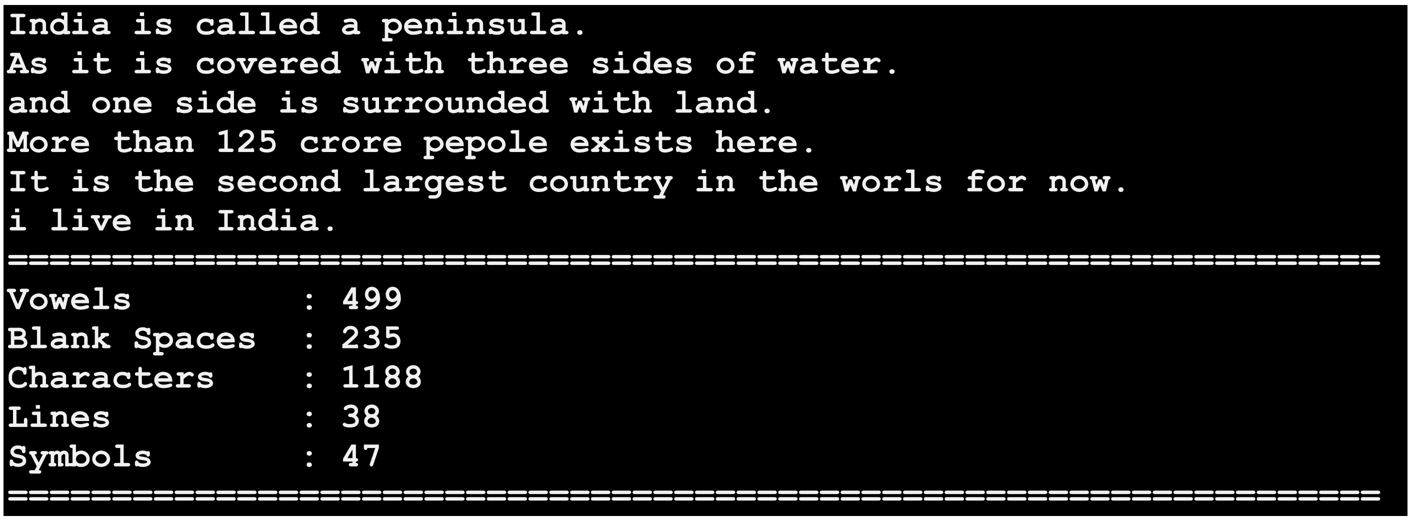
```
localhost:~$ sh Program1
Pass only one file
```

Sample Input/Output 04 :

```
localhost:~$ sh Program1 India europe
Pass only one file
```

Sample Input/Output 05 :

```
localhost:~$ sh Program1 India
------------------------------------------------------------------
       The Contents of the file India is :
------------------------------------------------------------------
The name of my country is India.
It lies in the continent of Asia.
It got independence on 15th August 1947.
India is famous country all over the world.
The national Language of India is 'Hindi'.
Regional Language : Tamil.
India ia beautiful Country!
Currency of India is rupee.
Capital of India is New Delhi.
The National Bird of India is peacock.
The National animal : Tiger , National flower : Lotus;
People of India are very kind.
The people of my country are very honest.
India is a agricultural country.
I love ny country very much.
I am very proud to be INDIAN!
India is a country , in which different languages and religions are
 co existing
National flag consists of three colors.
father of India Nation is 'Mahatma Gandhi'.
There are many famous personalities from INDIA.
Current PM of INDIA : NARENDRA MODI.
Here people of all religious live with peace and ethusiam.
It is the land of festivals.
More then 50 festivals are celebrated.
Both religious and National festivals are  celebrated.
INDIA got its name from the river INDUS.
Cricket and hockey are the famous sports in india.
NATIONAL SPORT OF INDIA IS HOCKEY!
The people here are called INDIANS or HINDUSTANI.
HOME of TAJ MAHAL which is one of the seven wonders of the worls.
Mumbai is the largest city.
Goa is the smallest city.
```



```
India is called a peninsula.
As it is covered with three sides of water.
and one side is surrounded with land.
More than 125 crore pepole exists here.
It is the second largest country in the worls for now.
i live in India.
==================================================================
Vowels         : 499
Blank Spaces   : 235
Characters     : 1188
Lines          : 38
Symbols        : 47
==================================================================
```

****