

Learning Report – LINUX OS and Programming

Course Code: <CODE>

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Ver. Rel. No.** | **Release Date** | **Prepared. By** | **Reviewed By** | **Approved By** | **Remarks/Revision Details** |
|  | 2-03-2021 | P Saikeerthi |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**Document History**

# 

Contents

[Introduction to Linux 3](#_Toc65909878)

[Concepts learnt as part of Activity 1 are as follows: 3](#_Toc65909879)

[**Activity 1** 4](#_Toc65909880)

[Case Study - Design & Link with Libraries 4](#_Toc65909881)

[**Activity 3 - Threads** 9](#_Toc65909882)

[**Activity 4 – Process** 10](#_Toc65909883)

[**Activity 5 – Semaphores and mutex** 10](#_Toc65909884)

Github Link: https://github.com/99003539/Linux\_Task\_1

# Introduction to Linux

Linux is a family of open source Unix like operating system based on Linux kernel.

## Concepts learnt as part of Activity 1 are as follows:

1. Understanding Tool chain
2. GNU Tools
3. Makefile
4. Static Libraries & Linking
5. Dynamic Libraries & Linking
6. Static Analysis of Code
7. Debugging Tool

Software or packages required are:

1. GNU tools
2. Vaalgrind
3. Make
4. Git

## **Activity 1**

## Case Study - Design & Link with Libraries

This case study has the following requirements:

* Source folder containing the source files of mystring, myutils, bit\_set\_reset
* Header folder containing the header files of mystring, myutils, bit\_set\_reset
* Finally, the test file

Functions in mystring file:

* Mystrlen
* Mystrcpy
* Mystrcat
* mystrcmp

Functions in myutils file:

* Factorial
* isPrime
* isPalindrome
* vsum

Functions in bit\_set-reset:

* Set
* Reset
* Flip
* Query
* Generated Makefile

Command to generate makefile

Build: test.c Mystring.c myutils.c bit\_set\_reset.c

gcc test.c Mystring.c myutils.c bit\_set\_reset.c

run: ./a.out

* Static library generation

Commands to generate static library file are as follows:

gcc filename.c -c

gcc filename.c -c

ar rc libsimple.a fiename.o

gcc test.c -c

gcc -L.test.o -o filename.out -lsimple

gcc -L.test.o -o filename.out –lsimple -static

* Dynamic library generation

Commands to generate Dynamic library are as follows:

gcc filename.c-c

gcc filename.c-c

gcc -shared-o libsample.so filename.os filename.o

gcc test.c-c

gcc -L.test.o-o d1.out-lsample

LD\_LIBRARY\_PATH=../d1.out

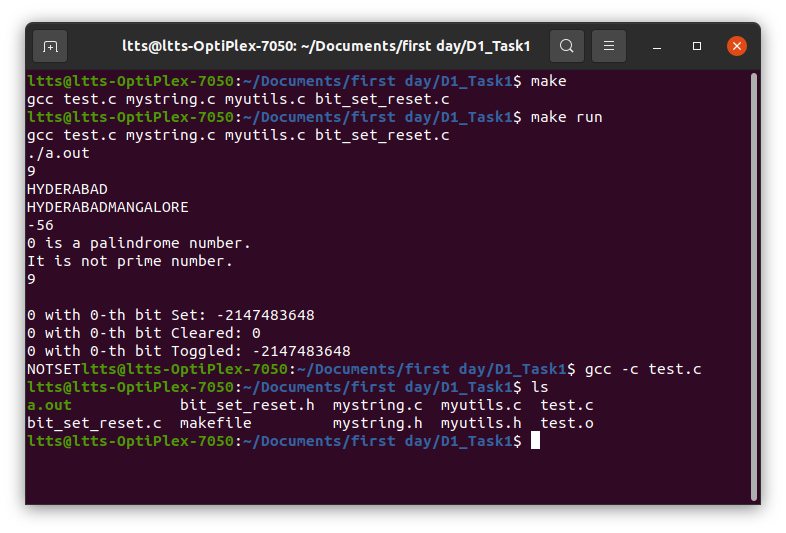


Fig1: Output for the program

**Activity 2- System Calls and Signals**

System calls provide interfacing between the process and the operating system.

**Description**

1. Write a program to copy one file contents to other using open, read, write, close system calls (like cp command, which takes source, destination files as cmd line args).

* Open: Used to open the file for reading or writing.
* Read: Reads From the file indicated by the file descriptor fd, the read() function reads cnt bytes of input into the memory area indicated by buf. A successful read() updates the access time for the file.
* fd: file descriptor
* buf: buffer to read data from
* cnt: length of buffer
* O\_RDONLY: read only, O\_WRONLY: write only, O\_RDWR: read and write, O\_CREAT: create file if it doesn’t exist, O\_EXCL: prevent creation if it already exists
* close: Tells the operating system that it is done with a file descriptor so it can Close the file which pointed by fd.
* Write: Writes cnt bytes from buf to the file or socket associated with fd. cnt should not be greater than INT\_MAX (defined in the limits.h header file). If cnt is zero, write() simply returns 0 without attempting any other action.

**Learning outcome**

Understood the usage of the system calls and was able to produce the output for above case study.

**Challenges**

Initially faced difficulty in using the open, close, write, read system calls.

2. Write a program to count number of lines, words, characters in given file (like wc command)

**Description**

To count number of lines, word, characters in the given file alpha.txt.

To execute this close, read and open systems calls are used.

**Learning outcome**

Understood the usage of the system calls and was able to produce the output for above case study

3. Write a program to send specific signal to a target process (with given id, like kill command)

**Description**

Here we send the specific signal to the target process to terminate that process

* We find the process Id of the process which must be terminated, and use kill command to do so.

**Learning outcome**

Understood the process of using commands to terminate the process without closing the terminal and was able to generate the output.

**Challenges**

Faced difficulty in understanding where when and how to use this command.

## **Activity 3 - Threads**

Parallel execution path for same process.

1.Write a program to find min/max element from large array (1000 data points) using parallel computations (multithreading).

**Description**

Declared an array and initialized the same to achieve the above task.

**Challenges**

Was not able to understand the question.

2.Write a program to print current time periodically.

**Description**

Time is a command used to print the time in hours, minutes and seconds.

## **Activity 4 – Process**

Program in execution is called as process. When the grogram is loaded into the memory it becomes process, this memory is divided as heap, stack, data, text.

1. Design a mini shell

2. Write a program to compile & link any c/c++ program within child process by launching gcc using execl/execlp.

**Description**

**Execlp**: duplicates the actions of the shell in searching for an executable file.

**Fork: T**his system call is used to create new process.

**Challenges**

Was not able to understand the usage of command.

## **Activity 5 – Semaphores and mutex**

Concepts learnt as part of activity 5 are as follows:

1. Mutex lock
2. Semaphores – named and unnamed
3. Race condition
4. Deadlock
5. Pipes
6. Shared memory
7. Message queue

Learning outcome:

* Understood the requirement of and usage of all the above.
* Implemented the gained knowledge on the activities.

**References:**

[1] <https://www.youtube.com/watch>? v=eoGkJWgxurQ&list=PLxCzCOWd7aiGz9donHRrE9I3Mwn6XdP8p&index=30

[2] https://www.youtube.com/watch?v=l5-3mbBV1BQ&list=PLxCzCOWd7aiGz9donHRrE9I3Mwn6XdP8p&index=31