sjsytu

**Learning Report – Linux OS and Programming**

Course Code: <CODE>

./



**Document History**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Ver. Rel. No.** | **Release Date** | **Prepared. By** | **Reviewed By** | **Approved By** | **Remarks/Revision Details** |
| 1.0 | 27/12/2020 | 99003125 | 99003124  99003126 | G Bharath |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

Table of Contents

[Table of Contents 3](#_Toc60662831)

[Activity - 1 4](#_Toc60662832)

[**Learning outcomes** 4](#_Toc60662833)

[Activity - 2 4](#_Toc60662834)

[**Learning outcomes** 4](#_Toc60662835)

[Activity - 3 4](#_Toc60662836)

[**Learning outcomes** 4](#_Toc60662837)

[Activity - 4 5](#_Toc60662838)

[**Learning outcomes** 5](#_Toc60662839)

[Activity – 5 5](#_Toc60662840)

[**Learning outcomes** 5](#_Toc60662841)

[References 6](#_Toc60662842)

# 

# Activity - 1

* To find the total number of lines in a file.
* Finding the total number of empty lines.
* Finding the number of lines with single character.

## **Learning outcomes**

* Usage of different commands to read the number of lines.
* Using command ‘sed’ to read from lines 2 to 5.
* Usage of commands like wc and grep.
* Learnt to use file accessing and appending commands along with filter commands.

# Activity - 2

* Valgrind check on file to check for memory leaks and append he error message to new text

file.

* CPP check on file and append he messge to new file.

## **Learning outcomes**

* Using valgrind to check for memory leaks and detecting any memory leaks in program.
* Using cppcheck to check for code quality and cpp check of the file.
* Usage of file handling commands to access and append.

# Activity - 3

Static library and Dynamic library.

* Creating user defined libraries and linking user defined functions as library both statically and

dynamically.

## **Learning outcomes**

* Implementing C program builder.
* Using utilities and implementing codes in separate header and C files.
* Created our library and learnt to link that to a static and dynamic type.
* Usage of idconfig to link a dynamic library.
* Implementing Makefile for the same.

# Activity - 4

* To count no.of lines, words, characters in given file.
* To copy one file contents to other using open,read,write,close system calls.
* to send specific signal to a target process
* Compile & link any c/c++ program within child process by launching gcc using execl/execlp.
* Designing a mini shell.
* Building multifile program using fork & exec.
* Print current time periodically.
* Finding min/max element from large array using parallel computations.
* Compute parallel sum of large array.

## **Learning outcomes**

* How to make system call and implement different system calls. Based on file descriptors by any process.
* How to handle and run a process.
* How to create parent and child process?
* Creating multiple child processes.
* Kill or stop process.
* Implementing how to wait a process and override in a child process to give our own.
* Learnt to avoid making blocking calls in thread to avoid getting the whole process blocked.
* Over writing child process using exec signals
* Blocking parent process till completion of child process.

# Activity – 5

Mutex and semaphores

## **Learning outcomes**

* Learnt to implement sequencing and mutual exclusion.
* Prioritizing or locking a particular process for sequencing the flow of program.
* Working with named and unnamed semaphores, and using named semaphores in shared memory.
* Analyzing the return type for mutex to check for success or failure.
* Using threads for working with producer and customer.
* Handling context switching in order to avoid deadlocks.
* Using pipes and fifo to overcome limitations of semaphores and mutex.
* Using operations on shared memory such as read write and update.

# References

1. <https://www.tutorialspoint.com/gnu_debugger/index.htm>
2. <https://www3.ntu.edu.sg/home/ehchua/programming/cpp/gcc_make.html>