./

Learning Report – C, Linux & OS Programming



|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Ver. Rel. No.** | **Release Date** | **Prepared. By** | **Reviewed By** | **To be approved By** | **Remarks/Revision Details** |
| 1 |  | Name/PS No | Name/PS No | Module Owner Name | Comments |
| 2 | 15/02/21 | Shivanshu/99003731 |  |  |  |

**Document History**

Table of Contents

[Table of Figures 3](#_Toc65454203)

[Table of Tables 3](#_Toc65454204)

[ACTIVITY 1: BOXES (GROUP ACTIVITY) 4](#_Toc65454205)

[Requirements 4](#_Toc65454206)

[Introduction 4](#_Toc65454207)

[GIT COMMITS 4](#_Toc65454208)

[DOXYGEN FILE 5](#_Toc65454209)

[REFERENCES 5](#_Toc65454210)

[ACTIVITY 2: LINUX OS PROGRAMMING 6](#_Toc65454211)

[THEME 6](#_Toc65454212)

[EPIC 6](#_Toc65454213)

[USER STORY 6](#_Toc65454214)

[BASIC SYSTEM CALLS: 6](#_Toc65454215)

[PROCESS AND EXEC CALLS: 6](#_Toc65454216)

[THREADING: 6](#_Toc65454217)

[SEMAPHORE AND MUTEX: 6](#_Toc65454218)

[MESSAGE QUEUE AND PIPES: 7](#_Toc65454219)

[SHARED MEMORY: 7](#_Toc65454220)

## Table of Figures

[figure1 1 .CLASS DIAGRAM FOR HLR 9](#_Toc64285898)

[figure1 2. DEPLOYMENT DIAGRAM FOR LOW LLR 10](#_Toc64285899)

[figure1 3. USE-CASE DIAGRAM FOR LLR 11](#_Toc64285900)

[figure1 4.GIT COMMITS 14](#_Toc64285901)

[figure1 5.GIT MAKE FILE 14](#_Toc64285902)

[figure1 6. GIT CPP CHECK 15](#_Toc64285903)

## Table of Tables

[TABLE1 1.HIGH LEVEL REQUIREMENT 7](#_Toc64285904)

[TABLE1 2. LOW LEVEL REQUIREMENTS 8](#_Toc64285905)

[TABLE1 3. TEST PLAN 13](#_Toc64285906)

# ACTIVITY 1: BOXES (GROUP ACTIVITY)

[https://github.com/99003738/define\_box.git (Link](https://github.com/99003733/SDLC_N2_Calculator.git%20(Link) to GITHUB REPOSITORY)

# Requirements

# Introduction

In this activity we worked as a team wrote code using c programming for array of boxes where box is a structure with various parameters and various functions to access it for certain criteria’s. Here I implemented the code for adding data for extra boxes at the end and also method to calculate the difference in minimum and maximum volume of array of boxes.

## GIT COMMITS

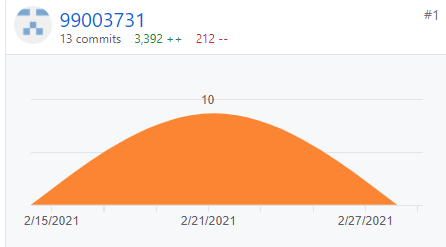


figure1 .GIT COMMITS

## DOXYGEN FILE

# REFERENCES

1. https://github.com/adam-p/markdown-here/wiki/Markdown-Cheatsheet)

2. https://guides.github.com/features/mastering-markdown/)

3. https://github.com/ejwa/gitinspector.git

4. https://docs.github.com/en/actions/learn-github-action

5. https://stackedit.io/app#

6. https://creately.com/

# ACTIVITY 2: LINUX OS PROGRAMMING

# THEME

Learnt basic terminal commands, then learnt about system calls, child and parent process using fork, use of exec commands , threading, semaphore, mutex, message queue and shared memory programming for developing API’s using C,

# EPIC

Epic according to the requirements provided are:

1. Basic System Calls.
2. Process and Exec calls.
3. Threading.
4. Semaphores and Mutex.
5. Message Queue and Pipes.
6. Shared Memory.

# USER STORY

## BASIC SYSTEM CALLS:

1. Open, read, write, close, signals these commands were learnt.

## PROCESS AND EXEC CALLS:

1. Creation of multiple child and parent process in same program using fork().
2. Calling child process of other sources files using in exec system calls.

## THREADING:

1. Threading used to connect variable data for child and parent process.

2. pthread.h library introduced.

## SEMAPHORE AND MUTEX:

1. Mutex system call is for monotonically sequencing data like it works as half duplex communication.

2. Semaphores used to sequence data in two –way communication but one after the another only.

## MESSAGE QUEUE AND PIPES:

1. Message is sent and received using First in and First Out concept.

## SHARED MEMORY:

1. Memory is shared between parent and child or child and child process.

2. This is the most fastest way of parent and child to communicate.