./

Learning Report – C and 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 | 01/03/21 | Himanshu Kumar/99003721 |  |  |  |
| 3 |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

**Document History**

Table of Contents

[ACTIVITY 1: SYSTEM 3](#_Toc53129064)

**[INTRODUCTION](#_Toc53129065)** [4](#_Toc53129065)

**[Basic Linux Commanad](#_Toc53129066)** [4](#_Toc53129066)

**[string breaker program](#_Toc53129067)** 5

**[group activity](#_Toc53129068)**5

**[box problem](#_Toc53129069)**6

[HIGH LEVEL DESIGN 7](#_Toc53129070)

[LOW LEVEL DESIGN](#_Toc53129071) 8

**[what we have learned](#_Toc53129072)**9

**[linux exe system call](#_Toc53129073)** [9](#_Toc53129073)

[getpid sample code 1](#_Toc53129074)0

**[thread and signal](#_Toc53129075)** [11](#_Toc53129075)

**[REFERENCES](#_Toc53129076)** [1](#_Toc53129076)1

**INTRODUCTION**

This is an attempt to learn and build c code and Linux programming we have implemented many problem statement and worked on some group activity project as well as Individual project.

BASIC COMMANDS FOR LINUX

### **What Is Linux?**

Linux is an operating system's kernel. You might have heard of UNIX. Well, Linux is a UNIX clone. But it was actually created by Linus Torvalds from Scratch. Linux is free and open-source, that means that you can simply change anything in Linux and redistribute it in your own name! There are several Linux Distributions, commonly called “distros”.

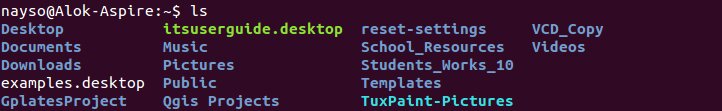
* Ubuntu Linux
* Red Hat Enterprise Linux
* Linux Mint
* Debian
* Fedora

### **Linux Shell or “Terminal”**

a shell is a program that receives commands from the user and gives it to the OS to process, and it shows the output. Linux's shell is its main part.

### **Linux Commands**

**Basic Commands**

****

**ls**— Use the **"ls"** command to know what files are in the directory you are in.

**pwd** — When you first open the terminal, you are in the home directory of your user.

**zY0aprEyMtOxLhiyyRKWvY3vwB9cPJJQtxubemw6**

Q1. Write a function to break the given string and return the correct string between '$' and ';' with no '$' or ';' in between.

#include <stdio.h>

int main() {

char str\_input[1000];

int start=0,end=0;

printf("Enter a string\_input ");

fgets(str\_input, sizeof(str\_input), stdin);

for (int i = 0; str\_input[i] != '\0'; ++i) {

if (str\_input[i] == '$')

start=i;

if (str\_input[i] == ';') {

end=i;

break;

}

}

for (int k = start; k<=end; ++k) {

printf("%c",str\_input[k]);

}

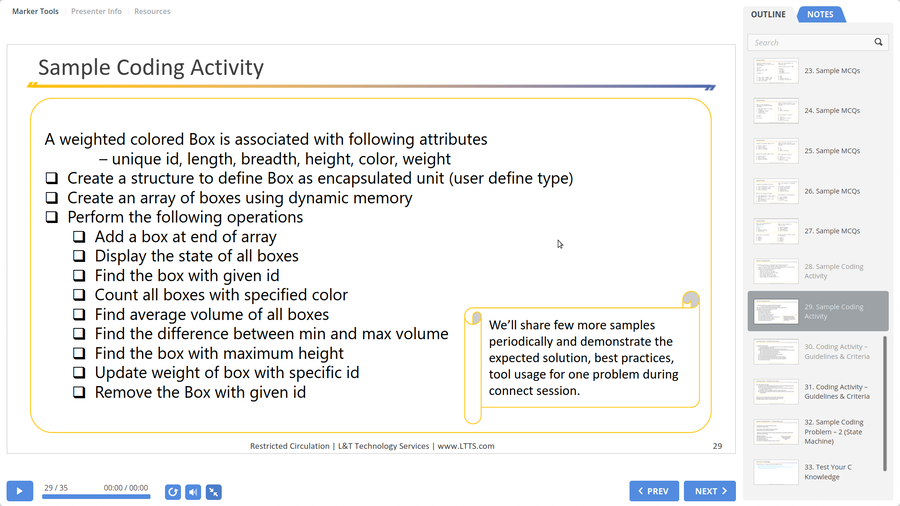
return 0;

}

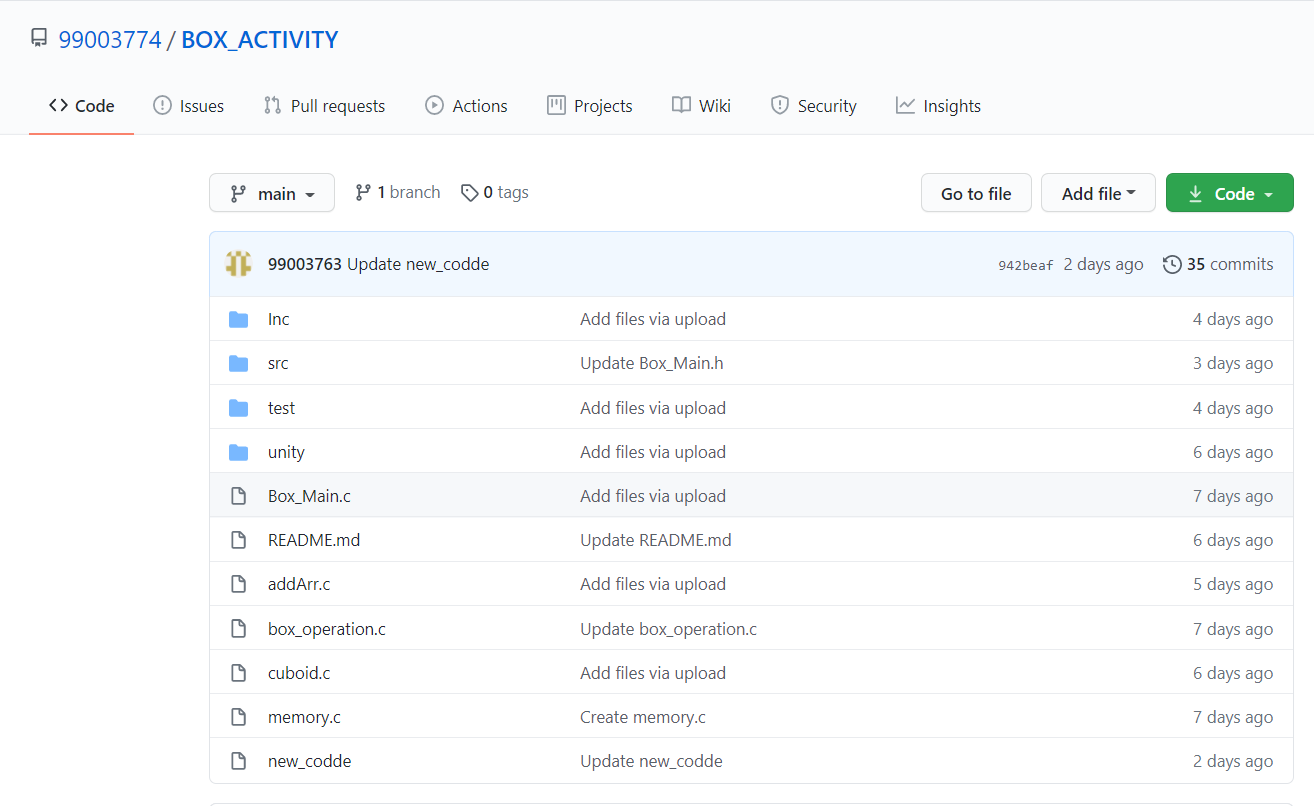
**The Build Process - C/C++**

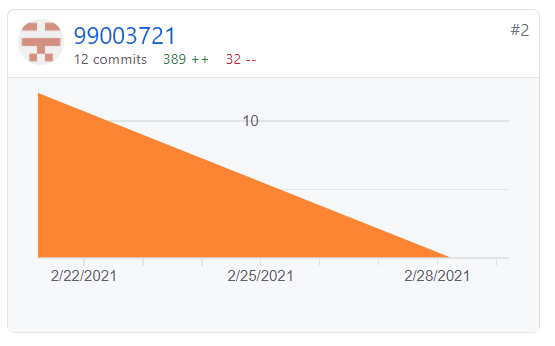
Sequence of Events : ****Editor**** => ****Type the Code**** => ****Press Build & Run**** => ****Wait[Build Process]...**** => ****Prompt/Window****

****Group Coding Activity****

****

**Git Link : -** <https://github.com/99003774/BOX_ACTIVITY.git>

****

****

**Topic we have learned :**

1. OS and its features
2. Process and Process Life cycle
3. Types of Kernel
4. Scheduling

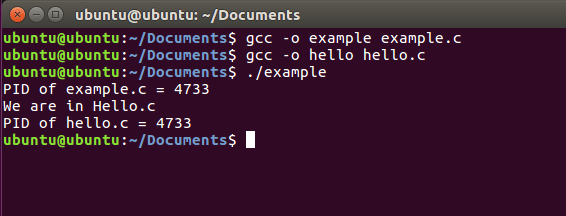
# **Linux Exec System Call**

The exec system call is used to execute a file which is residing in an active process. When exec is called the previous executable file is replaced and new file is executed.  
the standard names for these functions are as follows:

1. ****execl****
2. ****execle****
3. ****execlp****
4. ****execv****
5. ****execve****
6. ****execvp****

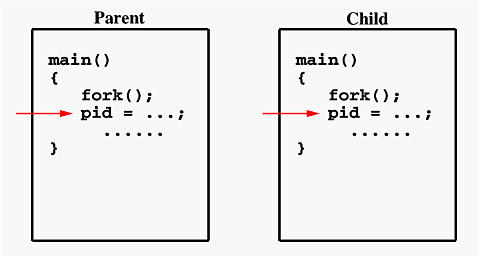
**Getpid code Example : -**

#include <stdio.h>  
#include <unistd.h>  
#include <stdlib.h>  
int main(int argc, char \*argv[])  
{  
[printf](https://www.opengroup.org/onlinepubs/009695399/functions/printf.html)("We are in Hello.c\n");  
[printf](https://www.opengroup.org/onlinepubs/009695399/functions/printf.html)("PID of hello.c = %d\n", getpid());  
return 0;  
}

****

# fork() System Call :-

System call **fork()** is used to create processes. It takes no arguments and returns a process ID

****

### **Threads and Signals**

 Thread has its own signal mask, but the signal disposition is shared by all threads in the process.

If the signal is related to a hardware fault or expiring timer, the signal is sent to the thread whose action caused the event.

**REFERENCES –**

**<https://www.youtube.com/watch?v=kjvy_zwhBuA>**

<https://www.csl.mtu.edu/cs4411.ck/www/NOTES/process/fork/create.html>

<https://linuxhint.com/linux-exec-system-call/>

<https://www.youtube.com/watch?v=83M5-NPDeWs>

Inter Process Communication: [https://www.youtube.com/watch?v=G2vwkBZy894](https://www.youtube.com/watch?v=G2vwkBZy894" \o "https://www.youtube.com/watch?v=G2vwkBZy894" \t "https://www.yammer.com/lnttsgroup.onmicrosoft.com/" \l "/threads/_blank)

Locking (Software Solutions): [https://www.youtube.com/watch?v=B\_lH2Xov\_g4](https://www.youtube.com/watch?v=B_lH2Xov_g4" \o "https://www.youtube.com/watch?v=B_lH2Xov_g4" \t "https://www.yammer.com/lnttsgroup.onmicrosoft.com/" \l "/threads/_blank)

Semaphores: [https://www.youtube.com/watch?v=UM4tk3J6WxQ](https://www.youtube.com/watch?v=UM4tk3J6WxQ" \o "https://www.youtube.com/watch?v=UM4tk3J6WxQ" \t "https://www.yammer.com/lnttsgroup.onmicrosoft.com/" \l "/threads/_blank)

Mutex: [https://www.youtube.com/watch?v=xKqO04SN6C0](https://www.youtube.com/watch?v=xKqO04SN6C0" \o "https://www.youtube.com/watch?v=xKqO04SN6C0" \t "https://www.yammer.com/lnttsgroup.onmicrosoft.com/" \l "/threads/_blank)