Subject Name: **Operating System**

Subject Code: **22CS005**

Session: **2022-23**

Department: **DCSE**

****

|  |  |  |
| --- | --- | --- |
| **Submitted By:**  Bhuvesh Mittal  2210991450  G05-A |  | **Submitted To:**  Dr. Rajwinder kaur |

**Index**

|  |  |  |  |
| --- | --- | --- | --- |
| **S. No.** | **Experiments** | **Page Number** | **Remarks** |
| 1 | **Installation**: Configuration & Customizations of Linux  **Introduction to GCC compiler:** Basics of GCC, Compilation of program, Execution of program, Time stamping, Automating the execution using Make file. |  |  |
| 2 | Implement Process concepts using C language by Printing process Id, Execute Linux command as sub process, Creating and executing process using fork and exec system calls. |  |  |
| 3 | Implement FCFS, SJF, priority scheduling, and RR scheduling algorithms in C language. |  |  |
| 4 | Implement the basic and user status commands like: su, sudo, man, help, history, who, whoami, id, uname, uptime, free, tty, cal, date, hostname, reboot, clear |  |  |
| 5 | Implement deadlock in C by using shared variable. |  |  |
| 6 | File system: Introduction to File system, File system Architecture and File Types. |  |  |
| 7 | Implement the commands that is used for Creating and Manipulating files: cat, cp, mv, rm, ls and its options, touch and their options, which is, where is, what is |  |  |
| 8 | Implement Directory oriented commands: cd, pwd, mkdir, rmdir |  |  |
| 9 | Implement File system commands: Comparing Files using diff, cmp, comm |  |  |

**EXPERIMENT 1**

PART 1:

**Installation**: Configuration & Customizations of Linux

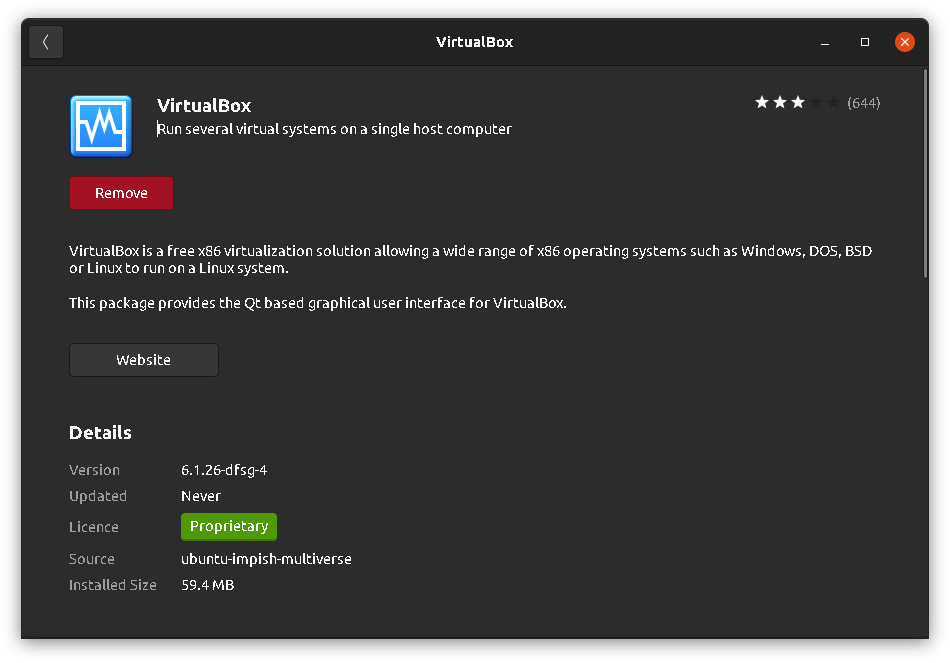
First Download an Ubuntu Image

• You can download an Ubuntu image

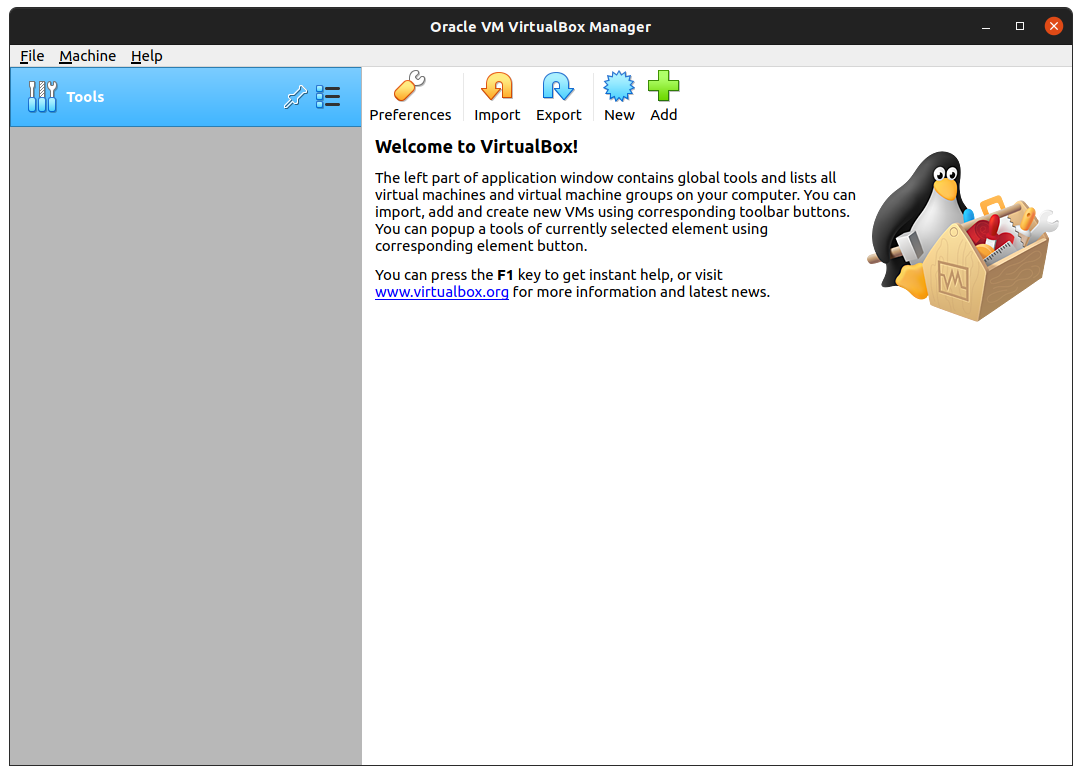
• https://ubuntu.com/download/desktop • Make sure to save it to a memorable location on your PC! For this tutorial, we will use the Ubuntu 20.04 LTS release.

• On Mac OS or Windows, you can download VirtualBox from the downloads page <https://www.virtualbox.org/wiki/Downloads>

• This page also includes instructions to download VirtualBox for Linux. However, on Ubuntu, you can find VirtualBox by simply searching for it in the Ubuntu Software app.



* Once you have completed the installation, go ahead and run VirtualBox.

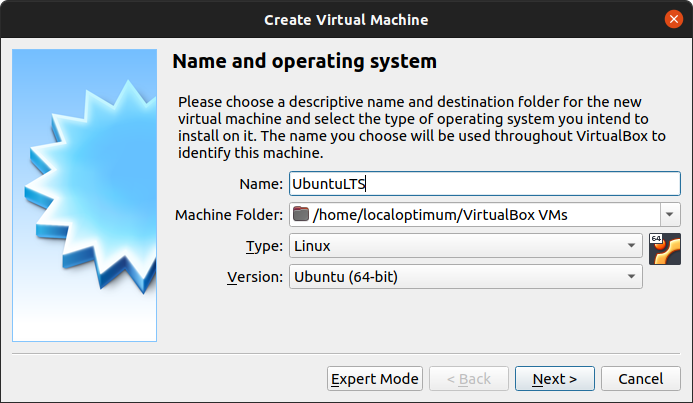


Step 2. Create a new virtual machine

* Click New to create a new virtual machine. Fill in the appropriate details
* Name: If you include the word Ubuntu in your name the Type and Version will

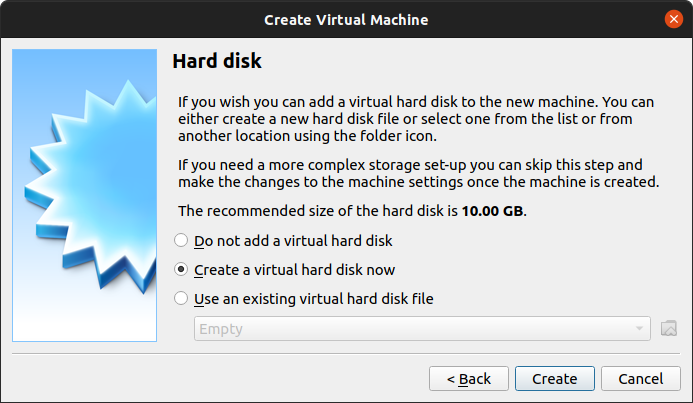
auto-update.

* Machine Folder: This is where your virtual machines will be stored so you can resume working on them whenever you like.
* Type: Linux
* Version: Ubuntu (64-bit)

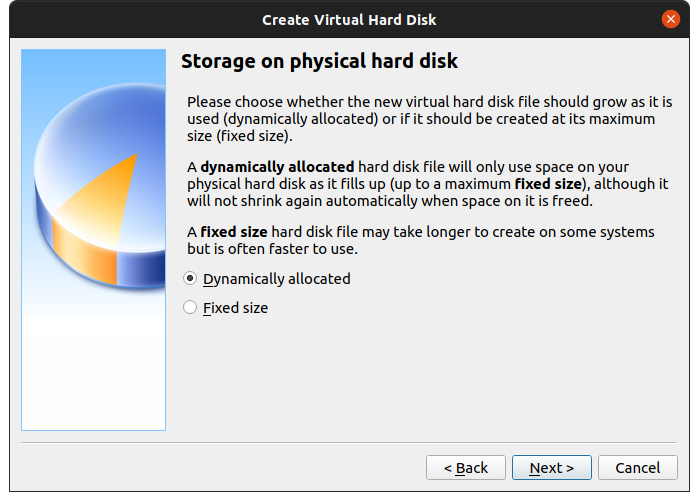
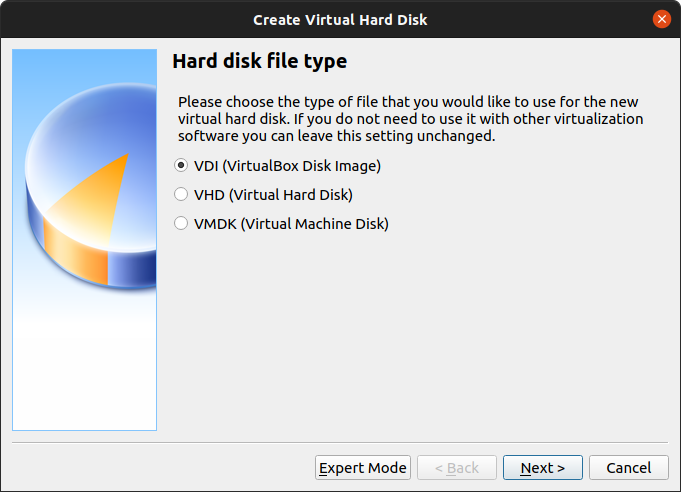


* After that, you can select how much of your hard disk your VM

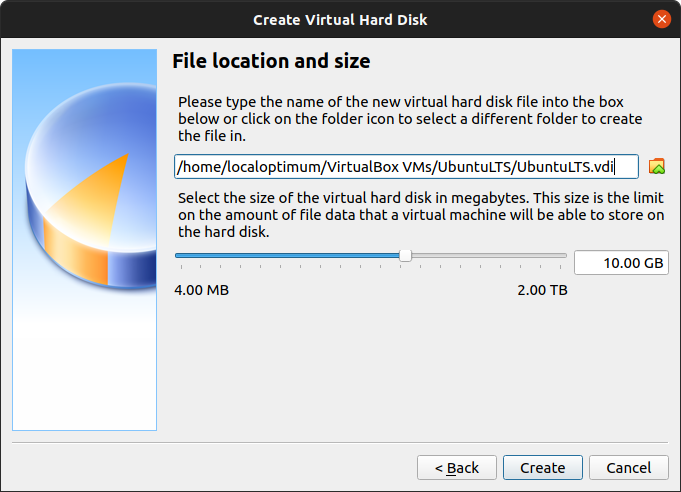
will use.



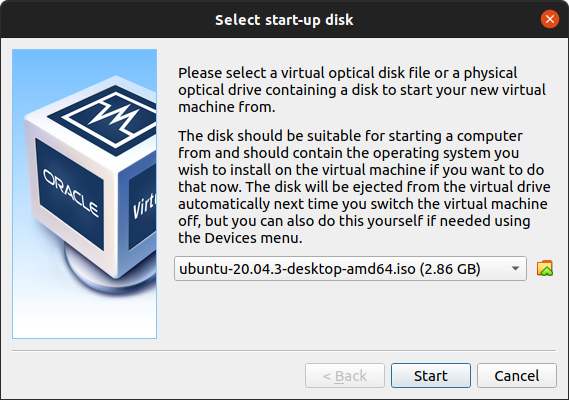
Then you can choose whether the hard disk is dynamically allocated (up to the limit we will set on the next screen), filling up as the VM requires it. Otherwise, we can tell it to allocate the full amount of memory right from the start. This will improve performance but may take up unnecessary space. We’ll leave it as dynamically allocated for this installation.

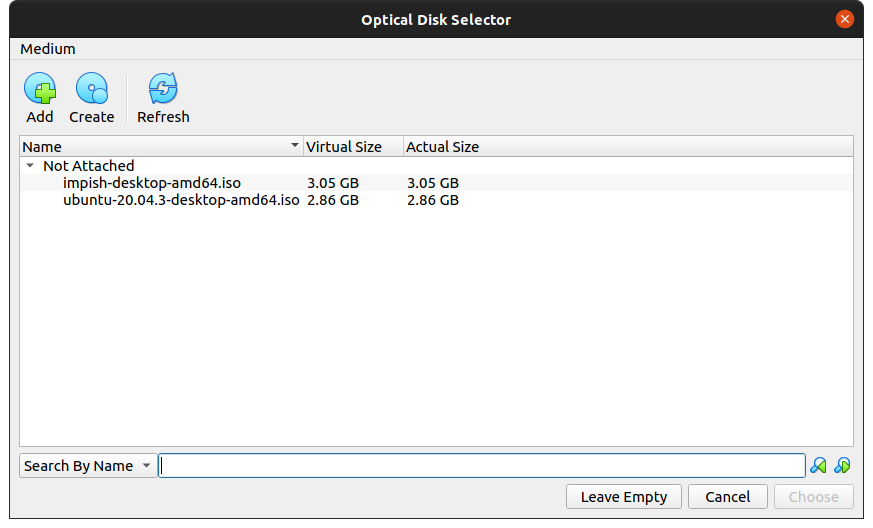


* Finally, you can set the maximum amount of memory your VM can access.



* Click Start to launch the virtual machine. You will be prompted to select the start-up disk. Use the file icon to open the Optical disc selector and click Add to find your .iso file

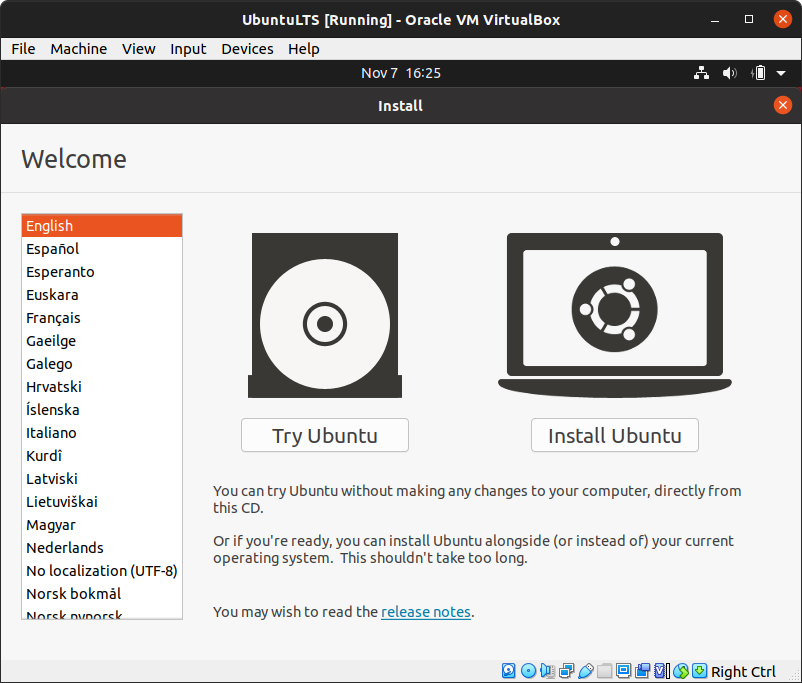




Choose the disc image you want to use, then click Start on the start-up disc window.

Note: If you close this window before selecting an image you can still do so from the Devices menu at the top of the VM window. Select Devices > Optical Drives > Choose/Create a disc image…

* Ubuntu desktop should now boot and display the installation menu.



Now follow instructions to complete installation

PART 2:

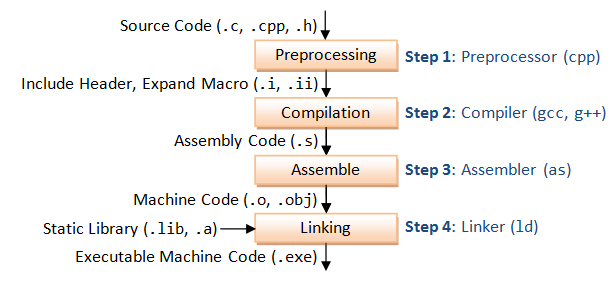
**Introduction to GCC compiler:** Basics of GCC, Compilation of program, Execution of program, Time stamping, Automating the execution using Make file.

**History About GCC**

* GCC is now used to refer to the “GNU Compiler Collection”.
* The GNU project was started in 1984 to create a complete Unix-like operating system as free software, in order to promote freedom and cooperation among computer users and programmers.
* Every Unix-like operating system needs a C compiler, and as there were no free compilers in existence at that time, the GNU Project had to develop one from scratch.

**Major features of GCC**

* First of all, GCC is a portable compiler—it runs on most platforms available today.
* GCC is not only a native compiler—it can also cross-compile any program, producing executable ﬁles for different system.
* GCC has a modular design, allowing support for new languages.
* Most importantly, GCC is free software.



**Compiling a C program**

* There are two ways of compiling a C program
* 1). $ gcc bad.c // compiling the C program.

$ ./a.out // executing the object file.

* 2). $ gcc bad.c -o bad // compilation with different object file name.

$ ./bad // executing the object file.

**Compiling multiple source ﬁles**

$ gcc -Wall main.c hello\_fn.c -o newhello

**Creating object ﬁles from source ﬁles using compile only option -c**

$ gcc -Wall -c main.c

**Creating executables from object ﬁles and linking them together**

$ gcc main.o hello\_fn.o -o hello

The resulting executable ﬁle can now be run:

$ ./hello

**Link order of object ﬁles.**

$ gcc main.o hello\_fn.o -o hello (correct order)

$ gcc hello\_fn.o main.o -o hello (incorrect order)

**Linking with external libraries**

* #include <math.h>

#include <stdio.h>

void main (void){

double x = sqrt (2.0);

printf ("The square root of 2.0 is %f\n", x);

}

***$ gcc -Wall calc.c /usr/lib/libm.a -o calc***

***$ gcc -Wall calc.c -lm -o calc***

***$ gcc -Wall -lm calc.c -o calc (incorrect order)***

***$ gcc -Wall calc.c -lm -o calc (correct order)***

**Search path**

* $ gcc –Wall –L /opt/gdbm-1.8.3/lib dbmain.c –lgdbm main.c

**Compiling with optimization**

$ gcc -Wall -O0 test.c -lm

$ time ./a.out

real 0m13.388s

user 0m13.370s

sys 0m0.010s

**nm" Utility - List Symbol Table of Object Files**

The utility "nm" lists symbol table of object files. For example,

> nm hello.o

00000000 b .bss

00000000 d .data

00000000 r .eh\_frame

00000000 r .rdata

00000000 t .text

U \_\_\_main

00000000 T \_main

U \_printf

U \_puts

> nm hello.exe | grep printf

00406120 I \_\_imp\_\_printf

0040612c I \_\_imp\_\_vfprintf

00401b28 T \_printf

00401b38 T \_vfprintf

"nm" is commonly-used to check if a particular function is defined in

an object file.

A 'T' in the second column indicates a function that is defined,

while a 'U' indicates a function which isundefined and should be

resolved by the linker.

**file" Utility - Determine File Type**

The utility "file" can be used to display the type of object files and executable files. For example,

> gcc -c hello.c

> gcc -o hello.exe hello.o

> file hello.o

hello.o: 80386 COFF executable not stripped - version 30821

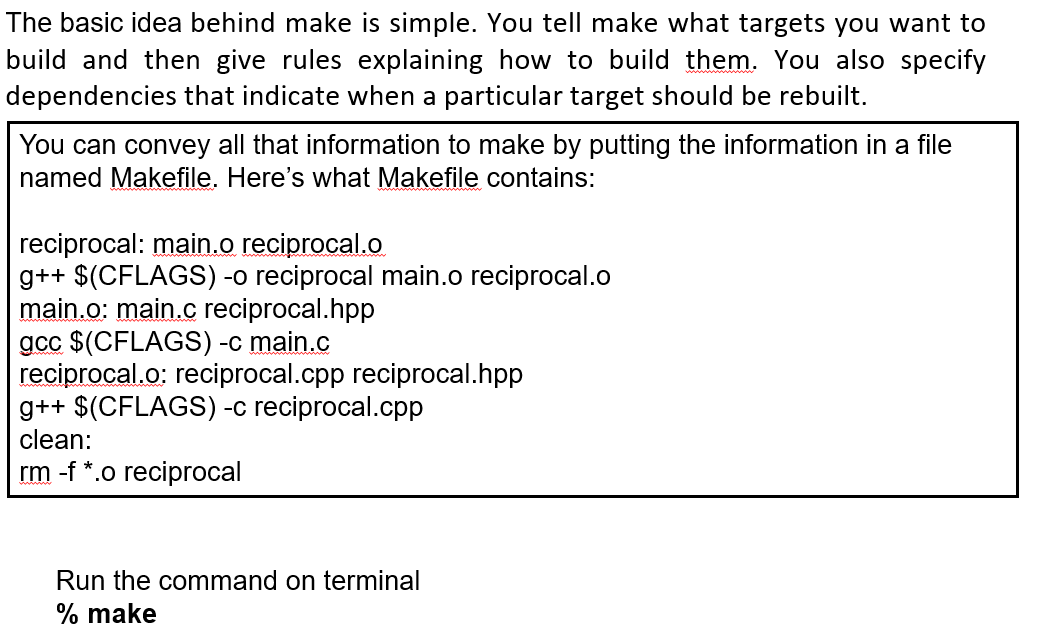
> file hello.exe

hello.exe: PE32 executable (console) Intel 80386

Verbose Options: Compile in verbose mode (-v) to study the library-

paths (-L) and libraries (-l) used in your system

**Make File**



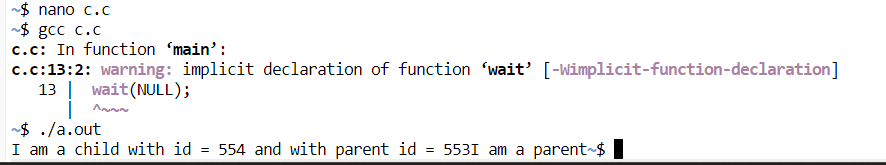
**EXPERIMENT 2**

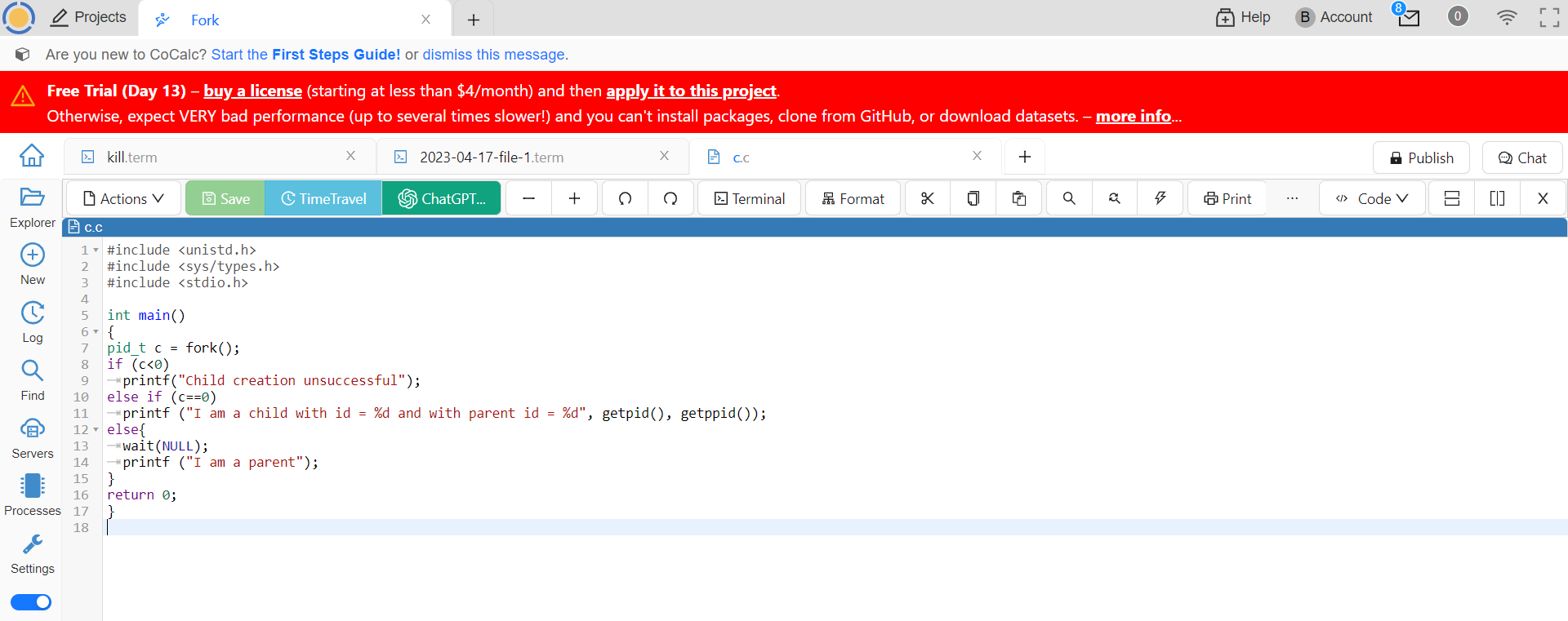
Implement Process concepts using C language by printing Process ID, Execute Linux Command as sub process, Creating and executing process using fork and exec system calls.

FORK SYSTEM CALL

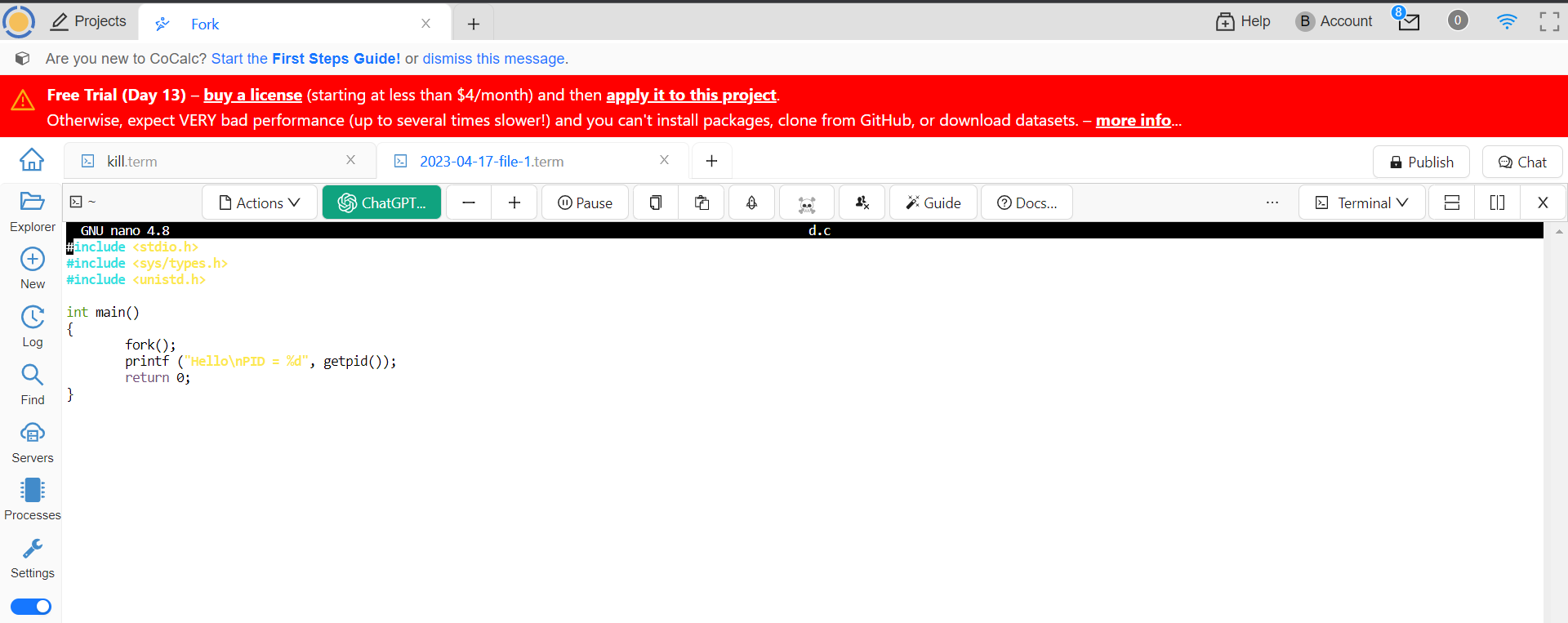
* It is used to create processes.
* It creates a new process called child process which runs concurrently with the parent process.
* Fork returns the following values:

1. Negative value when child process creation is unsuccessful.
2. Zero when a new child process is created.
3. Positive value when the id of the child process is returned to the parent.





****

****

EXEC SYSTEM CALLS

* 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 entire content of the process is replaced by a new program.



