

A thick black L-shaped frame is positioned on the left and right sides of the slide, framing the central text.

INTRODUCTION TO LINUX

LECTURE (8)

Elzahraa Hasan

Agenda

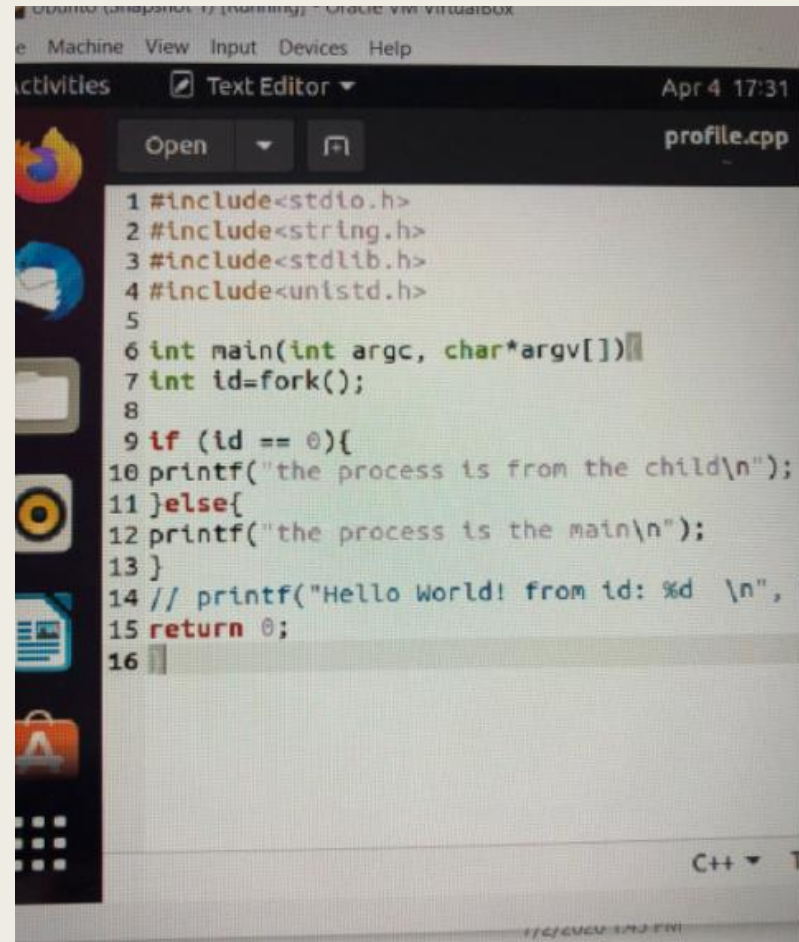
- process

Process definition:

- ***Program***: is a file exist on your device without being run.
- ***Process***: a running program in your device.

As an administrator you must be able to control the process, to run, to pause, to stop, and to terminate it.

To create a process using C++



```
1 #include<stdio.h>
2 #include<string.h>
3 #include<stdlib.h>
4 #include<unistd.h>
5
6 int main(int argc, char*argv[])
7 {
8     int id=fork();
9     if (id == 0){
10 printf("the process is from the child\n");
11 }else{
12 printf("the process is the main\n");
13 }
14 // printf("Hello World! from id: %d \n",
15 return 0;
16 }
```

The screenshot shows a text editor window titled 'profile.cpp' with a dark theme. The code is a C++ program that demonstrates process creation using the `fork()` system call. It includes headers for `stdio.h`, `string.h`, `stdlib.h`, and `unistd.h`. The `main` function calls `fork()` to create a child process. If the return value is 0, it prints 'the process is from the child\n'. Otherwise, it prints 'the process is the main\n'. The program ends with `return 0;`. The editor interface includes a menu bar with 'Machine', 'View', 'Input', 'Devices', and 'Help', and a toolbar with 'Open' and a file icon. The system clock in the top right corner shows 'Apr 4 17:31'.

Process ID

- Every process has a number, process I.D “PID”
- Every process has a parent which also has a number “PPID”

- To display the current process:

Ps

- To display the current process in all shells

Ps a

- To display more details

Ps aux

Process ID

```
[root@server ~]# ps aux
```

USER	PID	%CPU	%MEM	VSZ	RSS	TTY	STAT	START	TIME	COMMAND
root	1	0.0	0.3	126692	7404	?	Ss	17:48	0:02	/usr/lib/systemd/systemd --swi
root	2	0.0	0.0	0	0	?	S	17:48	0:00	[kthreadd]
root	3	0.0	0.0	0	0	?	S	17:48	0:00	[ksoftirqd/0]
root	5	0.0	0.0	0	0	?	S<	17:48	0:00	[kworker/0:0H]
root	7	0.0	0.0	0	0	?	S	17:48	0:00	[migration/0]
root	8	0.0	0.0	0	0	?	S	17:48	0:00	[rcu_bh]
root	9	0.0	0.0	0	0	?	S	17:48	0:00	[rcuob/0]
root	10	0.0	0.0	0	0	?	S	17:48	0:00	[rcuob/1]
root	11	0.0	0.0	0	0	?	S	17:48	0:00	[rcuob/2]
root	12	0.0	0.0	0	0	?	S	17:48	0:00	[rcuob/3]
root	13	0.0	0.0	0	0	?	S	17:48	0:00	[rcuob/4]
root	14	0.0	0.0	0	0	?	S	17:48	0:00	[rcuob/5]
root	15	0.0	0.0	0	0	?	S	17:48	0:00	[rcuob/6]
root	16	0.0	0.0	0	0	?	S	17:48	0:00	[rcuob/7]
root	17	0.0	0.0	0	0	?	S	17:48	0:00	[rcuob/8]

To kill process

■ Lab

Ps auxfrom you tty (tty3 =^f3)

^f4to go to different tty

Login

Touch file1

Nano file1 to write something

^f3 to go to tty3

Ps auxto get the process id (nano file1)

Kill 7400 Put PID

^f4to check the process been killed

To force killing a process

Nano file1 ... to write something

^f3 To go to tty3

Kill -9 7400 Put PID

^f4to check the process been killed

To find a process and kill it

- If you have a running process but you need to stop it to save your resources for another application.

- Lab

^f4to go to different tty

Nano file1 to write something

^f3 to go to tty3

Pgrep nanoto find process id

Kill 7400 Put PID

^f4to check the process been killed

To find a parent of a process

- If you have a halted process but you need to stop, you can kill its parent.

- Lab

Pstreeto know the parent

Pstree | less to know the parent

Ps -ef | lessto find parent process id

/ nano

Kill -9 3456kill parent ID

It terminates the process and the parent.