

# ***Sistemas Operativos***

***Cursada 2023***

**Comisión S21 y S22**

# Procesos

## Definiciones – Que es un Proceso?

- Algunos autores dicen:
  - **Un programa en ejecucion** (introducción al tema)
- En realidad es mucho mas que eso
  - **Es una entidad dinámica** (cambia con el tiempo)  
compuesta por el: Código del programa ejecutable, los recursos necesarios para su ejecucion y el tiempo.
- En cambio el código del programa fuente es una entidad estática

# *Procesos*

## Repaso de Arquitectura

- Teníamos un programa escrito en Assembly, el cual para su ejecución necesitaba cargar en memoria?

- **Segmento de Código**

- **Segmento de Datos**

- **Segmento de Pila**

- **Segmento Extra**

Este ambiente de ejecución estaba planteado para un sistema llamado?

- **Monotarea o Monousuario**

# Procesos

Ahora si un proceso es una entidad dinámica,  
o sea que:

- Se crea
- Se Ejecuta
- Termina

Vamos a tener lo que llamamos

***Ciclo de vida de un Proceso***



# *Procesos*

En la década del 60 desarrollada por IBM para sus mainframe aparece lo que llamamos

***Multiprogramación***

Que no es mas que la ejecucion de procesos en forma

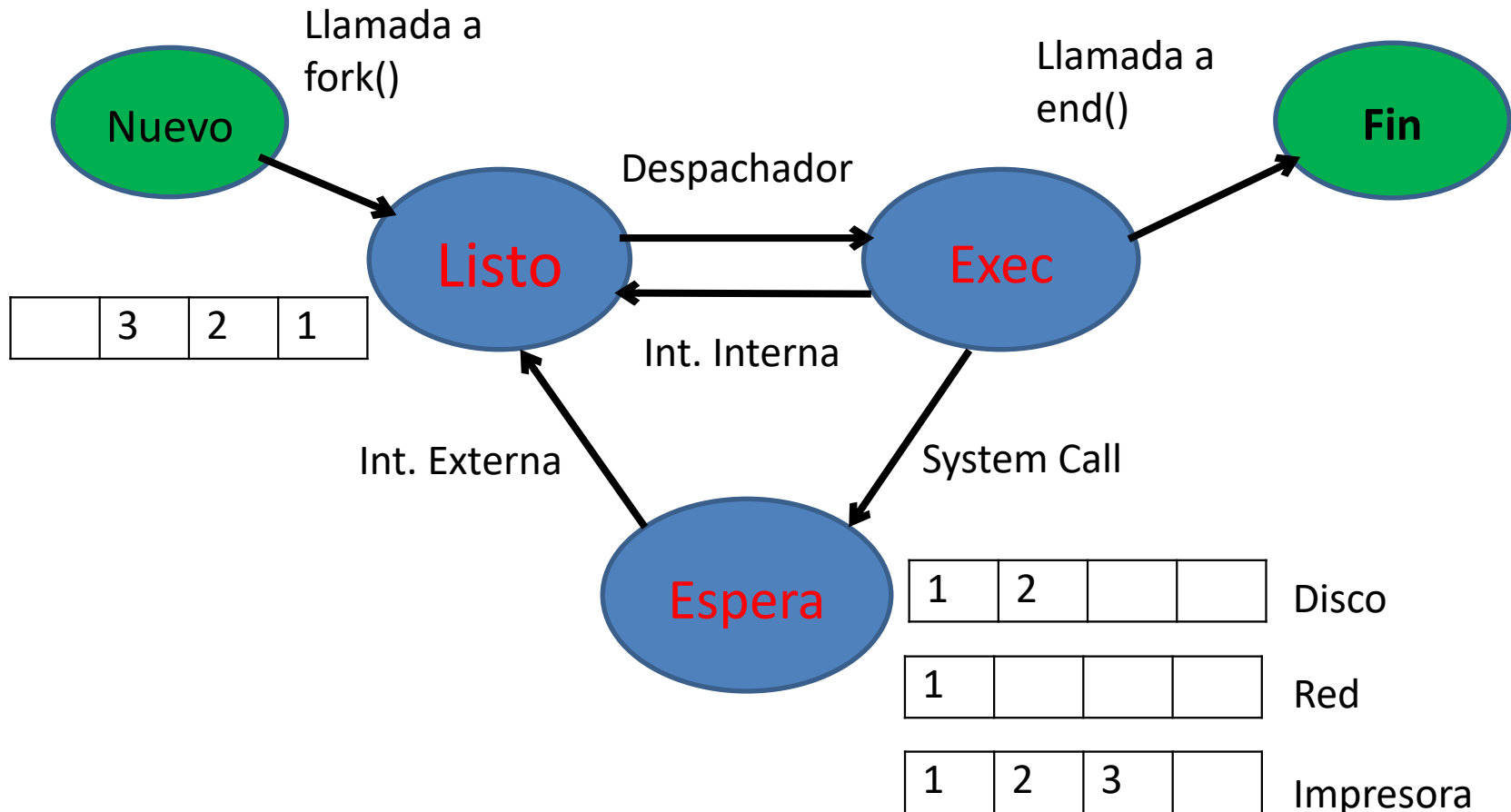
***Concurrente***

Este ciclo de vida de 5 estados es para una sola CPU, si tenemos mas de una, podemos tener procesos ejecutándose en forma

***Paralela o simultanea***

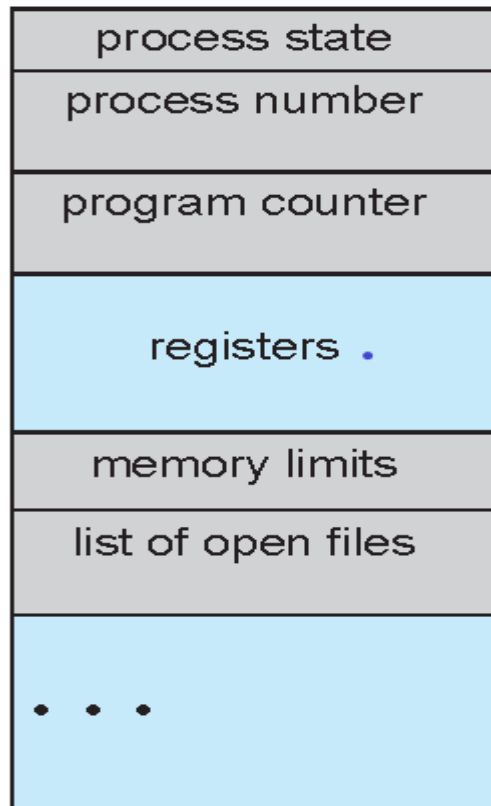
# Procesos

- Ciclo de Vida – diagrama de 5 estados



# Procesos

La información de cada proceso esta en una estructura llamada **PCB (Process Control Block)**



# Procesos

**Parte de la información que contiene la *PCB***

ID del proceso

Estado del proceso

Contador de Programas

Registros de la CPU

Información del planificador

Información de la administración de memoria



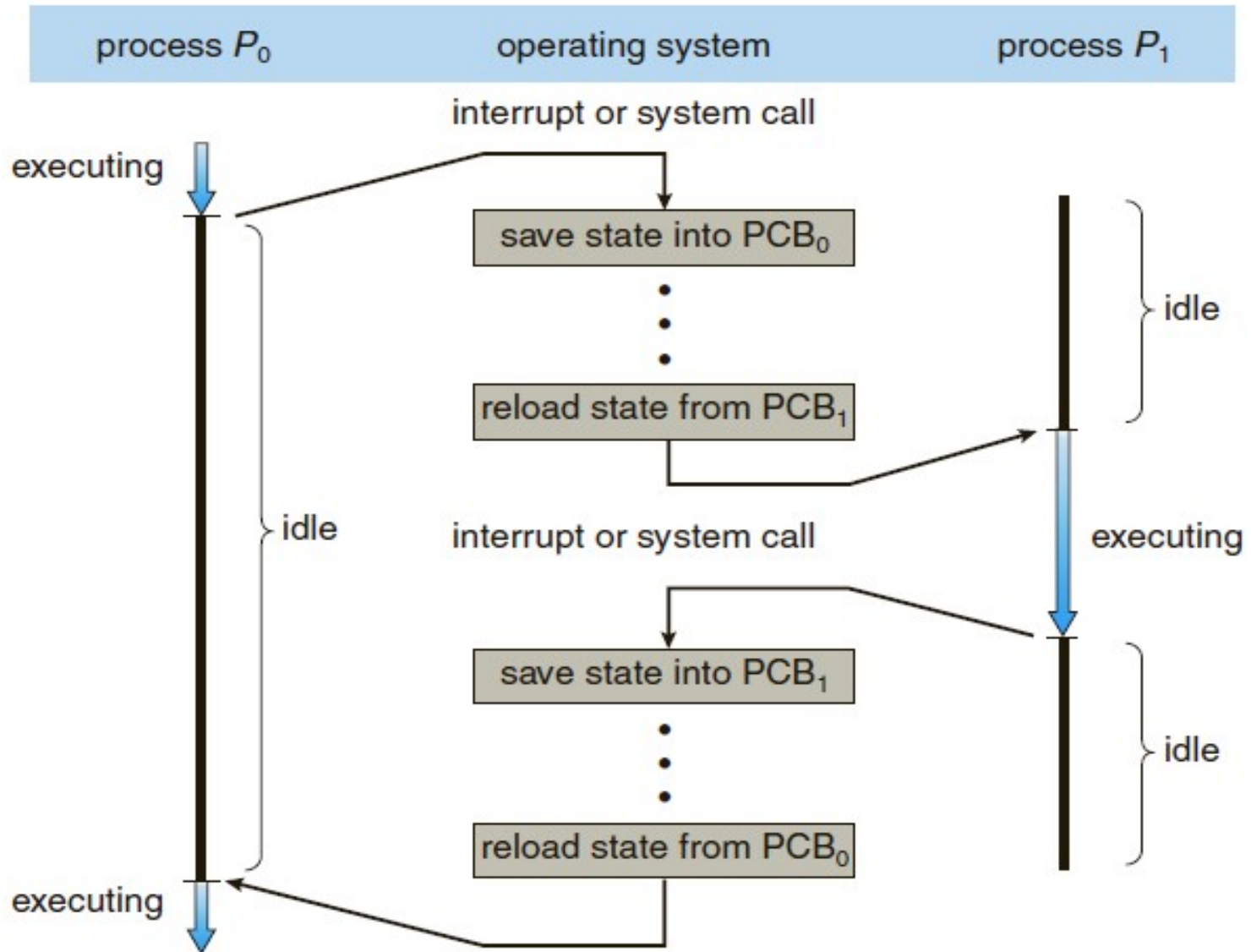
# Planificación de Procesos

El S.O. debe seleccionar los procesos de las distintas colas. Esta selección la realizan los:

## *Planificadores*

- **Planificador de largo plazo:** Selecciona procesos para cargar en memoria
- **Planificador de corto plazo:** Selecciona procesos de la cola de listo
- **Planificador de Mediano Plazo:** Puede descargar procesos de la memoria

# *Cambio de Contexto*



# Planificación de Procesos

- **Procesos Independientes:** No son afectados por otros procesos en ejecución
- **Procesos Cooperativos:** Si pueden ser afectados *por* otros procesos en ejecución, las razones pueden ser:

***Compartir Información*** y para esto es necesario que los procesos se?

***Comuniquen entre ellos***

*Los pueden hacer a través de:*

***Pasaje de mensajes y Memoria compartida***

# Comunicación entre Procesos

- **Directa:** Tienen que estar referenciados
  - ✓ Sincrónica (Ambos se nombran)
  - ✓ Asincrónica (Solo el emisor nombra)
- **Indirecta:** Se reciben a través de buzones o puertos (Se produce un enlace entre procesos si tienen un buzón compartido)
  - ✓ Puede ser univoco entre dos procesos
  - ✓ Puede estar asociado a mas de dos procesos
  - ✓ Varios enlaces y cada uno a un buzón

# Comunicación entre Procesos

## ➤ Indirecta:

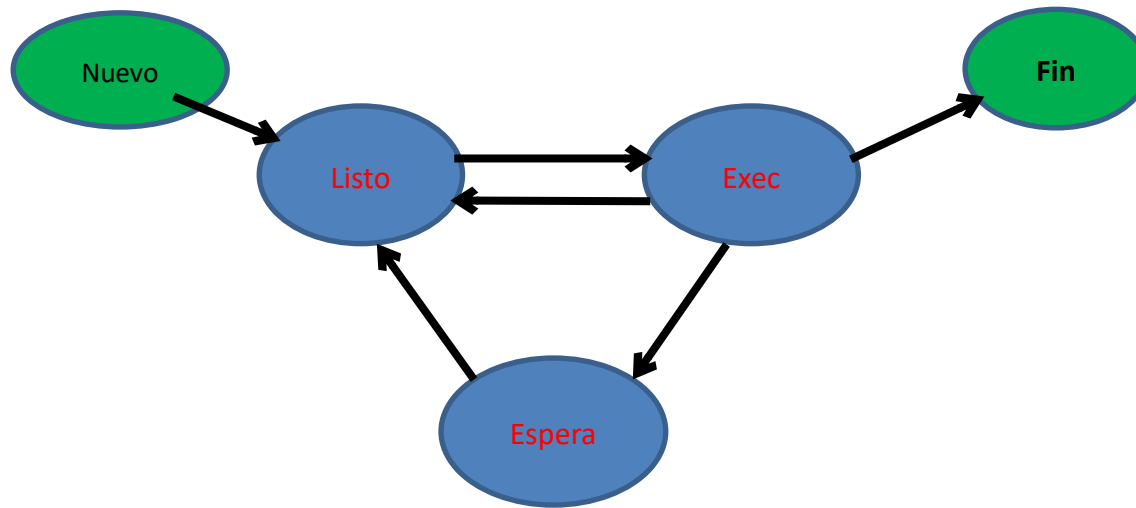
- ✓ **Puede ser propiedad del Proceso**

(forma parte del espacio de direcciones del mismo)

- ✓ **Puede ser propiedad del Sistema operativo**

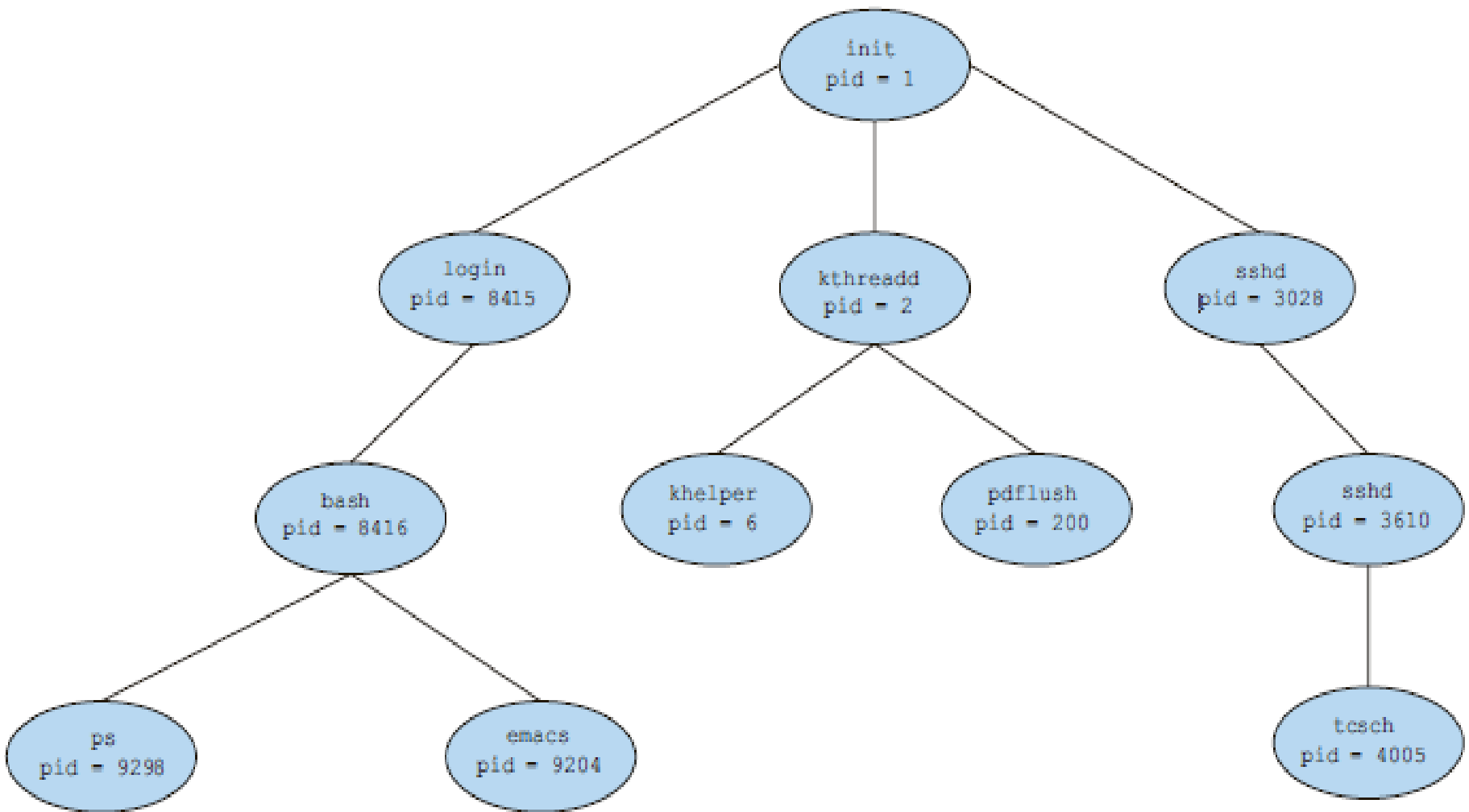
(no estaría ligado a un proceso en particular, el SO debe proporcionar mecanismos para que el proceso puede crear, borrar o compartir un buzón)

# *Como Nace un Proceso*



- Usuario
- Sistema Operativo
- Otro Proceso

# *Arranque del Sistema Linux*



# Comando top de linux

nksistemas@nksistemas: ~

Archivo Editar Ver Buscar Terminal Ayuda

```
top - 10:23:21 up 24 min, 2 users, load average: 0,00, 0,04, 0,19
Tasks: 132 total, 1 running, 131 sleeping, 0 stopped, 0 zombie
%Cpu(s): 2,4 us, 1,0 sy, 0,0 ni, 96,2 id, 0,3 wa, 0,0 hi, 0,0 si, 0,0 st
KiB Mem: 2051796 total, 498932 used, 1552864 free, 5028 buffers
KiB Swap: 1012732 total, 0 used, 1012732 free. 139084 cached Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
776	root	20	0	274800	62572	19828	S	4,3	3,0	0:07.36	Xorg
1200	nksiste+	20	0	419312	29392	22128	S	1,7	1,4	0:02.36	gnome-termina+
1050	nksiste+	20	0	1273500	115328	58400	S	1,0	5,6	0:09.74	gnome-shell
141	root	20	0	0	0	0	S	0,3	0,0	0:00.87	kworker/0:3
1640	root	20	0	23512	2872	2428	R	0,3	0,1	0:00.02	top
1	root	20	0	110524	4688	3024	S	0,0	0,2	0:01.31	systemd
2	root	20	0	0	0	0	S	0,0	0,0	0:00.00	kthreadd
3	root	20	0	0	0	0	S	0,0	0,0	0:00.19	ksoftirqd/0
5	root	0	-20	0	0	0	S	0,0	0,0	0:00.00	kworker/0:0H
7	root	20	0	0	0	0	S	0,0	0,0	0:00.35	rcu_sched
8	root	20	0	0	0	0	S	0,0	0,0	0:00.00	rcu_bh
9	root	rt	0	0	0	0	S	0,0	0,0	0:00.00	migration/0
10	root	rt	0	0	0	0	S	0,0	0,0	0:00.00	watchdog/0
11	root	0	-20	0	0	0	S	0,0	0,0	0:00.00	khelper



# Comando htop de linux

```
nksistemas@nksistemas: ~  
Archivo  Editar  Ver  Buscar  Terminal  Ayuda  
  
CPU[||| 0.9%] Tasks: 83, 132 thr; 1 running  
Mem[||||| 352/2003MB] Load average: 0.19 0.17 0.22  
Swp[ 0/988MB] Uptime: 00:27:06  
  
PID USER PRI NI VIRT RES SHR S CPU% MEM% TIME+ Command  
1050 nksistema 20 0 1247M 114M 58408 S 7.3 5.7 0:10.66 /usr/bin/gnome-shell  
980 nksistema 20 0 1079M 34968 27544 S 1.5 1.7 0:00.38 /usr/lib/gnome-sett  
776 root 20 0 268M 62572 19828 S 1.0 3.0 0:07.98 /usr/bin/Xorg :0 -n  
1001 nksistema -6 0 359M 10468 8048 S 1.0 0.5 0:00.22 /usr/bin/pulseaudio  
998 nksistema 9 -11 359M 10468 8048 S 1.0 0.5 0:00.32 /usr/bin/pulseaudio  
1103 nksistema 20 0 520M 11948 10228 D 1.0 0.6 0:00.06 zeitgeist-datahub  
1109 nksistema 39 19 498M 13796 11112 S 1.0 0.7 0:00.08 /usr/lib/tracker/tr  
1964 root 20 0 24316 3508 2892 R 0.5 0.2 0:00.07 htop  
1139 nksistema 20 0 498M 13796 11112 S 0.5 0.7 0:00.01 /usr/lib/tracker/tr  
1200 nksistema 20 0 409M 29392 22128 S 0.0 1.4 0:02.66 /usr/lib/gnome-term  
1 root 20 0 107M 4688 3024 S 0.0 0.2 0:01.31 /sbin/init  
147 root 20 0 30028 4088 3672 S 0.0 0.2 0:00.27 /lib/systemd/system  
153 root 20 0 42128 4492 2768 S 0.0 0.2 0:00.72 /lib/systemd/system  
391 root 20 0 25388 8932 2032 S 0.0 0.4 0:00.00 dhclient -v -pf /ru  
433 root 20 0 37068 2672 2244 S 0.0 0.1 0:00.00 /sbin/rpcbind -w  
442 statd 20 0 37268 2916 2332 S 0.0 0.1 0:00.00 /sbin/rpc.statd  
456 root 20 0 23348 200 4 S 0.0 0.0 0:00.00 /usr/sbin/rpc.idmap  
477 root 20 0 269M 6352 5616 S 0.0 0.3 0:00.07 /usr/lib/accountsse  
F1Help F2Setup F3Search F4Filter F5Tree F6SortBy F7Nice -F8Nice +F9Kill F10Quit
```

# ***Función para crear procesos***

Usamos la sentencia **fork()** para procesos padre e hijo.

**Declaracion:** `pid_t pidf=fork(void)`

- Crea un nuevo proceso llamado – hijo –
- El proceso hijo es un duplicado del “padre”
- Los dos procesos (padre e hijo)

Tienen identificadores (PIDs) diferentes

Corren en espacios de memoria diferentes

Cuando se crea el hijo los espacios de memoria como las variables tienen los mismos valores, a partir de acá, cada uno modifica las variables en forma independiente

## **Retorno del fork()**

Al padre se le retorna el valor del PID del hijo y al hijo se le retorna 0

Si hay algún problema en la creación del hijo devuelve el valor -1

***Fin clase***