

**DEPARTAMENTO DE SISTEMAS DE INFORMACIÓN**

# Sistemas Operativos

Laboratorio 3

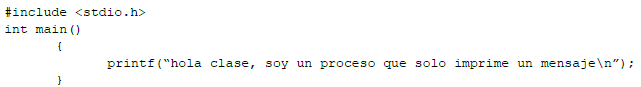
Procesos

# INTRODUCCIÓN

Los primeros computadores solo podían ejecutar un programa a la vez, y ese programa tenía control total del sistema y acceso a todos los recursos. Sin embargo, los sistemas informáticos modernos permiten cargar múltiples programas en la memoria y ejecutarlos de manera concurrente. Para manejar esto, se introdujo el concepto de “proceso”, que es un programa en ejecución. Los procesos son la unidad de trabajo en un sistema informático actual. Aunque su principal función es ejecutar programas de usuario, también se encargan de tareas del sistema operativo. En resumen, un sistema consiste en una colección de procesos que pueden ejecutarse simultáneamente en la CPU.

# ANTECEDENTES: PROCESO

Todo programa en ejecución es un proceso, por ejemplo:

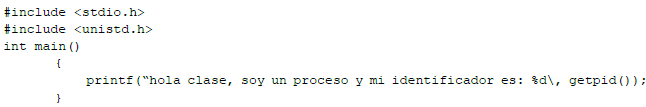


Una vez que se compila y ejecuta, este programa se volverá un proceso, dicho proceso será creado en la memoria principal con los datos necesarios para su ejecución, como un contador de programa, memoria para sus variables, etc. Usando esos datos en memoria, el proceso será capaz de ejecutar las instrucciones que componen al programa (en este caso un *printf*).

**SISTEMA OPERATIVO**

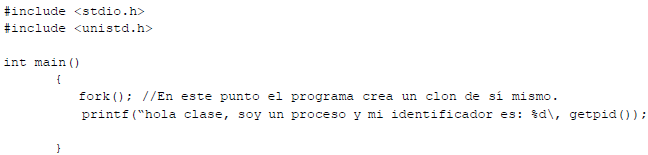
Todos los procesos son manejados por el sistema operativo, el cual los diferencia entre sí con un identificador único de proceso, o PID.

El PID lo podemos consultar mediante la función “getpid()” de la librería “unistd.h”, dicha función permitiría a un programa imprimir por pantalla su propio PID.



**Función fork**

Normalmente los procesos se inician al ejecutar los programas. Sin embargo, a veces surge la necesidad de que un proceso cree a su vez otro proceso independiente. Cuando llamamos a la función *fork*, se crea un clon idéntico del proceso que la ejecutó.



En principio había un solo proceso, al invocar a fork, se crea un proceso al que se le llama “hijo” idéntico al proceso original llamado “padre”. Lo único que los diferencia es el identificador de procesos (PID).

# OBJETIVOS

El objetivo de este laboratorio es que el estudiante sea capaz de interaccionar con el intérprete de comandos BASH utilizando C. Comprender el concepto de proceso, en el que se basa todo el funcionamiento de un sistema informático y comprender los diversos mecanismos relacionados con los procesos, incluyendo los de planificación, creación y finalización de procesos.

# Procesos (15pts)

**Muestre capturas de pantalla de los resultados que obtuvo de los códigos anteriores, si muestra errores, soluciónelo.**

**Texto

Descripción generada automáticamente**

**El código del primer ejemplo de proceso no presentaba errores y se ejecutó correctamente.**

**Texto

Descripción generada automáticamente**

**Texto

Descripción generada automáticamente**

**El código del segundo ejemplo de procesos relacionado a sistemas operativos tenia 2 errores de redacción relacionados a la falta de un “ *“* ” y la letra “ n ” para el “ *\n* “**

**Interfaz de usuario gráfica, Texto

Descripción generada automáticamenteEl código que crea un clon también presento errores en la redacción, pero se solucionaron.**

# Procesos con fork (10pts)

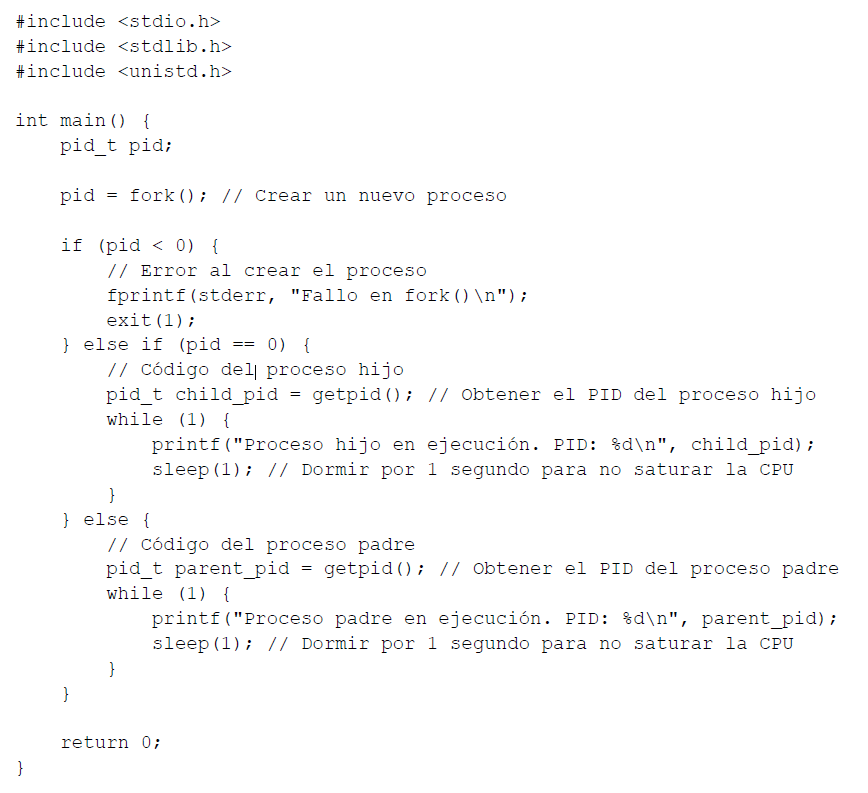
**Crear un proceso principal (padre) y que a su vez éste cree otro proceso (hijo) y que se identifiquen por su identificador de Proceso (PID).**

**Texto

Descripción generada automáticamente**

# Procesos con fork, while (40pts)

**El siguiente código crea un nuevo proceso con la función fork y entra en un bucle. Utilizando la herramienta htop investiga e identifica el proceso padre, hijo, huérfano y zombie. (agrega capturas de cada proceso identificado)**

****

**Texto

Descripción generada automáticamente**

**Texto

Descripción generada automáticamente**

**Aquí se observan 2 procesos, el padre y el hijo ejecutando en paralelo:**

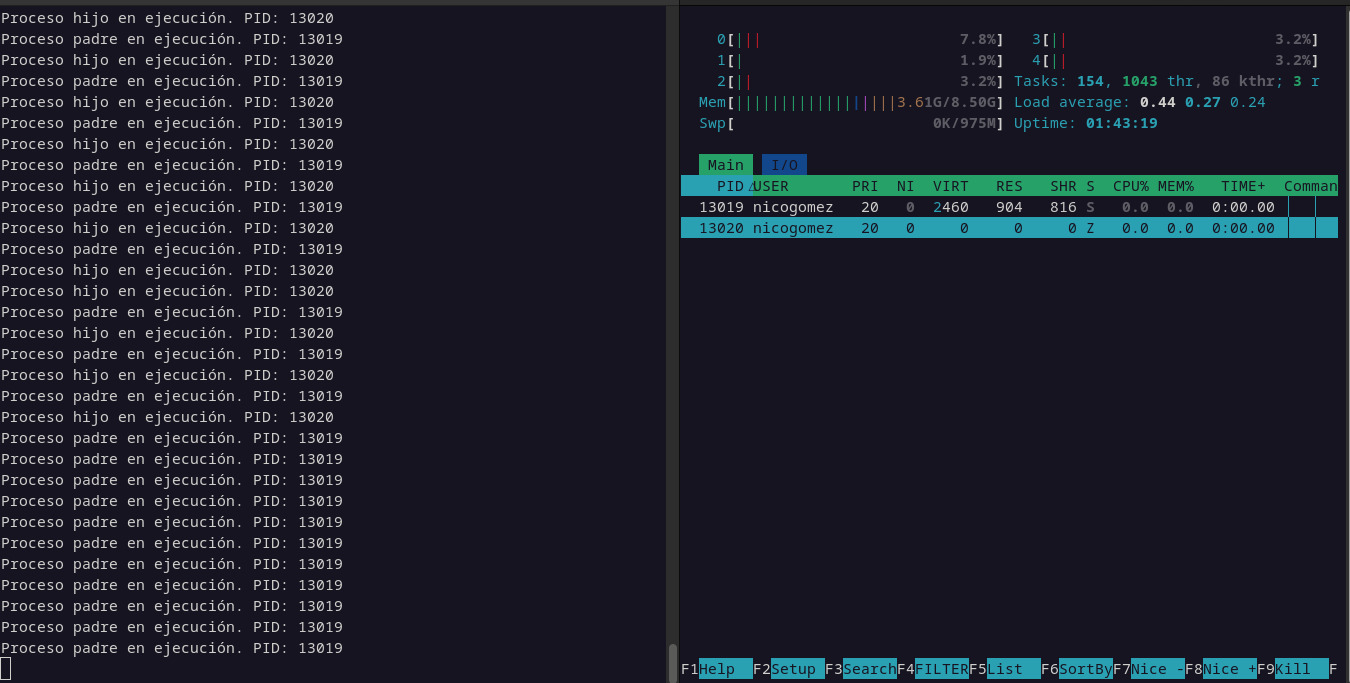
Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

**Un proceso se vuelve huérfano cuando el proceso padre que lo antecede es terminado, lo cual se realizó con la opción F9 de htop, provocando lo siguiente:**

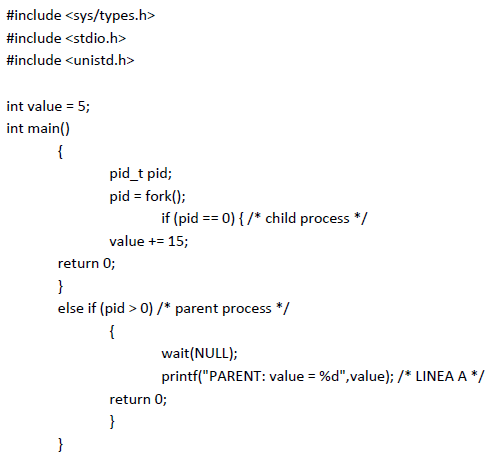
Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

**Un proceso se denomina zombi cuando ha finalizado su ejecución, pero su entrada en la tabla de procesos aún persiste porque el proceso padre no ha recogido su estado de salida mediante una llamada a wait(). En este caso, se utilizó el comando kill para intentar eliminar el proceso hijo, pero este no se eliminó completamente. Como se muestra en la imagen, solo el proceso padre continúa ejecutándose, mientras que el PID del proceso hijo aún aparece en htop como zombi.**

# Procesos (25pts)

**Usando el siguiente programa, explique cuál será la salida en la LINEA A. Si contiene errores soluciónelo.**

****

**Se predice que el valor de la salida será 5 ya que en el código entregado la alteración de la variable *value* solo se da en el proceso hijo, por lo que el proceso padre mantendrá el valor original.**

**Texto

Descripción generada automáticamente**

# Entrega (10 pts)

Documente (screenshots) cada una de las acciones antes señaladas. Es de exclusiva responsabilidad del estudiante respetar el formato de entrega de informe de esta guía **(debajo de cada enunciado su screenshot en donde aparezca de forma clara las sentencias utilizadas)**. El formato de entrega debe ser en PDF, y el nombre del archivo debe contener su nombre y apellido (**Laboratorio\_3\_Nombre\_Apellido**). Todas las actividades deben ser entregadas (subidas) a la plataforma digital en las fechas establecidas, por cada hora de atraso, se descontará 1 pto.