Segundo Parcial Sistemas Operativos

Apellido y Nombre: Flores Lucas Nicolás

Fecha: 24/10/2025

Materia: Sistemas Operativos

Docente: Villalba Carlos Lisandro

Carrera: Tecnicatura Superior en Desarrollo de Software (TSDS)

Año: 2025

Consigna

El presente examen corresponde al segundo parcial de la asignatura Sistemas Operativos correspondiente al primer año de la tecnicatura superior en desarrollo de software (TSDS).

Para el examen se solicita que entreguen el mismo vía Classroom además de que utilicen las computadoras de escritorio del aula. Se debe utilizar Google cloud Shell para poder ejecutar y probar el código fuente, pero de ninguna manera se podrá consultar a la inteligencia artificial, por ejemplo: Gemini.

El examen será anulado y retirado correspondiendo a nota 1 (uno) si y solo si ocurre una de las siguientes cuestiones:

* Uso de celular en horario de examen
* Uso de IA para resolver el examen

Además, para todas las actividades presentes se les solicita que

1. Ejecute todos los programas presentes
2. Captura la salida de pantalla (PID, PPID, mensajes, etc)
3. Entregue
   1. Código fuente de cada programa
   2. Capturas o evidencias de ejecución
   3. Una breve reflexión sobre cómo el sistema operativo gestiona los procesos

La entrega será en formato .zip ya que deberá contener el código fuente en C y las capturas y el análisis en un documento único

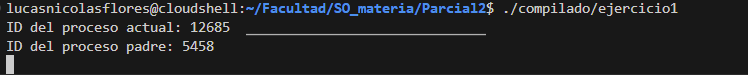
Los puntajes por cada respuesta correcta será la siguiente:

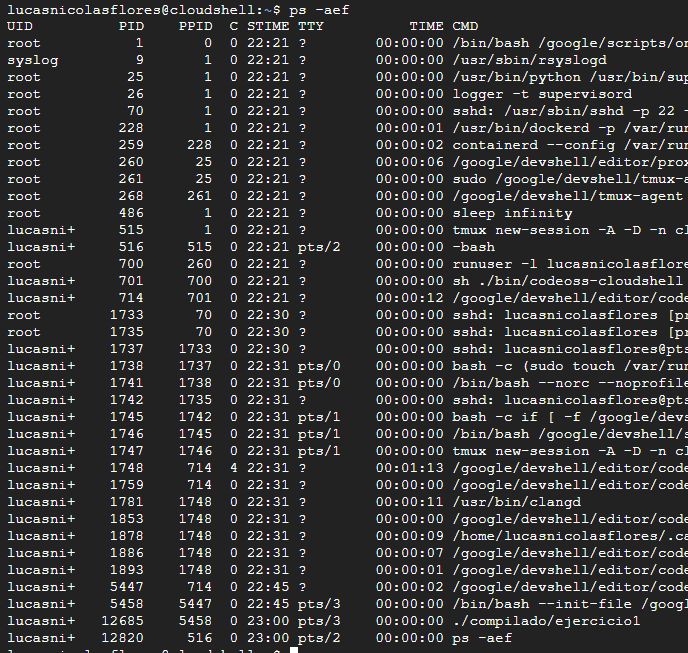
1. 3 puntos si la actividad 1 está correcta
2. 2,5 puntos si la actividad 2 está correcta
3. 2 puntos si la actividad 3 está correcta
4. 2,5 puntos si la actividad 4 está correcta

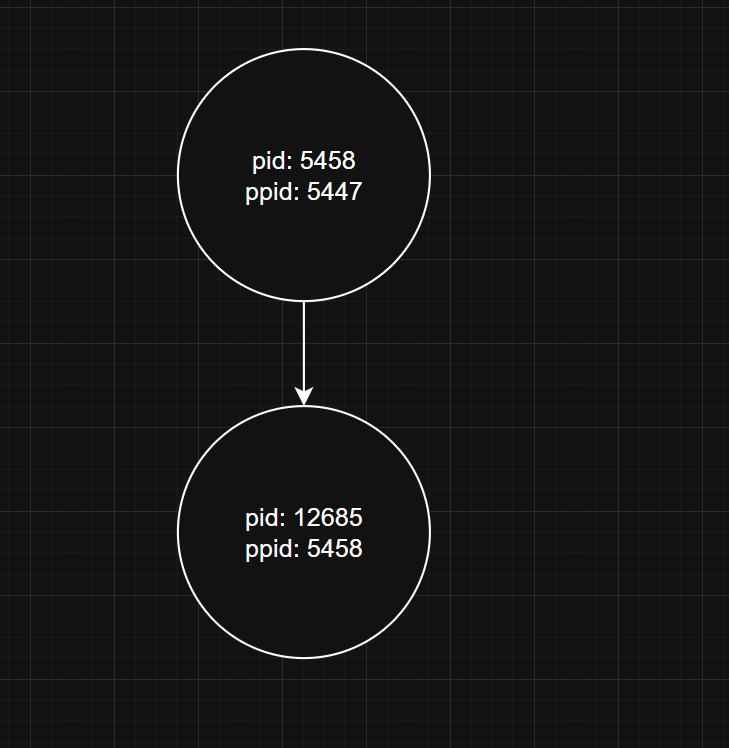
**Actividad 1**

Cree un programa en C llamado actividad1.c que:

1. Muestre el PID del proceso actual
2. Muestre el PPID del proceso actual
3. Incluya una pausa con getchar() antes de finalizar, con el propósito de permitir la verificación del proceso activo mediante otra terminal utilizando el comando ps -aef
4. Luego de ejecutarlo:
   1. Capture la salida del comando ps -aef donde se visualice su proceso
   2. A partir de esa información, dibuje el árbol de procesos, indicando claramente las relaciones padre->hijo mediante líneas o flechas
   3. Identifique (si hubiere) procesos intermedios





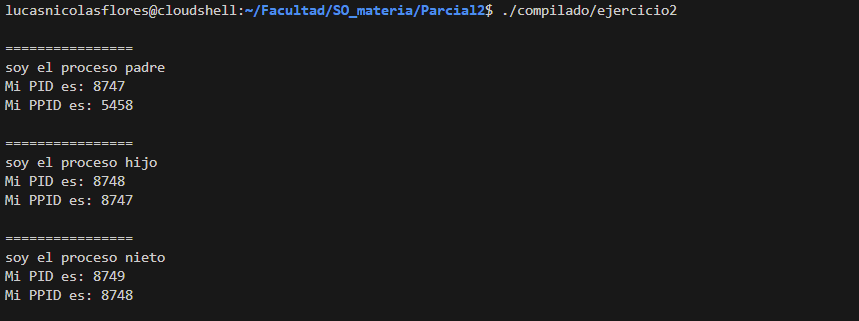


Tiene una gestión sencilla y directa ya que el fork que se realiza es (por tiempos de ejecución sumado a la máquina de shell.cloud.google) no hay procesos intermedios

**Actividad 2**

Cree un programa llamado actividad2.c que:

1. Genere un proceso hijo, y desde ese hijo genere un proceso nieto
2. Cada proceso debe imprimir un mensaje indicando
   1. Qué tipo de procesos es (padre, hijo, nieto)
   2. Su PID y PPID
3. El proceso padre debe esperar la finalización de ambos procesos

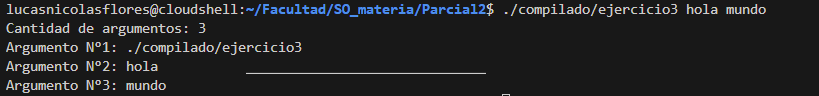


Acá se puede ver como la consigna nos designa esta espera de que el padre espere por sus hijos, evitando el tema de los procesos huérfanos o zombies, que de igual manera se deberían solucionar automáticamente por el SO al adoptar los procesos huérfanos para matarlos

**Actividad 3**

Cree un programa llamado actividad3.c que:

1. Reciba parámetros por línea de comandos
2. Muestre:
   1. La cantidad total de argumentos (argc)
   2. El contenido de cada uno (argv[i]xx
3. Si no se recibe ningún argumento, mostrar un mensaje de error y finalizar con exit(1)



Actividad 4

Cree un programa actividad4.c que

1. Cree tres procesos hijos
2. Cada proceso:
   1. Imprima su PID y PPID
   2. Duerma por distintos segundos (2, 4, 6)
   3. Finalice con exit(0)
3. El proceso padre:
   1. Espere la finalización de todos con wait()
   2. Imprima un mensaje confirmando la finalización total

