**Práctica 06**

**Comandos del Shell, variables, redirectores y pipes.**

**INDICACIONES SOBRE EL DESARROLLO**

En todos los puntos que sigue tendrá que explicar con detalle cómo logró llevar a cabo lo pedido, indicando trayectorias, comandos con su despliegue y / o acciones realizadas. En el caso de despliegue de comandos explique el significado de lo desplegado. Cuando haya preguntas específicas, no basta con aplicar y desplegar el resultado del comando, deberá usted contestarlas. Tendrá que abrir una terminal-ventana, en Ubuntu, y también puede ayudarse del File Manager.

**PREVIO AL INICIO**

* Copie el contenido del fólder *LabBF06* de Sistemas Operativos (SO) en Kepler. Este fólder contiene tanto el texto de la práctica como material de soporte.
* Después arranque la máquina virtual de Ubuntu y entre a la cuenta de “sisops”.

**ACTIVIDADES INICIALES**

Lea el archivo *UbuntuShell.pptx*, en formato de notas, que contiene el material de soporte de la práctica.

Desde su directorio base (*home*), copie el archivo  ***uctar1*** del URL [***ftp://ftp.itam.mx/pub/investigadores/rrios/SisOpe/Lab/PracBF06/***](ftp://ftp.itam.mx/pub/investigadores/rrios/SisOpe/Lab/PracBF06/)***.*** Esto lo puede hacer, ya sea desde su browser, o desde su computadora usando el comando *wget*. Una vez en el directorio base aplique el comando ***tar –xvf uctar1*** que descomprimirá el archivo ***uctar1*** en su directorio base creando el directorio (folder) ***uc1*** que contiene a su vez varios directorios y archivos.

**DESARROLLO**

1. En una primera *Terminal*, que titulara “UNO”, aplique el comando *ps -el*.

En una segunda *Terminal*, que titulará “DOS”, ejecute el comando *ps -l -u sisops*.

¿Explique cuál es la diferencia entre ambos comandos *ps*?\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Además muestre cuantos procesos despliega cada comando \_\_\_\_\_\_\_\_, \_\_\_\_\_\_. También muestre los comandos con los que obtuvo la cuenta de los procesos \_\_\_\_\_\_\_\_\_\_\_, \_\_\_\_\_\_\_\_\_\_\_.

1. Vea la pág. 4.16 (*UbuntuShell.pptx*, de la práctica pasada), Si él “.” (punto), en lugar de estar al principio del *path* estuviera al final ¿pudiera afectar el que se ejecute alguno de sus programas ejecutables, que se encuentran en el directorio de trabajo? Piense en un posible escenario en que pudiera no ejecutarse. Explique: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
2. Para recibir el aviso de que un evento ha ocurrido, los procesos recibir señales (signals), atrapándolas (como excepciones) o en caso contrario someterse a la acción por default del sistema. Estas señales pueden ser generadas por procesos del sistema, por otros procesos de usuario o mediante el comando *kill*. ¿Cuál es la utilidad del comando *kill*? \_\_\_\_\_\_\_
3. Despliegue la lista de las señales con el comando *kill -l* : \_\_\_\_\_\_\_\_\_\_\_\_

Por ejemplo la señal número 2, INT, es para que el proceso receptor sea INTerrumpido. La señal 15, TERM, es para provocar la TERMinación del proceso que la recibe.

¿Por cierto como se llama la señal 9? \_\_\_\_\_\_\_ y ¿cuál efecto causará sobre el proceso que la recibe? \_\_\_\_\_\_\_\_

1. Dentro de la ayuda de *kill* ¿cuál es la acción por default de las señales 2, 9 y 15 respectivamente? \_\_\_\_\_\_\_\_\_\_
2. El archivo *CiInfi.cc*, dentro de *uc1*, contiene un programa en C++, que cicla indefinidamente hasta que recibe ya sea la señal SIGINT (2 o INT) o la señal SIGTERM (15 o TERM).

Despliegue a continuación, el contenido el archivo \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

¿Cuál es el objetivo de las funciones *signal* y *Fin*.?\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Después, en una *Terminal* compile *CiInfi.cc* y ejecute *CiInfi.exe*.

Ahora, donde está corriendo este programa teclee la combinación de teclas ***<Control><c>***. ¿Qué pasa? ¿Qué imprimió el programa? ¿***<Control><c>***, que señal genera hacia el programa? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Mantenga las dos Terminales, en una de ellas deje ejecutando *CiInfi.exe*. Desde la otra terminal obtenga tanto el PID como el PPID del mismo *CiInfi.exe*. Indique cual es el proceso padre de *CiInfi.exe*. Para detener a *CiInfi.exe*, desde la otra terminal, ejecute el comando “*kill -2 PID*” (o “*kill -INT PID”*) que enviará la señal 2, o “*kill -15 PID*” la 15, que atrapará *CiInfi.exe* y detendrá su ejecución. Sólo reporte este inciso una sola vez.
2. Estudie las páginas de notas de la 4.26 a la 4.30. Al terminar su estudio, revise los archivos *EjemShell4a30.pdf* y *EjemShell4b30.pdf* que muestran con flujo la ejecución de los comandos de la pág. 4.30.
3. El programa *Desordenado.cc* (directorio *uc1*) genera una lista de números desordenados. Combina los comandos que conoces mediante ¿separadores “;”, redirectores “>, <”, pipe “|”, para que en una sola línea de comando, por inciso:
4. Genere un archivo con los números ordenados (comando de ordenamiento *sort*) \_\_\_\_\_\_\_
5. Despliegue cuántos números se generaron \_\_\_\_\_\_\_\_\_\_
6. Genere un archivo con los primeros diez números ordenados \_\_\_\_\_\_\_\_\_\_
7. Genere un archivo con los últimos diez números \_\_\_\_\_\_\_\_\_\_\_
8. Combina en una sola línea los comandos necesarios para crear un archivo, que contengan las líneas 7 a 15, a partir de lo generado por *Desordenado.exe* al ejecutarse. \_\_\_\_\_\_\_\_
9. El programa *sumon.cc* (directorio *uc1*) suma los *n* datos que se le agreguen en la lista de comandos (p.e. *sumon.exe dato1  dato2  ... daton* ). El programa *iva.cc* calcula el iva (16%) del monto que se le entregue como dato en lectura. Realice:
10. Compile y ejecute *sumon.cc* \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
11. Compile y ejecute *iva.cc* \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_
12. En una línea de comando haga que el resultado de *sumon.cc* sea pasado a *iva.cc*. \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Esta práctica deberá ser entregada en Comunidad ITAM.