SISTEMAS OPERATIVOS

TRABAJO DE PROGRAMACION 1

El objetivo de este primer programa es que apliquen las técnicas de administración de procesos en los sistemas operativos tipo POSIX y desarrollen programas multiproceso usando un lenguaje de programación que permita el uso de estas técnicas, implementando un Shell.

El shell o Intérprete de Comandos es la interface fundamental de un Sistema Operativo.

El programa a realizar para este primer trabajo es un shell simple que tenga las siguientes propiedades (indicándose entre paréntesis el porcentaje que aporta):

1. El shell debe soportar los siguientes comandos internos:
    a. **cd** [directorio] - cambia la ubicación actual a la indicado por el argumento directorio, si es que se proporcionó (los corchetes indican que es opcional). Si el argumento directorio no está presente, debe reportar (mostrar) el directorio actual. Si no existe el directorio entonces debe indicarse con un mensaje de error. Este comando debe también cambiar la variable de ambiente PWD para que almacene el directorio al que ha sido cambiado. El directorio puede consistir de una ruta relativa o una ruta absoluta donde cada elemento de la ruta es separado por el carácter /. Debe manejar correctamente los directorios especiales **.** y **..** (7.5% por el manejo de rutas relativas, 7.5% por el manejo de rutas absolutas, 2.5% por el manejo de errores y 2.5% por el manejo del comando cuando no se da directorio)
    b. **dir** [directorio] - lista el contenido (lista de archivos) del argumento `directorio` si es que se proporcionó. Si el argumento no está presente,, debe mostrar el contenido del directorio actual. Solo es necesario listar el nombre del archivo (20%, 5% extra si se da la siguiente información extra de cada archivo: tamaño y fecha de creación)
    c. **clr** – limpia la pantalla (5%)
    d. **environ** – lista las variables de ambiente seguidas de su contenido (5%). El ambiente del shell debe contener las siguientes variables:
        i. SHELL – esta debe contener la ruta absoluta a su programa ejecutable
        ii. PATH – esta contiene las rutas donde se buscaran los comandos externos, su valor debe ser /bin:/usr/bin:/sbin:/usr/sbin:.
        iii. HOME – esta contiene la ruta al directorio de usuario, la cual será inicializada al directorio desde donde se ejecute el shell
        iv. PWD – esta contiene el directorio actual
    e. **echo** comentario - despliega el comentario en la pantalla seguido de un avance de línea. El comentario puede consistir de texto con espacios. Debe poderse mostrar el contenido de las variables de ambiente cuando se les anteponga el símbolo $. (10% si solo despliega comentarios sin sustituir las variables de ambiente, 15% si las sustituye correctamente)
    f. **pwd** – indica el directorio actual (5%)

g. **quit** - sale del shell (5%)

2. Todas las otras entradas de comando serán interpretadas como una invocación de comandos externos ó programas ejecutables que deben ser realizadas usando las llamadas al sistema ***fork*** y ***exec*** para crear un proceso hijo que ejecute el programa invocado (ó con cualquier llamada al sistema similar que exista en el lenguaje de su elección, también si desean lo podrían hacer a nivel de hilos en vez de procesos). En caso de que el comando externo no exista debe indicarlo con un mensaje de error. Estos comandos externos deben manejar los argumentos que se les pasen. Como referencia del uso de fork y exec consulte el código en https://repl.it/@rsolisuaz/SisOp17Sep (20%)
3. El indicador del shell debe contener la ruta del directorio actual, seguida de un espacio y un **>** (5%)
4. El shell debe ser realizado para funcionar bajo ambiente UNIX/Linux bajo el ambiente POSIX.
5. Se debe usar GitHub para estar subiendo sus versiones al repositorio remoto (15% extra si se realizan commits y push de acuerdo a lo indicado en los requerimientos)

REQUERIMIENTOS DEL TRABAJO DE PROGRAMACION

1. Clone el repositorio que se creó al aceptar esta tarea usando el comando git clone URL (donde URL es el URL del repositorio) en una terminal o ventana de comandos
2. Su código fuente puede constar de varios archivos pero el nombre del archivo fuente principal debe ser **AXXXXXXXX** con la extensión que corresponda al lenguaje de programación que utilice y donde las X se sustituyen por su matrícula a 8 dígitos. Cree este archivo (y los otros archivos que pudiera necesitar en la carpeta que se creó al clonar el repositorio. Se proporciona como guía un archivo denominado base_shell.c que muestra una posible estructura de lo que sería el código, NO ES NECESARIO QUE USE TAL CÓDIGO COM BASE O QUE ESCRIBA SU SOLUCIÓN EN C. SOLO SE PROPORCIONA COMO REFERENCIA.
3. El código fuente DEBE ser extensamente comentado y apropiadamente estructurado para permitir a alguien más entender y mantener fácilmente el código. Al principio del código debe incluir un comentario donde se indique el nombre y matricula del autor.
4. En una terminal (o ventana de comandos), colóquese en la carpeta del repositorio y emita los comandos:

   ```
   git add .
   git commit -m "Mensaje"
   ```

   donde Mensaje es una descripción (clara y evitando ser demasiado extensa) de lo que se ha realizado desde el último commit, por ejemplo, para el primer commit (que debería hacerse después de crear el archivo mencionado en el paso 2 y colocando en él su matrícula y nombre en un comentario) el mensaje pudiera ser ***Inicialización de código***

   ```
   git commit -m "Inicialización de código"
   ```
5. Emita el siguiente comando para subir el código al repositorio remoto:

   ```
   git push
   ```
6. Emita un commit después de cada cambio significativo a su código, asignándole un mensaje que deje claro que se realizó desde el commit anterior, y haga push

frecuentemente. El código fuente deberá estar finalizado en el repositorio remoto a más tardar el lunes 12 de octubre del 2020 a las 11:59 PM.

7. Recuerde que el hecho de entregar un programa copiado resulta en la reprobación automática del curso. El programa debe ser realizado de manera INDIVIDUAL. También tome en cuenta que en el primer o segundo examen parcial se le pedirá realizar un programa sencillo, el cual de no poder ser resuelto resultará en la anulación del programa. SI USTED REALIZA ESTE TRABAJO DE PROGRAMACION DEBE PODER RESOLVER EL PROGRAMA DEL EXAMEN PUES SERA CON RESPECTO A ALGO QUE DEBE MANEJAR EN EL TRABAJO DE PROGRAMACION 1.
8. No puede hacer uso de funciones tal como ***`system`*** (ó cualquier otra similar) para realizar el trabajo que deben hacer los comandos internos, con la única posible excepción del comando ***clr***. Esto significa que los comandos internos deben ser manejados con código escrito por ustedes, no por comandos ya proporcionados por el sistema operativo subyacente. **Si no usa tales funciones TAMPOCO para `clr`, obtendrán 10 puntos extra**.