SISTEMAS OPERATIVOS II

Tercer curso Ingeniería Técnica Informática de sistemas. Curso 2006-2007

Práctica 2: Procesos en UNIX.

Comenzar la codificación de un intérprete de comandos (shell) en UNIX, que se irá completando en suscesivas prácticas. Dicho intérprete incluirá de momento los comandos que se citan a continuación. Los argumentos entre corchetes [] son opcionales. Los argumentos separados por `|` indican que debe ir uno u otro, pero no ambos simultaneamente. No debe dilapidar memoria (ejemplo: variable que se asigna cada vez que se llama a una función y no se libera). Cuando el shell no pueda ejecutar una acción por algún motivo, debe indicarlo con un mensaje como el que se obtiene con `sys_errlist[errno]` o con *perror()* (por ejemplo, si no puede cambiar de directorio debe indicar por qué). En ningún caso debe producir un error de ejecución. Las direcciones de memoria deben mostrase en **hexadecimal**. La información que se nuestra en pantalla no debe incluir en ningun caso líneas en blanco. El shell leerá de su entrada estándar y escribirá en su salida estándar, de manera que podría ser ejecutado un archivo de comandos invocando el shell con su entrada estándar redireccionada a dicho archivo.

autores Indica los nombres y los logins de los autores de la práctica.

fin Termina la ejecución del intérprete de comandos.

exit Termina la ejecución del intérprete de comandos.

chdir [dir] Cambia el directorio actual a *dir*. Si no se le suministra argumento informa del directorio actual.

prompt *prompt* Cambia el indicador (prompt) del intérprete de comandos.

- Los siguientes comandos manipulan la ruta de búsqueda del intérprete de comandos.
  La ruta de búsqueda del intérprete de comandos es el conjunto de directorios donde se buscan los **ejecutables** (Su misión es análoga a la variable de entorno PATH en el shell del sistema). NO DEBE implementarse con una variable de entorno. Los directorios se especificarán sin el caracter '/' al final Ejemplos

```
#pathadd /usr/local/bin /* correctos */
#pathadd /
#pathdel .
......
```

```
#pathchk /usr/local/bin/ /*incorrecto*/
#pathdel ./              /*incorrecto*/
```

pathadd [**dir**] Añade *dir* a la ruta de búsqueda.

pathdel [**dir**] Elimina *dir* de la ruta de búsqueda.

pathchk [**dir**] Indica si *dir* está en la ruta de búsqueda.

pathempty Vacía la ruta de búsqueda.

pathpath Añade los directorios de la variable de entorno PATH a la ruta de búsqueda.

path Muestra la ruta de búsqueda.

Si no se especifica *dir* en *pathadd, pathdel* o *pathchk*, se mostrará la ruta de búsqueda.

whereis ejecutable Busca *ejecutable* en la lista de directorios que constituyen la ruta de búsqueda del intérprete de comandos y devuelve la trayectoria completa hasta él (análogo al comando *which* del sistema). Si no lo encuentra lo indicará. Si se le indica una trayectoria completa (empezando por ., por / o por ..) indicará si dicho fichero existe o no

```
#pathadd /usr/openwin/bin
#whereis xterm
/usr/openwin/bin/xterm
#whereis xternp
 xternp: no encontrado
#whereis ./aa.out
 ./aa.out: no encontrado
#whereis ./a.out
 ./a.out
```

getpid Devuelve el pid del proceso y de su proceso padre.

fork El intérprete de comandos crea un hijo con fork() y se queda (el propio intérprete de comandos) en espera hasta que dicho hijo (el creado con fork) termine.

exec *com* El intérprete de comandos ejecuta, sin crear un nuevo proceso (reemplaza su código), el programa especificado en *com*. *com* representa un ejecutable con sus parámetros. Para poder ser ejecutado, dicho ejecutable debe residir en uno de los directorios de la ruta de búsqueda del intérprete de comandos o bien especificarse la trayectoria completa hasta él (comenzando por /, . o ..). Debe usarse la llamada al sistema

execv.
Ejemplo

```
#exec /usr/bin/ls -l /home  /*ejecuta /usr/bin/ls*/
  ....
#exec ./a.out -l /*ejecuta a.out,
                 /*que esta en el directorio actual */
......
#exec ls -l   /* para ejecutar esto, es necesario */
              /*que el directorio  /usr/bin,  */
              /*donde esta el programa ls, se haya incluido */
              /* a la ruta de busqueda */
```

com Totalmente análogo al comando exec, salvo que el shell crea el proceso que ejecuta dicho comando y espera a que termine (la ejecución es, por tanto, en primer plano). Todo lo dicho sobre los ejecutables en el comado *exec* es aplicable aquí. ***com* representa un ejecutable junto con sus parámetros.**
Ejemplo

```
#/usr/bin/ls -l /home  /*crea un proceso que ejecuta /usr/bin/ls*/
....
#./a.out -l   /* crea un proceso que ejecuta a.out,*/
......        /* que esta en el directorio actual    */
#ls -l        /* para ejecutar esto, es necesario que el directorio */
              /* /usr/bin, donde esta el programa ls, se haya incluido */
              /* a la ruta de busqueda */
```

.

background *com* Análogo a *com* pero la ejecución es en segundo plano (el shell no espera a que el proceso termine). Ejemplos

```
#background xterm -e csh
```

crea un proceso que ejecuta xterm en segundo plano.

recursiva [-a|-n|-d] n Invoca a la función recursiva n veces, la opción -a, -n o -d indica si la memoria asignada con malloc en dicha función debe liberarse (a)ntes de la siguiente llamada recursiva, (d)espués o (n)o liberarse. Si no se indica una opción se supone que la memoria se libera despues de la llamada (-d). La función recursiva recibe dos parámetros: uno que indica el número de veces que se tiene que invocar y otro que indica cuando liberar la memoria asignada con malloc; tiene 3 variables, un array automatico de 512 caracteres, un array estático de 512 caracteres

y un puntero a caracter.
Esta función debe hacer lo siguiente

1. asignar memoria (mediante malloc) al puntero para 512 caracteres.
2. imprimir
   - el valor y la dirección del parámetro que recibe.
   - el valor y la dirección del puntero.
   - la dirección de los dos arrays (el nombre del array como puntero).
3. si se ha especificado la opción -a liberar la memoria asignada al puntero.
4. invocarse a si misma con (n-1) como parámetro (si n>0).
5. si se ha especificado la opción -d liberar la memoria asignada al puntero.

Un posible código pra la función recursiva (sin las definiciones de constantes)podría ser:

```
void recursiva (int n, int liberar)
{
char automatico[TAMANO];
static char estatico[TAMANO];
void * puntero;

puntero=(void *) malloc (TAMANO);
printf ("parametro n:%d en %p\n",n,&n);
printf ("valor puntero:%p en direccion: %p\n", puntero,&puntero);
printf ("array estatico en:%p, array automatico en %p\n", estatico,automatico);
if (liberar==ANTES)
   free (puntero);
if (n>0)
   recursiva(n-1,liberar);
if (liberar==DESPUES)
   free (puntero);
}
```

direcciones [-f|-v] Imprime las nombres y las direcciones de las funciones que implementan los comandos del shell (-f), de las variables globales (-v). Si no se especifica ni -f ni -v imprime las direcciones de las funciones y de las variables globales.

Información detallada de las llamadas al sistema y las funciones de la librería debe obternerse con man (exec, chdir, fork, strtok, waitpid, getpid..)

FORMA DE ENTREGA Va a ser utilizado el servicio de recogida de prácticas suministrado por el Centro de Cálculo de esta Facultad y parte del proceso de corrección de las prácticas va a ser automt́ico (compilación, listado de practicas entregadas etc) por lo cual deben entregarse **exactamente** como se indica a continuación:

- Se colocará el código fuente de la práctica en el directorio asignado para ello antes de la fecha tope de entrega de la práctica.
- Se entregará UN SOLO fichero fuente por práctica, de nombre pN.c (N el número de práctica). Por ejemplo, para esta práctica será p2.c (en minúsculas).
- Los grupos de prácticas son de **2 (DOS)** alumnos. La práctica SOLO DEBE SER ENTREGADA POR UNO DE LOS MIEMBROS DEL GRUPO
- en el código fuente de la práctica debe figurar como comentario el nombre de los autores **exactamente** en el siguiente formato

  ```
  /*
  AUTOR:apellido11 apellido12, nombre1:login_del_que_entrega_la_practica
  AUTOR:apellido21 apellido22, nombre2:login_del_que_entrega_la_practica
  */
  ```

  donde:

  1. La palabra autor aparece en mayúsculas.
  2. Los apellidos y el nombre de los autores están totalmente en minúsculas.
  3. apellidoij representa el apellidoj del componente i del grupo de prácticas.
  4. No hay espacios antes y despues de los dos puntos.
  5. El login que aparece es el del que entrega la práctica (aparece el mismo login en las dos líneas).
  6. Los símbolos de comentarios están en líneas distintas.

FECHA DE ENTREGA VIERNES 1 DICIEMBRE 2006