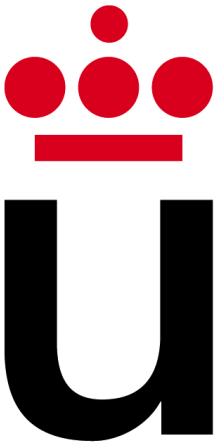

Universidad Rey Juan Carlos

Escuela Técnica Superior Ingeniería Informática

Sistemas Operativos

PRÁCTICA 2: MINISHELL

GIS+GII Luis León Gámez

GIS Carlos Vázquez Sánchez

Móstoles - 13 de diciembre de 2014

Índice general

Capítulo 1

Código

1.1. Funciones Auxiliares

1.1.1. void senal (int s)

Función que se encarga de activar o desactivar las señales SIGINT y SIGQUIT, de tal modo que en caso de tratarse de un proceso en background o de la propia minishell, no respondan. Sin embargo no se activarán si es un proceso foreground.
Su funcionamiento es muy sencillo: usando la función *signal* si se le pasa el parámetro 0 las desactiva, y si es un 1 las activa

1.1.2. void comandoCD(char *ruta)

Mediante la función *chdir* cambia el directorio de trabajo al especificado en "ruta". En caso de no especificar una ruta, accede a HOME. En caso de introducirse una ruta inexistente avisará del error.

1.1.3. int** crearPipes (int n)

Función que se encarga de crear las tuberías necesarias. Crea $n-1$ tuberías, ya que para el último hijo no es necesario, al ser éste el que ejecuta el último mandato.

1.1.4. void liberarTodasPipes(int** pipes,int n)

Método que cierra todas las tuberías usadas hasta ese momento.

1.1.5. int redirecSalida (char* nombreFichero)

En caso de que se haya introducido una orden con el símbolo > y se trate del último comando, se activará la redirección por salida. Para ello se creará un fichero con el nombre especificado con permisos de lectura y escritura

para todo el mundo, y si no hay error al crearlo copiará la salida estándar al fichero.

1.1.6. int redirecEntrada (char* nombreFichero)

En caso de que se haya introducido un comando con el símbolo < y se trate del primer mandato, se activará la redirección por entrada. Para ello abre el fichero con el nombre especificado con permisos de solo lectura, y si no hay error al leerlo, copiará el contenido del fichero a la entrada estándar.

1.1.7. int redirecError (char* nombreFichero)

Se encarga de comprobar que no hay error al crear o leer un fichero en caso de ser necesaria una redirección.
Su funcionamiento es muy similar a *redirecSalida*, pero a la hora de llamar a la función *dup*2 lo hacemos redirigiendo al error en vez de a la salida.

1.2. Main

Sigue un esquema similar al siguiente
ESQUEMAAAA
Explicamos muy brevemente su funcionamiento: en primer lugar comprueba que se haya introducido una orden (a excepción de "cd", cuyo caso es diferente). A continuación analiza si es el último mandato, en cuyo caso imprimirá el resultado. Si no es el último, gracias a la función *dup*2 copiamos su contenido, y mediante tuberías se lo pasamos al siguiente proceso. Finalmente se realizarán las redirecciones en caso de ser necesarias.

Capítulo 2

Comentarios personales

2.1. Problemas encontrados

AL igual que la práctica anterior, hemos ido añadiendo funcionalidades de manera incremental. De esta manera en un principio comprobamos el correcto funcionamiento al ejecutar un solo comando. A continuación probamos que redireccionara la salida y la entrada correctamente, después añadimos la funcionalidad para el comando "cd". Finalmente añadimos la comunicación entre procesos. Este último paso fue el que más problemas nos acarreó, ya que en un principio no poseíamos los conocimientos teóricos suficientemente claros y no sabíamos como debían crearse los procesos (se crean hijos sucesivos, o se crean hijos hermanos). Además nos dieron muchos problemas las tuberías, especialmente en el último proceso, ya que no se cerraba correctamente. Por ello tuvimos que añadir el método *cierraTodasPipes* que cierra todas las tuberías anteriores, asegurándonos así que no nos cejábamos algo abierto.

2.2. Posibles mejoras

Debido a la falta de tiempo no hemos podido comprobar las fugas de memoria, así que no sabemos cual es la calidad de nuestro código en este aspecto. Por otra parte, se podrían continuar añadiendo funcionalidades a la shell hasta conseguir una similar a la original. Podríamos por tanto añadir antes del promp el usuario que ha ejecutado la terminal, o añadir la funcionalidad del autocompletado con la tecla tabulación, que facilita enormemente el uso de la terminal cuando hay que escribir rutas largas o tratar con nombres de archivos largos. Siguiendo con ampliaciones que mejoran el manejo de la shell, usar las teclas de arriba y abajo para acceder al historial de comandos introducidos
Por otro lado, la estructura de nuestro método *main* main no nos parece la más eficiente posible, ya que tiene un número muy alto de ramificaciones

y, francamente, es complicado de entender hasta para nosotros. Por tanto nuestra primera prioridad si dispusiéramos de más tiempo sería cambiar la estructura del main, reduciendo todo lo posible las ramificaciones y añadiendo más métodos auxiliares que ayuden a la comprensión del código.