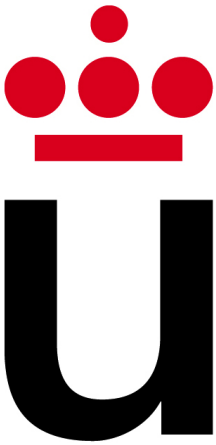

Universidad Rey Juan Carlos

Escuela Técnica Superior Ingeniería Informática

Sistemas Operativos

PRÁCTICA 2: MINISHELL

GIS+GII Luis León Gámez

GIS Carlos Vázquez Sánchez

Móstoles - 11 de diciembre de 2014

Índice general

1. Código **3**
1.1. Funciones Auxiliares 3
1.1.1. void senal (int s) 3
1.1.2. void comandoCD(char *ruta) 3
1.1.3. int** crearPipes (int n) 3
1.1.4. void liberarPipes(int** pipes,int n) 3
1.1.5. int redirecSalida (char* nombreFichero) 3
1.1.6. int redirecEntrada (char* nombreFichero) 4
1.2. main 4

2. Comentarios personales **5**
2.1. Problemas encontrados 5
2.2. Posibles mejoras 5

Capítulo 1

Código

1.1. Funciones Auxiliares

1.1.1. void senal (int s)

Fución 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.

1.1.2. void comandoCD(char *ruta)

Cambia el directorio de trabajo a la especificada en "ruta". En caso de no especificar una ruta, accede a HOME.

1.1.3. int** crearPipes (int n)

Función que se encarga de crear las tuberías necesarias. Crea $n-1$ tuberías ¿¿¿¿¿¿¿¿¿¿SEGURO??????????.

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

HAY QUE ACABARLO

1.1.5. int redirecSalida (char* nombreFichero)

En caso de que se haya introducido un comando 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, 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, y si no hay error al leerlo, copiará el contenido del fichero a la entrada estándar.

1.2. main

HAY QUE ACABARLO

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 redireccionará 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 teníamos muy clara como debían crearse los procesos (se crean hijos sucesivos, o se crean hijos hermanos).

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. Para ello se podría por ejemplo añadir antes del promp el usuario que ha ejecutado la terminal. También podría añadirse 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.
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 hasta para nosotros. Por tanto esta sería nuestra primera prioridad si dispusiéramos de más tiempo .