SUMMARY – 19/04/2024

Implementa un servicio para calcular estadísticas sobre ficheros. Este servicio estará compuesto por un servidor que calculará estadísticas sobre los ficheros que los clientes le envíen.

Servidor

El servidor calculará las estadísticas que los clientes le envíen. En concreto, tiene que calcular las siguientes estadísticas:

-número de ficheros procesados

-número de carácteres procesados

-número de líneas procesadas

Para ejecutar el servidor se hará desde la línea de comandos utilizando la sintaxis siguiente:

```
> ./server port
```

Donde *port* es el número de puerto que utilizarán los clientes para enviar ficheros. Este puerto tiene que estar configurado para que utilice una conexión TCP.

La estructura interna del servidor es la siguiente. Existirá un thread principal que será el responsable de crear e inicializar todos los recursos necesarios y aceptará las conexiones de los clientes. Además, habrá un pool de threads secundarios (por ejemplo, 10 threads) que procesarán las peticiones recibidas por el thread principal. El thread principal y los secundarios se conectarán utilizando una estructura (decide tu cual será) compartida.

Además, el servidor, cada vez que reciba el signal SIGUSR1 mostrará por el terminal las estadísticas actualizadas hasta ese momento. Si el servidor recibe SIGUSR2, reiniciará las estadísticas.

El servidor finalizará cuando reciba el signal SIGKILL.

Cliente

El cliente será el responsable de enviar uno o más ficheros al servidor. Pueden existir más de un cliente ejecutándose simultáneamente.

La sintaxis para ejecutar el cliente es:

```
> ./client port file [list_of_files]
```

Como parámetros, el cliente se tiene que ejecutar con, al menos, un nombre de fichero *file* aunque puede tener más de uno en *list_of_files*.

El cliente creará tantos procesos hijos como *files* tenga y cada uno de estos procesos hijos se conectará con el servidor usando el parámetro *port* como puerto de conexión para enviar el fichero correspondiente. Una vez haya enviado todo el fichero, el proceso hijo se desconectará y acabará.

En caso de que el fichero como parámetro no exista o no se pueda abrir, tiene que mostrar un error.

El proceso padre tiene que esperar la finalización de todos los procesos hijos para terminar. Cuando termine, tiene que mostrar el número de bytes enviados en total y el tiempo que ha tardado en enviarlos.

Detalles finales:

1.- Siempre que se pueda, se tendrán que liberar los recursos que se han cogido

2.- Se tienen que mostrar con textos claros que errores se producen

3.- El código tiene que ser lo más claro y eficiente posible

4.- Se tienen que reducir, en la medida de lo posible, el número de llamadas al sistema que se ejecutan.