**Sistemas Operativos**

Práctica 1

David Ángel Leo Acedo (GIS) y Silvia Moreno Uribarri (GIS+MAT)

**Extracto**

Diseño de una librería en C que muestra N líneas por la salida estándar

Tabla de contenido

Introducción 3

Arquitectura 3

Descripción de la solución 3

Head 3

Tail 3

LongLines 4

Problemas encontrados 4

Aspectos a mejorar 4

# Introducción

Ya que no tenemos experiencia previa en C optamos por la modalidad uno de la práctica. Nuestra solución consta de un programa principal llamado “test” al que se le pasarán al invocarlo dos parámetros, la función que se desee ejecutar (head, tail o longlines) y el número de líneas que debe devolver la función invocada (siempre positivo). Tal como indicaba el enunciado los requisitos de las funciones son:

* Head (número): Escribe por salida las primeras n líneas introducidas
* Tail (número): Devuelve las n últimas líneas introducidas.
* Longlines (número): Devuelve las n líneas más largas introducidas ordenadas descendentemente por tamaño.

# Arquitectura

Para resolver este problema hemos organizado el código en un programa principal (test.c) que tiene como función probar la librería, verdadero objetivo de la práctica. Esta librería contiene las funciones principales detalladas arriba y una pequeña función auxiliar (LeerCola (cola)) que detallaremos más adelante.

Por otro lado tenemos dos ficheros con sendas estructuras de datos (colastruc y liststruc) en las que se apoyan tail y longlines respectivamente. Sólo tienen la funcionalidad más básica posible, y, aunque hechos ad hoc para las funciones que lo requerían pueden adaptarse fácilmente a otros problemas de ser necesario.

# Descripción de la solución

## Head

Es la función más sencilla de todas. Simplemente, tantas veces como indique el parámetro m se escribe por pantalla la linea introducida por el usuario. El grueso del código sólo comprueba que no haya habido un error de E/S.

## Tail

Tail se apoya en una estructura de datos de cola, es decir, una lista que la que la entrada se hace por el principio y la salida por el final. Cada vez que el usuario introduzca una línea, Tail lo encola y comprueba si ya ha superado el tamaño designado en el parámetro. Si es así, descola la línea más antigua.

La cola se ha implementado como una lista dinámica, con métodos propios que manejan los punteros de acuerdo a la lógica de la cola. Su implementación puede encontrarse en colastruc.c.

Finalmente se lee el contenido de la cola al dejar el usuario de escribir mediante la funcion “leerCola”. La estructura se elimina en el subproceso con “eliminarCola”.

## LongLines

Longlines utiliza en este caso una lista dinámica ordenada. Cada vez que el usuario introduce una línea, longlines llama a la función “addInOrder” que inserta la línea en el lugar que le corresponde dentro de la lista. En cada iteración, si la lista es mayor de lo que el parámetro indica, llama a rmv(r), que elimina el elemento número r+1 de la lista (que debería ser, por tanto, el útimo y el menor de ésta). Se recorre la lista resultante escribiéndola por pantalla, y finalmente se elimina la memoria explícitamente con “rmvList”.

# Problemas encontrados

La práctica no era complicada de realizar, pero nuestra inexperiencia con C significó que muchos pasos triviales para una persona mínimamente acostumbrada al lenguaje supusiesen dificultad para nosotros. Principalmente los problemas se originaron en el manejo de los punteros, no tanto por el concepto que ya era conocido, si no por su sintaxis.

Consideramos que haber sabido desarrollar tests en C hubiera facilitado la identificación de errores y por tanto habernos ahorrado mucho tiempo. Debido a las dimensiones reducidas del problema era sin embargo asequible realizar pruebas más o menos fiables a mano con GDC.

# Aspectos a mejorar

Al ser el desarrollo parte de nuestro proceso de aprendizaje, llegado el final del proyecto hay múltiples aspectos que podrían mejorarse o deberían serlo. En primer lugar las decisiones de diseño deberían ser consistentes (¿hacer E/S en funciones auxiliares? ¿usar una función auxiliar para leer la lista?). También podríamos haber implementado las tres soluciones usando un mismo tipo de estructura de datos auxiliar, decisión que no tomamos para practicar algo más con punteros.

La eficiencia tampoco ha sido nuestro principal objetivo en el desarrollo y se nota en sitios como la función longlines, en el que en vez de comprobar si un input es susceptible de entrar entre los mayores lo insertamos y luego lo eliminamos. Sin embargo, dada la limitada funcionalidad de la librería y la potencia de las máquinas actuales éste no es un problema demasiado importante.