

Sistemas Operativos

[Práctica 1 – PROGRAMACIÓN C]

Miguel Quero y Adrián Dueñas

Tabla de contenido

[Autores 2](#_Toc115812632)

[Descripción del Código 3](#_Toc115812633)

[Diseño del Código 3](#_Toc115812634)

[Principales Funciones 3](#_Toc115812635)

[Casos de Prueba 3](#_Toc115812636)

[Comentarios Personales 4](#_Toc115812637)

# Autores

**Los autores de la realización de esta memoria y de los ficheros entregados somos Adrián Dueñas Mínguez y Miguel Quero Prieto. Los dos estudiamos en la Universidad Rey Juan Carlos, Adrián estudia Ingeniería del Software y Miguel estudia un Doble Grado de Ingeniería Informática e Ingeniería del Software. Ambos nos encontramos en el tercer curso.**

# Descripción del Código

**<< MÁXIMO 4 PÁGINAS >>**

## Diseño del Código

**<< EXPLICACIÓN DEL DISEÑO DEL CÓDIGO:**

* **ALGORITMOS UTILIZADOS PARA CADA FUNCIONALIDAD PEDIDA**
* **ESTRUCTURA DEL PROGRAMA TEST**

**>>**

**<< NO INCLUIR CÓDIGO FUENTE >>**

**<< SE PUEDE UTILIZAR PSEUDOCÓDIGO O DIAGRAMAS DE APOYO A LA EXPLICACIÓN >>**

* **Función int head(int N):** el diseño utilizado para esta función es muy simple. El argumento indica las líneas que se van a mostrar por pantalla según la entrada del usuario.Dentro de la función utilizamos “input”, un puntero que apunta al comienzo de un bloque de memoria asignado, para almacenar la entrada escrita por el usuario. Para mostrar por pantalla el texto escrito por usuario utilizamos un bucle “for” que decrementa en uno el número de líneas hasta que llega a cero, dentro del bucle se recoge la entrada del usuario mediante la función “fgets” evitando fácilmente los desbordamientos de búfer. Después, se muestra por pantalla el valor de input con la función “printf”. Por último, se libera la memoria dinámica de input mediante la función “free” y se devuelve el valor cero indicando que no se ha producido ningún tipo de error.
* **Función int tail(int N):**
* **Función int longlines(int N):**
* **Estructura del programa test:**

## Principales Funciones

**<< DESCRIPCIÓN DE LAS PRINCIPALES FUNCIONES IMPLEMENTADAS INCLUYENDO LA FUNCIÓN PRINCIPAL, MAIN>>**

**<< UTILIZAR EL FORMATO DE LA SIGUIENTE TABLA PARA CADA FUNCIÓN >>**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Head** | **Nombre** | **Tipo** | **Descripción** |
| **Argumentos** | Argumento 1 | **N** | **Entero** | **Indica el número de líneas a leer** |
| **Variables Locales** | Variable 1 | **Input** | **Puntero a un carácter** | **Almacena en memoria dinámica la entrada escrita por el usuario** |
| **Valor Devuelto** | **0** | **0** | **Entero** | **Indica que no se ha producido ningún error al terminar la ejecución** |
| **Descripción de la Función** | **Muestra las N primeras líneas en la salida estándar recibidas por la entrada estándar** |  |  |  |

## 

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Tail** | **Nombre** | **Tipo** | **Descripción** |
| **Argumentos** |  |  |  |  |
| **Variables Locales** |  |  |  |  |
| **Valor Devuelto** |  |  |  |  |
| **Descripción de la Función** |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Longlines** | **Nombre** | **Tipo** | **Descripción** |
| **Argumentos** |  |  |  |  |
| **Variables Locales** |  |  |  |  |
| **Valor Devuelto** |  |  |  |  |
| **Descripción de la Función** |  |  |  |  |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Main** | **Nombre** | **Tipo** | **Descripción** |
| **Argumentos** |  |  |  |  |
| **Variables Locales** |  |  |  |  |
| **Valor Devuelto** |  |  |  |  |
| **Descripción de la Función** |  |  |  |  |

## Casos de Prueba

**<< LISTA Y BREVE EXPLICACIÓN DE CASOS DE PRUEBA UTILIZADOS PARA VALIDAR LA PRÁCTICA >>**

# Comentarios Personales

**<< MÁXIMO 2 PÁGINAS >>**

## PROBLEMAS ENCONTRADOS

<< Descripción somera de los problemas encontrados mientras se desarrollaba la práctica y cómo se han solventado >>

* Memoria Dinámica: al principio de la práctica empleamos en algunas funciones memoria estática con arrays y solventamos los requisitos de utilizar memoria dinámica con la función “malloc”.
* Leer entrada estándar: al utilizar “scanf” para leer por entrada estándar nos dimos cuenta de que se podía producir desbordamiento de búfer, así que lo solucionamos utilizando la función “fgets”.

## CRÍTICAS CONSTRUCTIVAS

<< Lista de críticas constructivas sobre el alcance y enfoque de la práctica >>

Respecto al alcance de la práctica y el enfoque sinceramente que es una práctica útil para aprender a manejar memoria dinámica, la entrada y la salida estándar de C. Además, es imprescindible mencionar la utilidad de la gestión de las librerías de C, tanto crear una librería como compilarla o utilizarla en otros ficheros.

## PROPUESTA DE MEJORAS

<< Lista de posibles mejoras sobre la práctica, alcance, tiempo dedicado, apoyo… Se puede basar en la lista de críticas constructivas >>

## EVALUACIÓN DEL TIEMPO DEDICADO

<< Evaluación estimada del tiempo dedicado al desarrollo de la práctica, no sólo el tiempo de programación >>