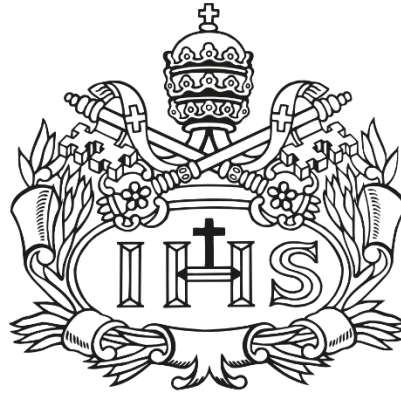


**Pontificia Universidad Javeriana**

**Facultad de Ingeniería**



Pontificia Universidad  
**JAVERIANA**  
Colombia

**Taller01 – Sistemas Operativos**

**Profesor: John Corredor, Ph.D.**

**Brayan Fajardo**

**Septiembre 2024**

## 1. Introducción

Este informe describe el desarrollo e implementación del programa solicitado en el Taller 01 de Sistemas Operativos, que consiste en la aplicación de los conceptos de procesos y comunicación entre procesos en lenguaje C. El programa, llamado taller\_procesos, recibe archivos que contienen arreglos de enteros, los carga en memoria dinámica, realiza operaciones aritméticas a través de múltiples procesos, y comunica los resultados entre ellos utilizando pipe(). En este documento se explican los objetivos, el desarrollo del código, la metodología empleada, y los resultados obtenidos.

## 2. Objetivos

- Aplicar los conceptos de procesos y comunicación entre procesos utilizando fork() y pipe() en un programa en lenguaje C.
- Leer y procesar datos almacenados en archivos de texto mediante arreglos dinámicos.
- Implementar un programa que divida las tareas entre varios procesos y permita la comunicación eficiente entre ellos.
- Evaluar el uso de memoria dinámica y la correcta gestión de recursos en un entorno multitarea.

## 3. Descripción del problema

El programa solicitado debe:

1. Recibir 4 argumentos desde la línea de comandos:
  - N1: Cantidad de elementos en el primer archivo.
  - archivo00: Archivo que contiene los primeros N1 números enteros.
  - N2: Cantidad de elementos en el segundo archivo.
  - archivo01: Archivo que contiene los segundos N2 números enteros.
2. Leer los archivos de texto proporcionados y cargar los enteros en arreglos dinámicos.
3. Crear 4 procesos: un proceso padre y tres procesos hijo.
  - El nieto calcula la suma de los elementos del primer archivo.
  - El segundo hijo calcula la suma de los elementos del segundo archivo.
  - El primer hijo calcula la suma total de ambos arreglos.

- El padre recibe los resultados de los hijos a través de pipe() y los imprime.
- 4. Manejar la memoria dinámica de manera adecuada, liberándola una vez que se haya utilizado.

## 4. Desarrollo del programa

### 4.1. Programa principal

El programa principal, taller01\_main.c, recibe cuatro argumentos y se encarga de leer los archivos y cargar sus valores en arreglos dinámicos. A continuación, se presenta la estructura básica del código

### 4.2. Creación de archivos de texto

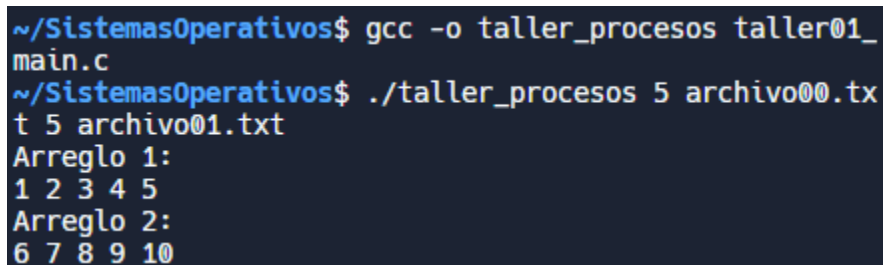
Se crearon dos archivos de texto, archivo00.txt y archivo01.txt, los cuales contienen los arreglos de enteros.

### 4.3. Ejecución del programa

El programa fue compilado y ejecutado correctamente utilizando el siguiente comando:

```
gcc -o taller_procesos taller01_main.c  
./taller_procesos 5 archivo00.txt 5 archivo01.txt
```

La salida esperada fue:



```
~/SistemasOperativos$ gcc -o taller_procesos taller01_main.c  
~/SistemasOperativos$ ./taller_procesos 5 archivo00.txt 5 archivo01.txt  
Arreglo 1:  
1 2 3 4 5  
Arreglo 2:  
6 7 8 9 10
```

Esto confirma que los archivos fueron leídos correctamente y los datos se cargaron en los arreglos dinámicos.

## 5. Gestión de memoria

Una parte importante del proyecto fue la gestión de memoria dinámica para almacenar los arreglos de enteros. Se utilizó malloc() para asignar memoria a los arreglos y free() para liberar la memoria después de su uso, evitando fugas de memoria.

## 6. Creación de procesos y comunicación

Para la siguiente fase del proyecto, se implementarán los procesos necesarios utilizando `fork()` y la comunicación entre ellos mediante `pipe()`. Los cuatro procesos tendrán las siguientes responsabilidades:

- El nieto sumará los elementos del primer archivo.
- El segundo hijo sumará los elementos del segundo archivo.
- El primer hijo sumará el total de ambos arreglos.
- El padre recibirá los resultados de los tres hijos a través de `pipe()` y los imprimirá.

Este paso permitirá que el programa realice múltiples tareas simultáneamente, demostrando el manejo de procesos en C.

## 7. Conclusiones

En este informe se detalló el desarrollo de un programa en C que recibe argumentos desde la línea de comandos, lee archivos de texto y almacena sus contenidos en arreglos dinámicos. El programa cumple con los requisitos iniciales del taller, y el siguiente paso será la implementación de los procesos hijos y la comunicación entre ellos. El desarrollo de este taller fue una experiencia enriquecedora, ya que al principio los retos de manejar la lectura de archivos, la memoria dinámica y los procesos parecían complicados. Sin embargo, a medida que avancé en el código, fue gratificante ver cómo las piezas encajaban correctamente. La ejecución exitosa del programa, mostrando los arreglos como se esperaba, no solo reforzó mis conocimientos en C y en la gestión de memoria dinámica, sino que también me dio mayor confianza en el manejo de procesos y la comunicación entre ellos, conceptos fundamentales en los sistemas operativos.

## 8. Referencias

1. Hernández, J. (2017, noviembre 10). ¿Cómo funciona la función `fork`? StackOverflow en español. <https://es.stackoverflow.com/questions/179414/como-funciona-la-funci%C3%B3n-fork>
2. Gutiérrez, P. (2019, noviembre 20). Comunicación entre pipes en C. StackOverflow en español. <https://es.stackoverflow.com/questions/310188/comunicaci%C3%B3n-entre-pipes-en-c>