JORGE LÓPEZ GÓMEZ OUSSAMA BOLBAROUD

UCLM

PROGRAMACIÓN CONCURRENTE Y TIEMPO REAL PRÁCTICA 1.2

**ÍNDICE**

[INTRODUCCIÓN 2](#_Toc161421092)

[MANAGER 3](#_Toc161421093)

[PROCESADOR 4](#_Toc161421094)

[CONTADOR 5](#_Toc161421095)

[LISTA 6](#_Toc161421096)

INTRODUCCIÓN

Este documento describe las especificaciones para desarrollar un sistema que consta de tres programas que simulen un escenario de procesamiento concurrente. Este sistema se compone de tres tipos de procesos: el Manager, los Procesadores y los Contadores.

El Manager es responsable de supervisar la creación, ejecución y finalización de los procesos Procesador y Contador. Se ingresa la ubicación de dos archivos: uno contiene texto y el otro contiene patrones. Utilizando el archivo de texto, se generan operaciones de Contador, a la vez que se realizan operaciones de Procesador con el archivo de patrones.

Los Procesadores tienen la responsabilidad de detectar patrones particulares en el texto que se encuentra en el archivo dado. Se encarga de verificar si un patrón dado corresponde con alguna palabra del archivo de texto.

Además, los Contadores deben contar cuántas palabras hay en cada línea del documento de texto.

El Manager asegura que el programa termine cuando todos los Procesadores y Contadores hayan finalizado o si el usuario presiona Ctrl + C.

Mediante una demostración práctica, se muestra cómo el Manager gestiona la creación, ejecución y finalización de los procesos Procesador y Contador, exhibiendo el resultado de su desempeño.

El sistema propuesto ofrece una herramienta para procesar texto de manera eficiente utilizando varios procesos concurrentes, mostrando un enfoque práctico para manejar procesos en entornos de programación concurrente y tiempo real.

MANAGER

El archivo “**manager.c”** es el núcleo fundamental de un programa que simula un sistema de procesos concurrentes. Dentro de este archivo reside la esencia del proceso "Manager", encargado de coordinar y gestionar el comportamiento de los demás procesos en el sistema.

A continuación, vamos a hablar sobre algunas de las funciones más importantes del archivo “**manager.c**”;

* **procesar\_argumentos()**: Esta función es responsable de analizar los argumentos que se pasan al programa. Dichos argumentos incluyen el nombre del archivo, el patrón a buscar y el número de líneas en el archivo.
* **instalar\_manejador\_senhal()**: En esta función se instala un manejador de señal. Este manejador se encarga de gestionar las señales enviadas al proceso, como la señal de terminación.
* **procesar\_patrones()**: Su tarea consiste en examinar los patrones contenidos en el archivo. Lee cada patrón del archivo correspondiente y, por cada uno encontrado, crea un nodo en una lista para su posterior uso.
* **iniciar\_tabla\_procesos()**: Aquí se establece el inicio de la tabla de procesos. Esta estructura de datos mantiene un registro de todos los procesos creados por el manager durante la ejecución del programa.
* **crear\_procesos()**: Esta función desempeña el papel crucial de generar los procesos necesarios. Por cada línea en el archivo de texto, crea un proceso contador. Asimismo, por cada nodo en la lista de patrones, genera un proceso procesador.
* **esperar\_procesos()**: Su función principal es aguardar a que todos los procesos hayan finalizado su ejecución. Cuando todos los procesos completen sus tareas, el manager puede liberar los recursos utilizados.
* **liberar\_recursos()**: Finalmente, esta función se encarga de liberar los recursos que han sido reservados por el manager durante la ejecución del programa. Esto incluye la liberación de memoria asignada para la lista de patrones y la tabla de procesos.

En la función “**main**”, se ejecuta una secuencia lógica donde se procesan los argumentos, se instala el manejador de señales, se procesan los patrones del archivo, se inicia la tabla de procesos, se crean los procesos y se espera a que todos terminen. Por último, se procede a la liberación de los recursos.

Finalmente, podemos llegar a la conclusión que el archivo “**manager.c**” es de suma importancia para el funcionamiento integral del programa, ya que centraliza la creación, gestión y finalización de los procesos. Además, cumple un papel vital en el manejo de las señales y la liberación adecuada de recursos una vez que todos los procesos han completado sus tareas.

PROCESADOR

CONTADOR

LISTA