Práctica 1. 2021/1 Laboratorio de sistemas operativos

Realizada por
Alejandro Pérez
Juan Felipe Santa

Sistemas operativos

Universidad de Antioquia
Medellín, Colombia

Estrategias adoptada para la elaboración de la práctica

Para desarrollar esta práctica inicialmente se presentaron diferentes dificultades por el poco conocimiento que se tenía sobre el lenguaje de C, para lo cual se empezó a seguir con detenimiento las diferentes tutorías brindadas por el profesor del laboratorio. Luego de tener una versión del código basado en las diferentes tutorías donde se explicaba y se realizaba en C la lectura de archivos, recorrido y extracción de datos de cada una de las líneas. Pero cuando se empezó a desarrollar algunos métodos se tuvo confusión con respecto a los apuntadores, para lo cual se retoma las explicaciones y se realizan diferentes búsquedas en internet y con ayuda del material del curso se logra entender el funcionamiento de un apuntador (*), apuntador doble (**), etc (***n).

Una de las estrategias empleadas fue inicialmente desarrollar la práctica en Python y así evitar posibles errores de lógica durante la implementación en C, pues bien se sabía que no se contaba con mucha experiencia en C y sería más difícil encontrar un posible error de lógica allí, por lo cual garantizamos que el algoritmo funcionará en Python y así luego realizarlo en C.

Otra de las estrategias usadas para realizar el laboratorio fue realizar pseudocódigo de pequeños algoritmos, nunca estará de más escribir a mano para comprender algo, en este caso, para comprender el modelo matemático que se tenía planteado en la práctica, el cual inicialmente era algo confuso, e igualmente con la construcción de la matriz y con estas estrategias mencionadas anteriormente se logró llevar a cabo la práctica.

Principales retos encontrados durante el desarrollo de la práctica

Uno de los principales retos fue el manejo de los apuntadores, por ejemplo, en la construcción de la matriz se debía asignar puntero doble (**), y reservar el espacio en memoria antes de la creación, de lo contrario llenaba la matriz con valores de memoria que se tenían guardados, pero fue más sencillo de lo pensado, y nos ayudamos de la función ***malloc*** la cual nos permite hacer asignación de memoria dinámica y así evitar gastos innecesarios de memoria.

Se tuvo dificultad también con los finales de línea, pues se había realizado un copy and paste del archivo en Windows y estaba tomando otro carácter de más,

bastó con tomar otro archivo file.txt creado en Linux y así funcionó correctamente la función que identificaba los fin de línea.

Con respecto a las permutaciones se hacen mediante la ayuda del algoritmo **heap** el cual ordena un vector de n cantidad de elementos construyendo un heap con los n elementos y extrayendolos uno a uno del heap a continuación. En el momento que se realizaba una permutación se pasaba el vector solución al modelo matemático para realizar respectiva operación y calcular el máximo de ingredientes.

Básicamente estos eran los retos más complicados que se tuvieron durante el desarrollo de la práctica inicialmente, luego que se logró entender cómo funcionaban los apuntadores con respecto a una matriz y a un vector se fue haciendo más trabajable la práctica y pensar en una solución para algo en C se iba haciendo cada vez más entendible sin necesidad de estar buscando soluciones en otros lugares.

Análisis del lenguaje C con respecto a los lenguajes de alto nivel

Con respecto a otros lenguajes puede parecer un poco más complejo, por ejemplo comparado con Python, mientras en Python ignorábamos la gestión de la memoria aquí debíamos tener demasiado en cuenta esto, pues en Python se utiliza un recolector de basura automático para la gestión de la memoria, y en C nosotros debíamos realizar la respectiva gestión de la memoria. Otro punto es con respecto a las declaración de las variables, en Python no es necesario declarar el tipo, pero en C es obligatorio declarar el tipo de variable no importa si se inicializa o no. Además de la depuración del código, es un poco más tedioso realizar las pruebas y la depuración en C, esto resulta ser mucho más fácil en otros lenguajes.

Ahora bien, con respecto a otros aspectos como la velocidad, comparando ciertos procesos que implementamos en Python y luego en C, se puede evidenciar que la velocidad de C es mayor, Python es un poco más lento, por lo cual C tiene características bastante interesantes y más cuando se habla de implementación de aplicaciones relacionadas con el hardware, C es lo mejor en su respectivo campo.