SUMA DE ENTEROS EN C

Objetivo

Desarrollar y profundizar las competencias en el lenguaje de programación C, enfocándose en técnicas de manejo de archivos y gestión de memoria dinámica. Los estudiantes aprenderán a leer y procesar datos de archivos con formato específico, implementarán métodos eficientes para la manipulación de grandes volúmenes de datos en memoria, y emplearán herramientas de medición de tiempo para analizar la eficiencia de sus programas. Este enfoque no solo busca reforzar la comprensión de conceptos fundamentales de programación, sino también incentivar el desarrollo de aplicaciones eficientes y optimizadas en C.

Descripción de la actividad

Implementar un programa en C que procese un archivo de texto plano conteniendo una secuencia de números enteros. El archivo está estructurado de tal manera que el primer número entero, N, especifica la cantidad de números enteros subsiguientes, cada uno de los cuales se presenta en una línea separada. Estos enteros, denotados como k_i, cumplen con la condición $k_i \leq 10{,}000$. El programa debe calcular la suma total de estos enteros y medir el tiempo de ejecución de la operación de lectura y suma.

Descripción	Ejemplo
#N representando la cantidad de enteros	5
Entero1	1548
Entero2	2354
Entero3	1574
.	56
EnteroN	36

Implementación Propuesta

Paso 1: Inicialización y Lectura Preliminar

- *Apertura de Archivo*: Utilizar la función ***fopen*** con el nombre del archivo especificado como argumento en la línea de comandos y el modo de apertura "r" para leer. En caso de fallo al abrir el archivo, fopen retornará NULL.

```
FILE *file;
file = fopen(argv[1], "r");
if (file == NULL) {
    printf("Error al abrir el archivo\n");
    exit(1);
}
```

- *Lectura del Contador de Enteros*: Leer el primer entero N del archivo, que indica cuántos enteros subsiguientes necesitan ser procesados. Esta lectura inicial es crucial para determinar el tamaño del bloque de memoria que se reservará dinámicamente para almacenar estos enteros en la memoria principal.

```
int size;
int *array = NULL;
fscanf(file, "%d", &size);
```

- *Reserva de Memoria*: Posterior a la lectura de N, se debe realizar una reserva de memoria dinámica adecuada, utilizando ***malloc***, para un arreglo de N enteros. Esta operación es fundamental para asegurar que hay suficiente espacio en memoria para almacenar los enteros que serán leídos y procesados.

```
array = (int *)malloc(size * sizeof(int));
```

- *Lectura del contenido:* una vez se ha garantizado las necesidades de memoria se puede almacenar el contenido del archivo en memoria.

```
long int sum = 0;
for (int i = 0; i < size; i++) {
    fscanf(file, "%d", &array[i]);
    sum += array[i];
}
```

- Medición de los tiempos computacionales: Finalmente, se mide el tiempo computacional empleado para ejecutar estas instrucciones mediante la función *clock_gettime* de la librería de cabecera *<time.h>*. Se mide el tiempo utilizado para realizar la lectura del archivo, que incluye la apertura, la lectura del primer valor, la reserva de memoria y la carga en memoria principal de los N enteros. También se mide el tiempo utilizado para realizar la suma, que incluye el ciclo que recorre los valores y acumula su suma. Para medir el tiempo de ejecución, se utiliza la función *clock_gettime*, que toma como argumento un reloj y un puntero a una estructura *timespec*, que es llenada con el tiempo actual del reloj. La estructura *timespec* contiene dos campos: *tv_sec* para almacenar el tiempo en segundos y *tv_nsec* para almacenar el tiempo en nanosegundos.

```
struct timespec start, end;
double elapsed;
clock_gettime(CLOCK_REALTIME, &start);

// Operación de lectura y estimación de la suma
// ...

clock_gettime(CLOCK_REALTIME, &end);
elapsed = (end.tv_sec - start.tv_sec) + ((end.tv_nsec - start.tv_nsec)
```

- *Cierre del archivo y liberación de memoria:* Una vez que se ha realizado la operación de lectura y estimación de la suma, es necesario cerrar el archivo y liberar la memoria reservada dinámicamente.

```
fclose(file);
free(array);
```

Ajuste los fragmentos de código anterior para recibir el nombre del archivo por línea de comandos. La salida debe especificar el tiempo usando para la lectura del archivo, esto incluye la apertura, la lectura del primer valor, la reserva de memoria y la carga en memoria principal de los N enteros. Se debe reportar el tiempo utilizado para realizar la suma, esto incluye el ciclo que recorre los valores y acumula su suma.

Como trabajo independiente se sugiere complementar el programa para responder a los siguientes retos:

- Estimar la varianza, la desviación estándar y la mediana de los datos
- ¿Cuál es el valor que más se repite en los enteros leídos?

Como datos se prueba se suministran tres archivos:

Nombre del archivo	Cantidad de enteros	suma	valores más frecuentes (valor, repeticiones)
test1.in	100	501272	(717, 2); (4230,2)
test2.in	1.000.000	4996536236	(8822, 137); (4033, 136)
test3.in	5.000.000	24998583742	(9021, 602); (3906, 589)

Ejemplo de la salida esperada (*salida para el archivo de prueba test2.in*):