Bogotá, 18 de agosto de 2024

Juan Diego Muñoz Angulo

Sistemas Operativos

Documentación: Gestión de Memoria y Multiplicación de Matrices

Este código multiplica dos matrices con valores y con un tamaño proporcional al que desee proveer el usuario.

1. **imprimir**: Imprime los elementos de una matriz.
2. **insertar**: Rellena la matriz con un valor x, que nos da el usuario.
3. **multiplicar**: Realiza la multiplicación de dos matrices cuadradas de tamaño n x n. Se realiza una multiplicación matricial estándar y el resultado se guarda en una nueva matriz C, cuya memoria se asigna de forma dinámica, y luego se libera al salir del programa.

El programa está diseñado para ser ejecutado desde la línea de comandos Shell o de Linux, con dos argumentos que representan el número de elementos en la matriz, y los valores que deben llevar dentro, ejemplo, ***“.\'matrizFuncionalFinal.exe' 9 1 2”.***

Si no se proporcionan, el programa señala el error que sale al inicio de la función main, que significa que faltan o sobran argumentos. Puede ser compilado perfectamente con GCC.

```c
/*****************************************
*                                        *
* Autor: JD                              *
* Materia: Sistemas Operativos           *
* Fecha: 18-08-24                        *
* Tema: Gestion de Memoria               *
*  Entrega Tarea Gestion de Memoria      *
*                                        *
*****************************************/

#include <stdio.h>  //Bibliotecas
#include <stdlib.h>
#include <math.h>

// Función para imprimir una matriz
void imprimir(int size, int *matriz) {
    int n = (int)sqrt(size);
    for(int i = 0; i < size; i++) {
        if(i % n == 0) {
            printf("\n");
        }
        printf("%d ", matriz[i]); //Imprime cada elemento
    }
    printf("\n");
}

// Función para llenar una matriz
void insertar(int size, int dato, int *matriz) {
    for(int i = 0; i < size; i++) { //Recorre toda la matriz
        matriz[i] = dato; //Asigan valores
    }
}

// Función para multiplicar dos matrices
int* multiplicar(int size, int *mA, int *mB) {
    int n = (int)sqrt(size); //Calcula la dimensión de la matriz cuadrada.
    int *mC = (int*)malloc(size * sizeof(int)); //Reserva memoria para la matriz resultado
(mC)
    for(int i = 0; i < n; i++) { //Itera sobre filas de mA
        for(int j = 0; j < n; j++) { //Itera sobre columnas de mB
            mC[i*n + j] = 0; //Inicializa en 0 la matriz mC para evitar errores
            for(int k = 0; k < n; k++) {
                mC[i*n + j] += mA[i*n + k] * mB[k*n + j]; //Suma el productocorrespondientes.
            }
        }
    }
    return mC; //Devuelve el resultado en una matriz nueva
}


int main(int argc, char *argv[]) {
    if (argc != 4){ // Verifica si se pasaron exactamente 3 argumentos.

        printf("Error de argumentos.\n");
        printf("Debe tener 3 argumentos, el núm de espacios en la Matriz.\n");
        printf("Más los valores que llenar en la matriz 1 y 2.\n");
        printf("Ej: matrizFuncionalFinal 9 1 2 \n");
        return 0;
    }

    int size = atoi(argv[1]);
    int valorA = atoi(argv[2]);
    int valorB = atoi(argv[3]);
    int mA[size];
    int mB[size];

    insertar(size, valorA, mA);
    insertar(size, valorB, mB);

    printf("Matriz A:");
    imprimir(size, mA);
    printf("Matriz B:");
    imprimir(size, mB);

    int *mC = multiplicar(size, mA, mB);
    printf("Matriz C Resultado:");
    imprimir(size, mC);

    free(mC); //  Libera la memoria de a mC

    printf("\nEso es todo amigos.\n");
    return 0;
}

```