METODO PARA SUMA DE MATRICES

public class MatrizEjemplo {

public static void main(String[] args) {

// Definir una matriz de 3x3

int[][] matriz = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};

// Imprimir la matriz

System.out.println("Matriz:");

imprimirMatriz(matriz);

// Calcular la suma de todos los elementos de la matriz

int suma = calcularSumaMatriz(matriz);

System.out.println("La suma de todos los elementos de la matriz es: " + suma);

// Calcular la suma de cada fila y cada columna

calcularSumaFilas(matriz);

calcularSumaColumnas(matriz);

}

// Método para imprimir la matriz

public static void imprimirMatriz(int[][] matriz) {

for (int i = 0; i < matriz.length; i++) {

for (int j = 0; j < matriz[i].length; j++) {

System.out.print(matriz[i][j] + "\t");

}

System.out.println();

}

}

// Método para calcular la suma de todos los elementos de la matriz

public static int calcularSumaMatriz(int[][] matriz) {

int suma = 0;

for (int i = 0; i < matriz.length; i++) {

for (int j = 0; j < matriz[i].length; j++) {

suma += matriz[i][j];

}

}

return suma;

}

// Método para calcular la suma de cada fila de la matriz

public static void calcularSumaFilas(int[][] matriz) {

System.out.println("Suma de cada fila:");

for (int i = 0; i < matriz.length; i++) {

int sumaFila = 0;

for (int j = 0; j < matriz[i].length; j++) {

sumaFila += matriz[i][j];

}

System.out.println("Fila " + (i + 1) + ": " + sumaFila);

}

}

// Método para calcular la suma de cada columna de la matriz

public static void calcularSumaColumnas(int[][] matriz) {

System.out.println("Suma de cada columna:");

for (int j = 0; j < matriz[0].length; j++) {

int sumaColumna = 0;

for (int i = 0; i < matriz.length; i++) {

sumaColumna += matriz[i][j];

}

System.out.println("Columna " + (j + 1) + ": " + sumaColumna);

}

}

}

MATRIZ TRANSPUESTA

public class TransposeMatrix {

public static void main(String[] args) {

int[][] matrizOriginal = {{1, 2, 3}, {4, 5, 6}, {7, 8, 9}};

int filas = matrizOriginal.length;

int columnas = matrizOriginal[0].length;

// Crear una nueva matriz para almacenar la transpuesta

int[][] matrizTranspuesta = new int[columnas][filas];

// Calcular la transpuesta de la matriz original

for (int i = 0; i < filas; i++) {

for (int j = 0; j < columnas; j++) {

matrizTranspuesta[j][i] = matrizOriginal[i][j];

}

}

// Imprimir la matriz transpuesta

System.out.println("Matriz Transpuesta:");

for (int i = 0; i < columnas; i++) {

for (int j = 0; j < filas; j++) {

System.out.print(matrizTranspuesta[i][j] + " ");

}

System.out.println();

}

}

}

MULTIPLICACION DE MATRICES

public class MultiplicacionMatrices {

public static void main(String[] args) {

// Definir las matrices

int[][] matriz1 = {{1, 2, 3}, {4, 5, 6}};

int[][] matriz2 = {{7, 8}, {9, 10}, {11, 12}};

// Obtener las dimensiones de las matrices

int filasMatriz1 = matriz1.length;

int columnasMatriz1 = matriz1[0].length;

int filasMatriz2 = matriz2.length;

int columnasMatriz2 = matriz2[0].length;

// Verificar si las matrices se pueden multiplicar

if (columnasMatriz1 != filasMatriz2) {

System.out.println("No se pueden multiplicar las matrices. El número de columnas de la primera matriz debe ser igual al número de filas de la segunda matriz.");

return;

}

// Crear una matriz para almacenar el resultado de la multiplicación

int[][] resultado = new int[filasMatriz1][columnasMatriz2];

// Calcular la multiplicación de matrices

for (int i = 0; i < filasMatriz1; i++) {

for (int j = 0; j < columnasMatriz2; j++) {

for (int k = 0; k < columnasMatriz1; k++) {

resultado[i][j] += matriz1[i][k] \* matriz2[k][j];

}

}

}

// Imprimir la matriz resultado

System.out.println("Resultado de la multiplicación:");

for (int i = 0; i < filasMatriz1; i++) {

for (int j = 0; j < columnasMatriz2; j++) {

System.out.print(resultado[i][j] + " ");

}

System.out.println();

}

}

}

VEHICULOS

import java.util.Arrays;

import java.util.Scanner;

public class GestionVentaVehiculos {

private static final int MAX\_VEHICULOS = 100;

private String[] vehiculos = new String[MAX\_VEHICULOS];

private int numVehiculos = 0;

private Scanner scanner = new Scanner(System.in);

public static void main(String[] args) {

GestionVentaVehiculos programa = new GestionVentaVehiculos();

programa.iniciar();

}

public void iniciar() {

int opcion;

do {

mostrarMenu();

opcion = leerEntero("Seleccione una opción: ");

switch (opcion) {

case 1:

agregarVehiculo();

break;

case 2:

buscarVehiculo();

break;

case 3:

eliminarVehiculo();

break;

case 4:

ordenarYListarVehiculos();

break;

case 5:

editarVehiculo();

break;

case 6:

System.out.println("Saliendo del programa...");

break;

default:

System.out.println("Opción no válida. Por favor, seleccione una opción válida.");

}

} while (opcion != 6);

}

private void mostrarMenu() {

System.out.println("GESTIÓN DE VENTA DE VEHÍCULOS - MENÚ DE OPCIONES");

System.out.println("1. Agregar vehículo");

System.out.println("2. Buscar vehículo");

System.out.println("3. Eliminar vehículo");

System.out.println("4. Listar vehículos");

System.out.println("5. Editar vehículo por índice");

System.out.println("6. Salir");

}

private int leerEntero(String mensaje) {

System.out.print(mensaje);

while (!scanner.hasNextInt()) {

System.out.println("Por favor, ingrese un número entero.");

System.out.print(mensaje);

scanner.next();

}

return scanner.nextInt();

}

private void agregarVehiculo() {

if (numVehiculos < MAX\_VEHICULOS) {

System.out.print("Ingrese el nombre del vehículo: ");

String nombre = scanner.next();

vehiculos[numVehiculos] = nombre;

numVehiculos++;

System.out.println("Vehículo agregado correctamente.");

} else {

System.out.println("No es posible agregar más vehículos, capacidad máxima alcanzada.");

}

}

private void buscarVehiculo() {

System.out.print("Ingrese el nombre del vehículo a buscar: ");

String nombre = scanner.next();

boolean encontrado = false;

for (int i = 0; i < numVehiculos; i++) {

if (vehiculos[i].equalsIgnoreCase(nombre)) {

System.out.println("El vehículo \"" + nombre + "\" se encuentra en la posición " + i);

encontrado = true;

break;

}

}

if (!encontrado) {

System.out.println("Vehículo no encontrado.");

}

}

private void eliminarVehiculo() {

System.out.print("Ingrese el nombre del vehículo a eliminar: ");

String nombre = scanner.next();

boolean eliminado = false;

for (int i = 0; i < numVehiculos; i++) {

if (vehiculos[i].equalsIgnoreCase(nombre)) {

for (int j = i; j < numVehiculos - 1; j++) {

vehiculos[j] = vehiculos[j + 1];

}

numVehiculos--;

eliminado = true;

System.out.println("Vehículo \"" + nombre + "\" eliminado correctamente.");

break;

}

}

if (!eliminado) {

System.out.println("Vehículo no encontrado.");

}

}

private void ordenarYListarVehiculos() {

if (numVehiculos == 0) {

System.out.println("No hay vehículos registrados.");

} else {

Arrays.sort(vehiculos, 0, numVehiculos, String.CASE\_INSENSITIVE\_ORDER);

System.out.println("Listado de vehículos:");

for (int i = 0; i < numVehiculos; i++) {

System.out.println((i + 1) + ". " + vehiculos[i]);

}

}

}

private void editarVehiculo() {

if (numVehiculos == 0) {

System.out.println("No hay vehículos registrados para editar.");

} else {

System.out.print("Ingrese el índice del vehículo a editar: ");

int indice = leerEntero("");

if (indice >= 1 && indice <= numVehiculos) {

System.out.print("Ingrese el nuevo nombre del vehículo: ");

String nuevoNombre = scanner.next();

vehiculos[indice - 1] = nuevoNombre;

System.out.println("Vehículo editado correctamente.");

} else {

System.out.println("Índice fuera de rango.");

}

}

}

}