**Benemérita Universidad Autónoma de Puebla**



**EJERCICIO 6:**

**“Matriz”**

**Fac. de Cs. Computación**

**Asignatura:**

**Programación II**

**Profesora: Carmen Cerón Garnica**

**Equipo 4:**

**Emmanuel Flores Navarro 201929142**

**Erick Brian Ramírez Morales 201955520**

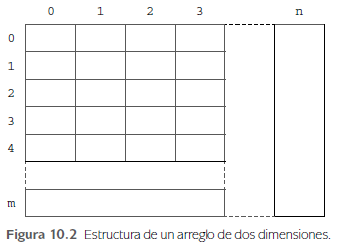
**07 / Mayo / 2021**

**INTRODUCCIÓN**

**Arreglos multidimensionales**

Los arreglos multidimensionales tienen más de una dimensión y, en consecuencia, más de un índice, los más utilizados son los de dos dimensiones, conocidos como tablas o matrices; sin embargo, es posible crear arreglos de tantas dimensiones como las aplicaciones requieran: tres, cuatro o más.

Un arreglo de dos dimensiones equivale a una tabla con múltiples filas y múltiples columnas (ver figura 10.2).



Obsérvese el arreglo bidimensional de la figura 10.2, si las filas se etiquetan de 0 a m y las columnas de 0 a n, el número de elementos que tendrá el arreglo será el resultado del producto (m+1)\*(n+1); un elemento es localizado mediante las coordenadas representadas por su número de fila y de columna (a, b); la sintaxis para la declaración de un arreglo de dos dimensiones es:

*<tipo de datoElemento> <nombre arreglo > [] [];*

o bien

*<tipo de datoElemento> [] []<nombre arreglo >;*

Ejemplos de declaración de matrices:

*char pantalla[][];*

*int puestos[][];*

*double [][]matriz;*

Estas declaraciones no reservan memoria para los elementos de la matriz, en realidad son referencias; para reservar memoria y especificar el número de filas y de columnas se utiliza el operador *new*. A partir de las declaraciones anteriores se crean las respectivas matrices:

*pantalla = new char[80][24]; // matriz con 80 filas y 24 columnas*

*puestos = new int[10][5]; // matriz de 10 filas por 5 columnas*

*final int N = 4;*

*matriz = new double[N][N]; // matriz cuadrada de N\*N elementos*

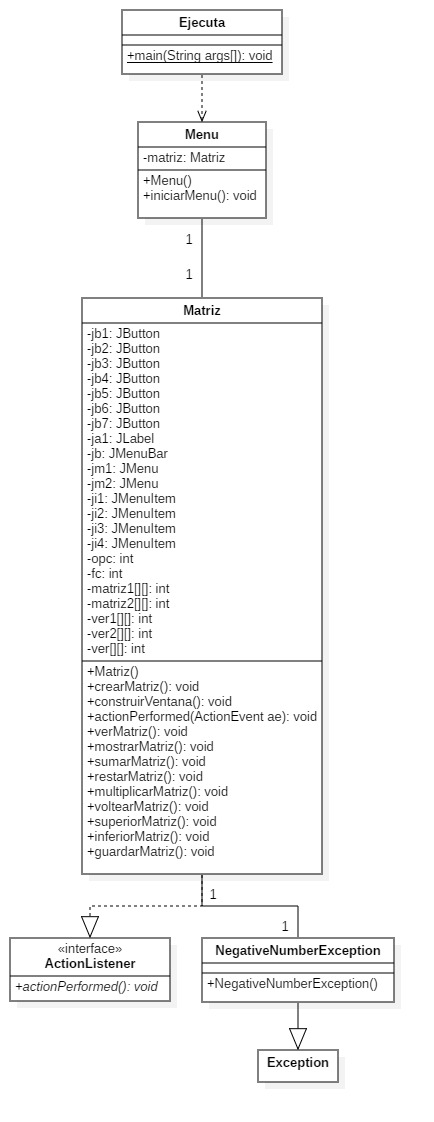
El operador new se puede aplicar a la vez que se hace la declaración; la sintaxis para definir una matriz es:

*<tipo de datoElemento> <nombre arreglo >[][]= new<tipo de datoElemento> [<NúmeroDeFilas<][<NúmeroDeColumnas>];*

Un arreglo bidimensional en realidad es un arreglo de arreglos; es decir, un arreglo unidimensional donde cada elemento no es un valor entero, de coma flotante o carácter, sino otro arreglo.

Los elementos se almacenan en memoria de modo que el subíndice más próximo al nombre del arreglo es la fila y el otro subíndice, la columna; la tabla 10.1 representa a todos los elementos y sus posiciones relativas en memoria del arreglo *int [][]tabla = new int[4][2].*

**DIAGRAMA UML**



**CÓDIGO DEL PROGRAMA**

**Ejecuta.java**

// Fecha: 07 / Mayo / 2021

// Objetivo: Crear una matriz y realizar algunos ejercicios.

// Elaboro Equipo 4:

// 1) Flores Navarro Emmanuel

// 2) Ramírez Morales Erick Brian

//Inicio de la clase

public class Ejecuta {

public static void main(String args[]) {

//Variable de instancia

Menu menu = new Menu();

//Acceder a un método

menu.iniciarMenu();

}

}

//Fin de la clase

**Matriz.java**

// Fecha: 07 / Mayo / 2021

// Objetivo: Crear una matriz y realizar algunos ejercicios.

// Elaboro Equipo 4:

// 1) Flores Navarro Emmanuel

// 2) Ramírez Morales Erick Brian

//Inicio de la clase

import javax.swing.JOptionPane;

import java.awt.event.\*;

import javax.swing.\*;

import java.awt.\*;

import java.io.\*;

public class Matriz implements ActionListener {

//Variables de instancia

private JButton jb1;

private JButton jb2;

private JButton jb3;

private JButton jb4;

private JButton jb5;

private JButton jb6;

private JButton jb7;

private JLabel ja1;

private JMenuBar jb;

private JMenu jm1;

private JMenu jm2;

private JMenuItem ji1;

private JMenuItem ji2;

private JMenuItem ji3;

private JMenuItem ji4;

//Atributos

private int opc = 0;

private int fc = 0;

//Matrices

private int matriz1[][] = new int[100][100];

private int matriz2[][] = new int[100][100];

private int ver1[][] = new int[100][100];

private int ver2[][] = new int[100][100];

private int ver[][] = new int[100][100];

//Método constructor

public Matriz() {}

//Metódo definido por el programador

public void crearMatriz() {

//Atributo

String panel;

int salir;

//Ciclo

do {

//Ingresar un dato

panel = JOptionPane.showInputDialog(null,"Ingrese el numero de filas y columnas de las matrices:","MATRIZ",JOptionPane.QUESTION\_MESSAGE);

//Iniciar excepcion

try {

//Asignar valores

fc = Integer.parseInt(panel);

salir = 1;

if(fc <= 0)

//Lanzar error

throw new NegativeNumberException();

} catch(NumberFormatException nfe) {

//Capturar y mostrar error

JOptionPane.showMessageDialog(null,"ERROR..."+nfe.getMessage()

+"\nNo leiste dato numerico.","ERROR",JOptionPane.WARNING\_MESSAGE);

salir = 0;

} catch(NegativeNumberException nne) {

//Capturar y mostrar error

JOptionPane.showMessageDialog(null,"ERROR..."+nne.getMessage()

+"\nEl numero de filas y columnas debe ser mayor que cero.","ERROR",JOptionPane.WARNING\_MESSAGE);

salir = 0;

}

} while(salir != 1);

//Ciclo

for(int i=0; i<fc; i++)

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

//Ciclo

do {

//Iniciar excepcion

try {

//Ingresar un dato

panel = JOptionPane.showInputDialog(null,"Ingrese un valor para Elemento["+(i+1)+"]["+(j+1)+"]:","MATRIZ 1",JOptionPane.QUESTION\_MESSAGE);

matriz1[i][j] = Integer.parseInt(panel);

ver1[i][j] = Integer.parseInt(panel);

salir = 1;

} catch(NumberFormatException nfe) {

//Capturar y mostrar error

JOptionPane.showMessageDialog(null,"ERROR..."+nfe.getMessage()

+"\nNo leiste dato numerico.","ERROR",JOptionPane.WARNING\_MESSAGE);

salir = 0;

}

} while(salir != 1);

}

//Ciclo

for(int i=0; i<fc; i++)

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

//Ciclo

do {

//Iniciar excepcion

try {

//Ingresar un dato

panel = JOptionPane.showInputDialog(null,"Ingrese un valor para Elemento["+(i+1)+"]["+(j+1)+"]:","MATRIZ 2",JOptionPane.QUESTION\_MESSAGE);

matriz2[i][j] = Integer.parseInt(panel);

ver2[i][j] = Integer.parseInt(panel);

salir = 1;

} catch(NumberFormatException nfe) {

//Capturar y mostrar error

JOptionPane.showMessageDialog(null,"ERROR..."+nfe.getMessage()

+"\nNo leiste dato numerico.","ERROR",JOptionPane.WARNING\_MESSAGE);

salir = 0;

}

} while(salir != 1);

}

//Acceder a un método

construirVentana();

}

//Metódo definido por el programador

public void construirVentana() {

//Variables de instancia

JPanel jp = new JPanel();

JFrame jf = new JFrame();

//Barra de herramientas

jb = new JMenuBar();

jb.setBounds(0,0,100,20);

jf.add(jb);

//Herramienta 1

jm1 = new JMenu("Matriz");

jb.add(jm1);

//Opcion 1

ji1 = new JMenuItem("Ver");

jm1.add(ji1);

ji1.addActionListener(this);

jm1.addSeparator();

//Opcion 2

ji2 = new JMenuItem("Guardar");

jm1.add(ji2);

ji2.addActionListener(this);

//Herramienta 2

jm2 = new JMenu("Ayuda");

jb.add(jm2);

//Opcion 1

ji3 = new JMenuItem("Tutorial");

jm2.add(ji3);

ji3.addActionListener(this);

jm2.addSeparator();

//Opcion 2

ji4 = new JMenuItem("Acerca de...");

jm2.add(ji4);

ji4.addActionListener(this);

//Etiqueta 1

ja1 = new JLabel("Ingrese el numero de la opcion a ejecutar:");

ja1.setBounds(20,30,240,30);

jf.add(ja1);

//Boton 1

jb1 = new JButton("Matrices originales");

jb1.setBounds(20,70,240,30);

jf.add(jb1);

//Presionar

jb1.addActionListener(this);

//Boton 2

jb2 = new JButton("Sumar");

jb2.setBounds(20,110,240,30);

jf.add(jb2);

//Presionar

jb2.addActionListener(this);

//Boton 3

jb3 = new JButton("Restar");

jb3.setBounds(20,150,240,30);

jf.add(jb3);

//Presionar

jb3.addActionListener(this);

//Boton 4

jb4 = new JButton("Multiplicar");

jb4.setBounds(20,190,240,30);

jf.add(jb4);

//Presionar

jb4.addActionListener(this);

//Boton 5

jb5 = new JButton("Transpuesta");

jb5.setBounds(20,230,240,30);

jf.add(jb5);

//Presionar

jb5.addActionListener(this);

//Boton 6

jb6 = new JButton("Triangular superior");

jb6.setBounds(20,270,240,30);

jf.add(jb6);

//Presionar

jb6.addActionListener(this);

//Boton 7

jb7 = new JButton("Triangular inferior");

jb7.setBounds(20,310,240,30);

jf.add(jb7);

//Presionar

jb7.addActionListener(this);

//Ventana

jp.setBounds(0,0,300,400);

jp.setLayout(null);

jf.setLayout(null);

jf.setSize(300,400);

jf.add(jp);

jf.setVisible(true);

jf.setTitle("MATRIZ");

jf.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

}

//Metódo definido por el programador

@Override

public void actionPerformed(ActionEvent ae) {

//Validar boton presionado

if(ae.getSource() == jb1)

//Acceder a un método

mostrarMatriz();

if(ae.getSource() == jb2)

//Acceder a un método

sumarMatriz();

if(ae.getSource() == jb3)

//Acceder a un método

restarMatriz();

if(ae.getSource() == jb4)

//Acceder a un método

multiplicarMatriz();

if(ae.getSource() == jb5)

//Acceder a un método

voltearMatriz();

if(ae.getSource() == jb6)

//Acceder a un método

superiorMatriz();

if(ae.getSource() == jb7)

//Acceder a un método

inferiorMatriz();

if(ae.getSource() == ji1)

//Acceder a un método

verMatriz();

if(ae.getSource() == ji2)

//Acceder a un método

guardarMatriz();

if(ae.getSource() == ji3)

//Mostrar ventana

JOptionPane.showMessageDialog(null,"Pasos a seguir:"

+"\n1) Lea cada una de las opciones en pantalla."

+"\n2) Presione el botón a ejecutar."

+"\n\nNota:"

+"\n - La herramienta Matriz mostrara y/o guardara la ultima matriz modificada.");

if(ae.getSource() == ji4)

//Acceder a un método

JOptionPane.showMessageDialog(null,"Ejercicio 6 - Matriz:"

+"\n - Equipo 4:"

+"\n + Autor 1: Flores Navarro Emmanuel.......201929142"

+"\n + Autor 2: Ramírez Morales Erick Brian...201955520"

+"\n - Fecha: 07 / Mayo / 2021"

+"\n - Derechos Reservados");

}

//Metódo definido por el programador

public void verMatriz() {

//Atributo

String texto = "";

if(opc == 0) {

//Ciclo

texto += "MATRIZ 1:\n";

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

texto += "|";

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

texto += " "+ver1[i][j]+" |";

}

texto += "\n";

}

//Ciclo

texto += "\n\nMATRIZ 2:\n";

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

texto += "|";

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

texto += " "+ver2[i][j]+" |";

}

texto += "\n";

}

} else {

//Ciclo

texto += "MATRIZ:\n";

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

texto += "|";

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

texto += " "+ver[i][j]+" |";

}

texto += "\n";

}

}

//Mostrar ventana

JOptionPane.showMessageDialog(null,texto,"MATRIZ",JOptionPane.INFORMATION\_MESSAGE);

}

//Metódo definido por el programador

public void mostrarMatriz() {

//Atributos

String texto = "";

opc = 0;

//Ciclo

texto += "MATRIZ 1:\n";

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

texto += "|";

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

texto += " "+matriz1[i][j]+" |";

ver1[i][j] = matriz1[i][j];

}

texto += "\n";

}

//Ciclo

texto += "\n\nMATRIZ 2:\n";

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

texto += "|";

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

texto += " "+matriz2[i][j]+" |";

ver2[i][j] = matriz2[i][j];

}

texto += "\n";

}

//Mostrar ventana

JOptionPane.showMessageDialog(null,texto,"MATRIZ",JOptionPane.INFORMATION\_MESSAGE);

}

//Metódo definido por el programador

public void sumarMatriz() {

//Atributos

String texto = "";

opc = 1;

//Ciclo

texto += "SUMA:\n";

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

texto += "|";

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

//Sumar valores

ver[i][j] = matriz1[i][j] + matriz2[i][j];

texto += " "+ver[i][j]+" |";

}

texto += "\n";

}

//Mostrar ventana

JOptionPane.showMessageDialog(null,texto,"MATRIZ",JOptionPane.INFORMATION\_MESSAGE);

}

//Metódo definido por el programador

public void restarMatriz() {

//Atributos

String texto = "";

opc = 1;

//Ciclo

texto += "RESTA:\n";

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

texto += "|";

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

//Sumar valores

ver[i][j] = matriz1[i][j] - matriz2[i][j];

texto += " "+ver[i][j]+" |";

}

texto += "\n";

}

//Mostrar ventana

JOptionPane.showMessageDialog(null,texto,"MATRIZ",JOptionPane.INFORMATION\_MESSAGE);

}

//Metódo definido por el programador

public void multiplicarMatriz() {

//Arreglos

int fila[] = new int[fc\*fc];

int columna[] = new int[fc\*fc];

//Atributos

int e1,e2,k,y,z,mul;

String texto = "";

opc = 1;

//Ciclo

k = 0;

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

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

//Asignar valores

fila[k] = matriz2[i][j];

k++;

}

}

//Ciclo

k = 0;

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

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

//Asignar valores

columna[k] = matriz1[i][j];

k++;

}

}

//Asignar valores

e1 = 0;

z = 0;

for(int x=0; x<fc; x++) {

//Asignar valores

y = 0;

e2 = 0;

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

//Asignar valor

mul = 0;

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

mul = mul + (fila[j+z]\*columna[j+y]);

}

//Asignar valores

y = y+fc;

ver[e1][e2] = mul;

e2++;

}

//Asignar valores

z = z+fc;

e1++;

}

//Ciclo

texto += "MULTIPLICACION:\n";

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

texto += "|";

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

//Asignar valor

texto += " "+ver[j][i]+" |";

}

texto += "\n";

}

//Mostrar ventana

JOptionPane.showMessageDialog(null,texto,"MATRIZ",JOptionPane.INFORMATION\_MESSAGE);

}

//Metódo definido por el programador

public void voltearMatriz() {

//Atributos

String texto = "";

opc = 0;

//Ciclo

texto += "MATRIZ 1:\n";

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

texto += "|";

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

texto += " "+matriz1[j][i]+" |";

ver1[j][i] = matriz1[j][i];

}

texto += "\n";

}

//Ciclo

texto += "\n\nMATRIZ 2:\n";

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

texto += "|";

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

texto += " "+matriz2[j][i]+" |";

ver2[j][i] = matriz2[j][i];

}

texto += "\n";

}

//Mostrar ventana

JOptionPane.showMessageDialog(null,texto,"MATRIZ",JOptionPane.INFORMATION\_MESSAGE);

}

//Metódo definido por el programador

public void superiorMatriz() {

//Atributos

int t1 = 0, t2 = 0;

//Ciclo

for(int i=1; i<fc; i++) {

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

//Validar valor

if(matriz1[i][j] != 0) {

t1++;

}

}

}

//Ciclo

for(int i=1; i<fc; i++) {

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

//Validar valor

if(matriz2[i][j] != 0) {

t2++;

}

}

}

//Validar valores

if(t1 == 0 && t2 == 0)

//Mostrar ventana

JOptionPane.showMessageDialog(null,"MATRIZ 1 es triangular superior."

+"\nMATRIZ 2 es triangular superior.","MATRIZ",JOptionPane.INFORMATION\_MESSAGE);

if(t1 > 0 && t2 == 0)

//Mostrar ventana

JOptionPane.showMessageDialog(null,"MATRIZ 1 no es triangular superior."

+"\nMATRIZ 2 es triangular superior.","MATRIZ",JOptionPane.INFORMATION\_MESSAGE);

if(t1 == 0 && t2 > 0)

//Mostrar ventana

JOptionPane.showMessageDialog(null,"MATRIZ 1 es triangular superior."

+"\nMATRIZ 2 no es triangular superior.","MATRIZ",JOptionPane.INFORMATION\_MESSAGE);

if(t1 > 0 && t2 > 0)

//Mostrar ventana

JOptionPane.showMessageDialog(null,"MATRIZ 1 no es triangular superior."

+"\nMATRIZ 2 no es triangular superior.","MATRIZ",JOptionPane.INFORMATION\_MESSAGE);

}

//Metódo definido por el programador

public void inferiorMatriz() {

//Atributos

int t1 = 0, t2 = 0;

//Ciclo

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

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

//Validar valor

if(matriz1[i][j] != 0) {

t1++;

}

}

}

//Ciclo

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

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

//Validar valor

if(matriz2[i][j] != 0) {

t1++;

}

}

}

//Validar valores

if(t1 == 0 && t2 == 0)

//Mostrar ventana

JOptionPane.showMessageDialog(null,"MATRIZ 1 es triangular inferior."

+"\nMATRIZ 2 es triangular inferior.","MATRIZ",JOptionPane.INFORMATION\_MESSAGE);

if(t1 > 0 && t2 == 0)

//Mostrar ventana

JOptionPane.showMessageDialog(null,"MATRIZ 1 no es triangular inferior."

+"\nMATRIZ 2 es triangular inferior.","MATRIZ",JOptionPane.INFORMATION\_MESSAGE);

if(t1 == 0 && t2 > 0)

//Mostrar ventana

JOptionPane.showMessageDialog(null,"MATRIZ 1 es triangular inferior."

+"\nMATRIZ 2 no es triangular inferior.","MATRIZ",JOptionPane.INFORMATION\_MESSAGE);

if(t1 > 0 && t2 > 0)

//Mostrar ventana

JOptionPane.showMessageDialog(null,"MATRIZ 1 no es triangular inferior."

+"\nMATRIZ 2 no es triangular inferior.","MATRIZ",JOptionPane.INFORMATION\_MESSAGE);

}

//Metódo definido por el programador

public void guardarMatriz() {

//Variables de instancia

FileWriter fw = null;

PrintWriter pw = null;

//Atributo

String texto = "";

if(opc == 0) {

//Ciclo

texto += "MATRIZ 1:\n";

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

texto += "|";

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

texto += " "+ver1[i][j]+" |";

}

texto += "\n";

}

//Ciclo

texto += "\n\nMATRIZ 2:\n";

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

texto += "|";

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

texto += " "+ver2[i][j]+" |";

}

texto += "\n";

}

} else {

//Ciclo

texto += "MATRIZ:\n";

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

texto += "|";

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

texto += " "+ver[i][j]+" |";

}

texto += "\n";

}

}

//Iniciar excepcion

try {

//Variables de instancia

fw = new FileWriter("Matriz.txt");

pw = new PrintWriter(fw);

//Guardar mensajes y resultado

pw.println("\n=============== MATRIZ ===============\n");

pw.println(texto);

pw.println("\n======================================\n");

} catch(Exception e1) {

//Capturar e imprimir error

JOptionPane.showMessageDialog(null,"ERROR..."+e1.getMessage()+"...","ERROR",JOptionPane.WARNING\_MESSAGE);

} finally {

//Iniciar excepcion

try {

//Validar dato

if (null != fw)

fw.close();

//Mostrar ventana

JOptionPane.showMessageDialog(null,"Matriz guardada","MATRIZ",JOptionPane.INFORMATION\_MESSAGE);

} catch(Exception e2) {

//Capturar e imprimir error

JOptionPane.showMessageDialog(null,"ERROR..."+e2.getMessage()+"...","ERROR",JOptionPane.WARNING\_MESSAGE);

}

}

}

}

//Fin de la clase

**Menu.java**

// Fecha: 07 / Mayo / 2021

// Objetivo: Crear una matriz y realizar algunos ejercicios.

// Elaboro Equipo 4:

// 1) Flores Navarro Emmanuel

// 2) Ramírez Morales Erick Brian

//Inicio de la clase

import javax.swing.JOptionPane;

public class Menu {

//Variables de instancia

private Matriz matriz = new Matriz();

//Método constructor

public Menu() {}

//Metódo definido por el programador

public void iniciarMenu() {

//Atributos

String panel;

int opc;

//Inicio Menu

do{

panel = JOptionPane.showInputDialog(null,"Ingrese el numero de la opcion a ejecutar:"

+"\n 1) Inicio"

+"\n 2) Salir","MENU",JOptionPane.QUESTION\_MESSAGE);

//Iniciar excepcion

try {

//Ingresar un valor

opc = Integer.parseInt(panel);

} catch(Exception e) {

//Capturar y mostrar error

JOptionPane.showMessageDialog(null,"ERROR..."+e.getMessage(),"ERROR",JOptionPane.WARNING\_MESSAGE);

opc = 0;

}

switch(opc) {

case 2 -> {

//Mostrar mensaje y cerrar panel

JOptionPane.showMessageDialog(null,"Gracias por usar este programa, hasta luego!!!");

System.exit(0);

}

case 1 -> {

//Acceder a un método

matriz.crearMatriz();

opc = 2;

}

default -> {

//Mostrar mensaje

JOptionPane.showMessageDialog(null,"Opcion no valida","ERROR",JOptionPane.WARNING\_MESSAGE);

}

}

}while(opc != 2);

//Fin Menu

}

}

//Fin de la clase

**NegativeNumberException.java**

// Fecha: 07 / Mayo / 2021

// Objetivo: Crear una matriz y realizar algunos ejercicios.

// Elaboro Equipo 4:

// 1) Flores Navarro Emmanuel

// 2) Ramírez Morales Erick Brian

//Inicio de la clase

public class NegativeNumberException extends Exception {

//Método constructor

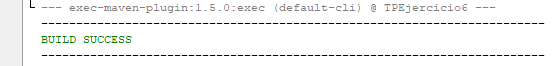
public NegativeNumberException() {}

}

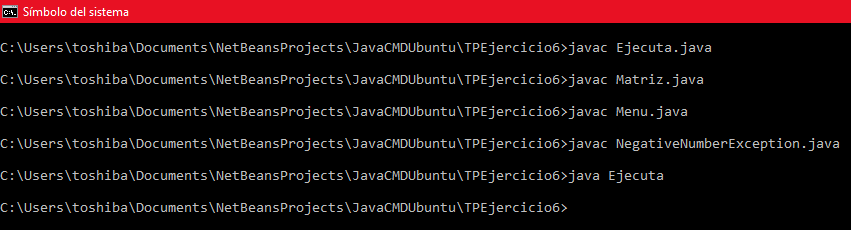
//Fin de la clase

**CAPTURAS DE PANTALLA**

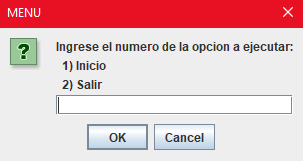
**NetBeans**

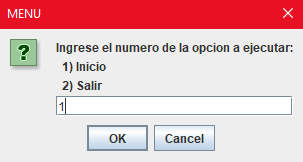


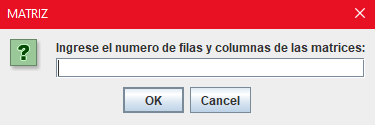
**CMD**

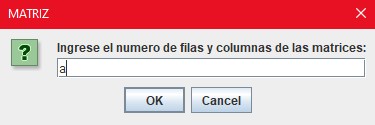


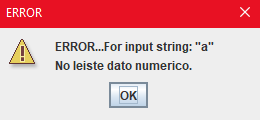
**Ventanas de ejecución**

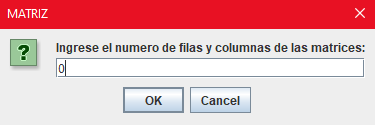


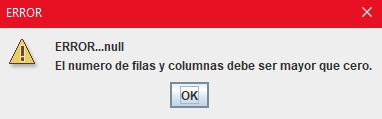


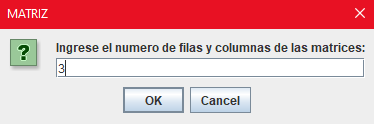


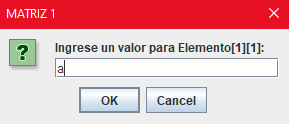


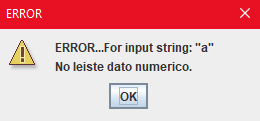


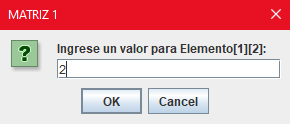
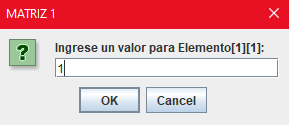


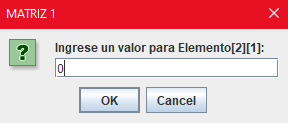
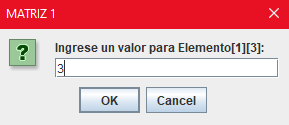


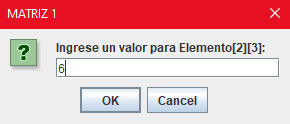
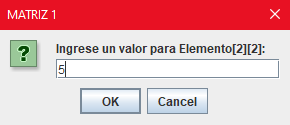


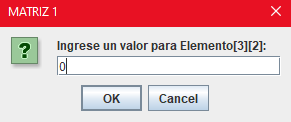
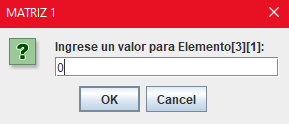


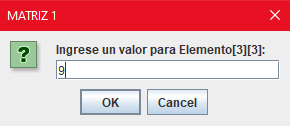


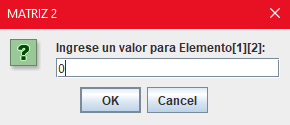
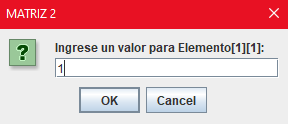


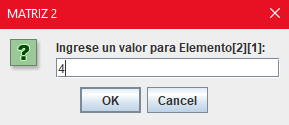
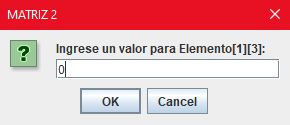


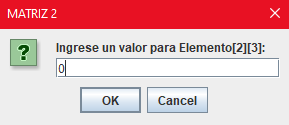
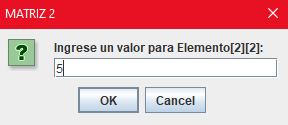


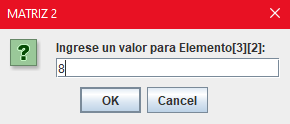
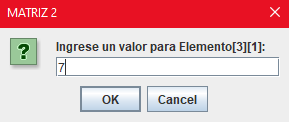


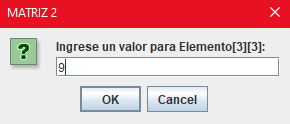


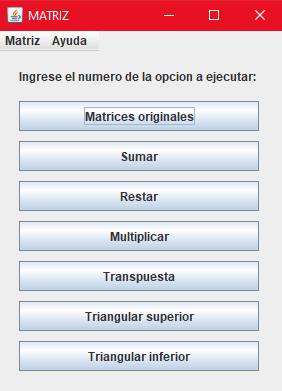


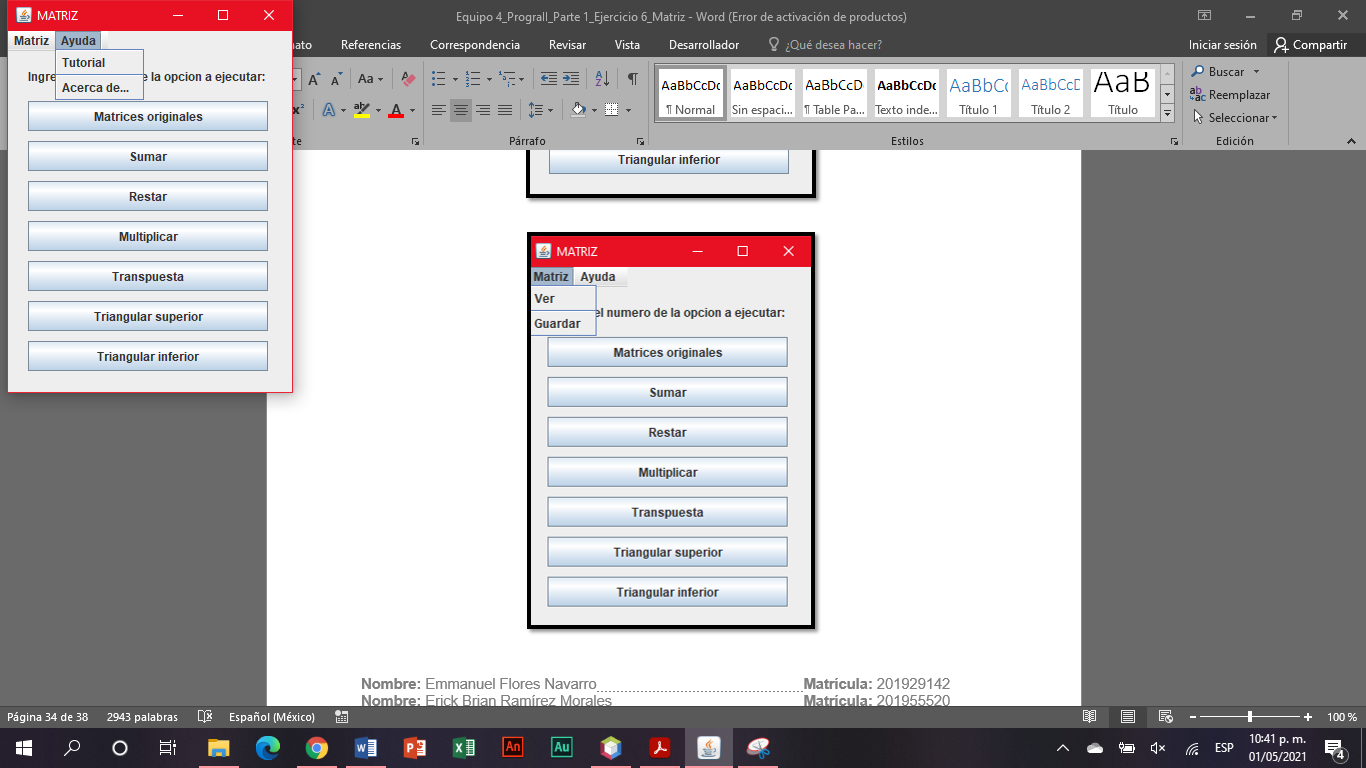
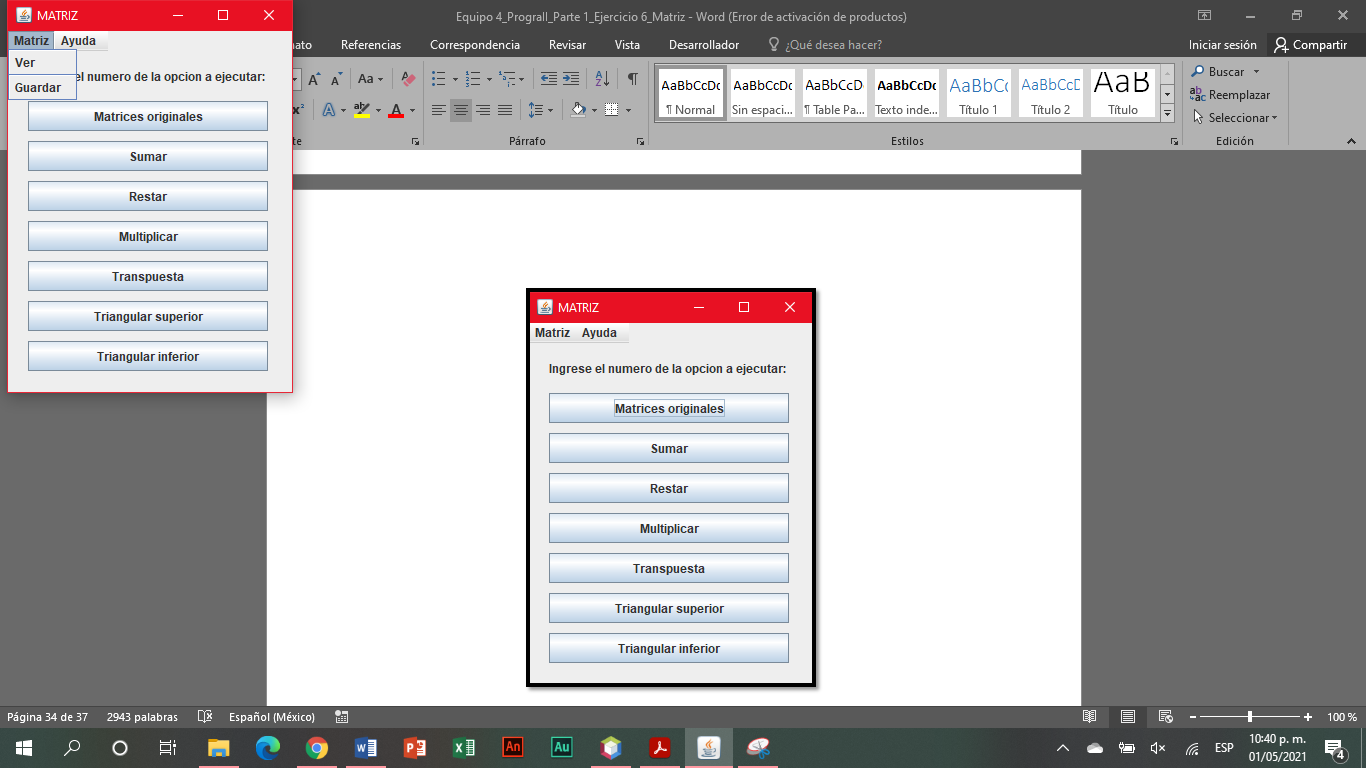


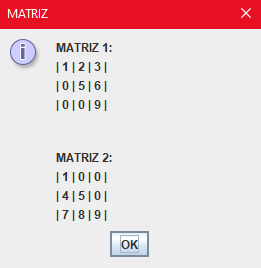


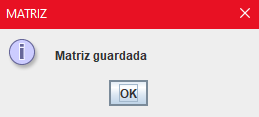


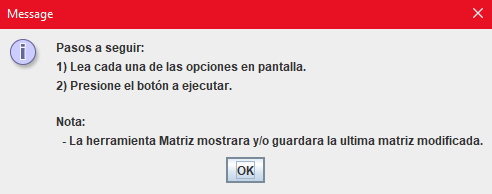


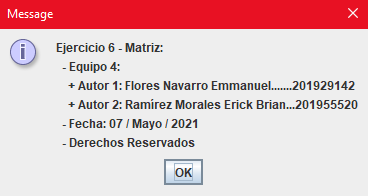


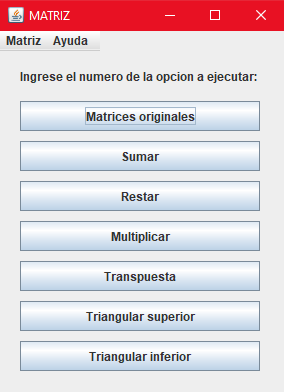


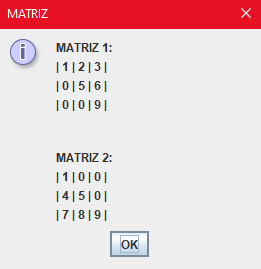


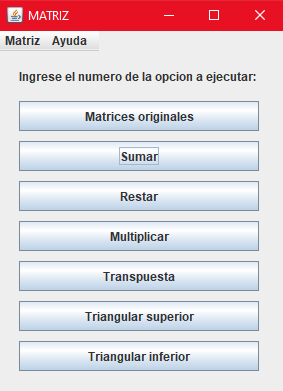


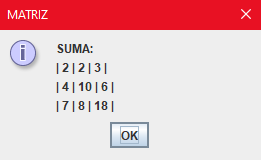


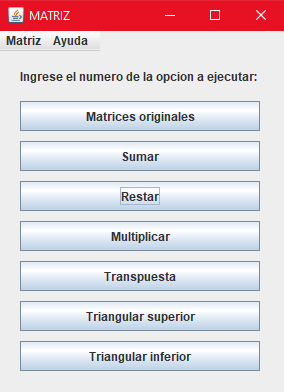




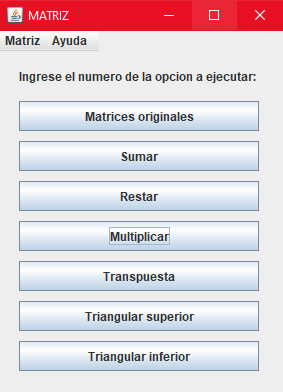




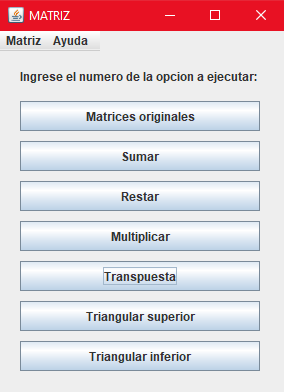


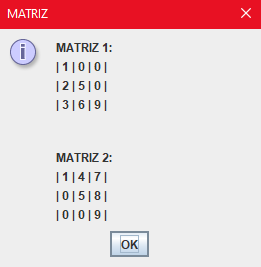


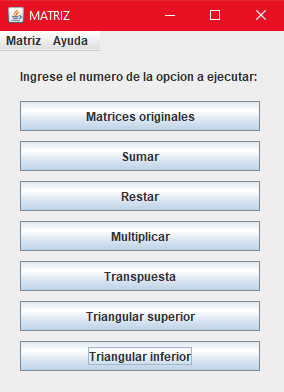
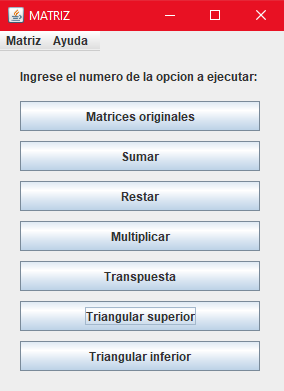


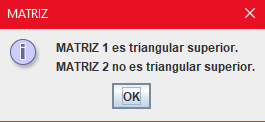
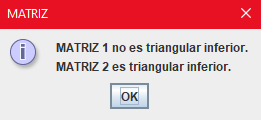


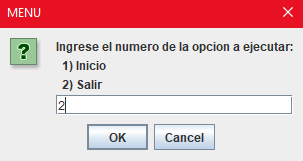


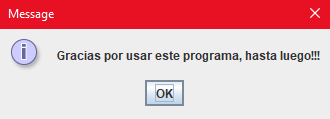










**Matriz.txt**



**CONCLUSIÓN**

Gracias a las matrices se pueden almacenar muchos más datos que los arreglos unidimensionales, ya que se componen de una fila por n columnas, mientras que los multidimensionales se componen de n filas por m columnas, con ellos es posible crear arreglos de tantas dimensiones como las aplicaciones requieran: tres, cuatro o más.

# **BIBLIOGRAFÍA**

Céron, C. (8 de Marzo de 2021). *Practica8a-ProgramacionII-2020*. Obtenido de Blackboard.

Joyanes, L., & Zahonero, I. (2011). Programación en Java 6. En L. Joyanes, & I. Zahonero, *Programación en Java 6* (págs. 251-252). México, D.F: Mc Graw Hill.

**PROGRAMACIÓN II**

**RÚBRICA HOJA DE EVALUACIÓN DE PRÁCTICAS**

MATRICULA: 201955520 FECHA: 07 / Mayo / 2021

NOMBRE: Ramírez Morales Erick Brian No. PRACTICA: Ejercicio 6 INDIVIDUAL ( ) COLABORATIVA ( X )

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Deficiente** | **Suficiente** | **Bueno** | **Excelente** | **Calificación Obtenida** | |
| **CRITERIOS** | **1-5.9** | **6‐7.9** | **8‐9** | **9.1‐10** |
| **%** | **Puntos** |
| **CONOCIMIENTO TEORICO**  **20%** | Conocimiento deficiente de los fundamentos teóricos de POO y no puede aplicarlos en el diseño de clases. | Conocimiento confuso de los fundamentos teóricos de POO y el diseño de las clases y relaciones es incompleto. | Conocimiento claro de los fundamentos teóricos POO pero requiere mejorar el modelado de las clases, sus métodos y sus relaciones entre clases. | Dominio del Conocimiento de los fundamentos teóricos POO y puede aplicarlos de forma completa en el modelado de las clases, métodos y todas las relaciones entre clase. |  |  |
| **EJECUCCIÓN DE LA PRACTICA**  **30%** | No puede realizar la práctica ya que desconoce el entorno de trabajo y desarrollo de la práctica en lenguaje UML y Java. | Realiza la práctica de forma incompleta ya que desconoce el entorno de trabajo del lenguaje UML y Java | Realiza la práctica, pero requiere mejorar en el manejo del entorno de trabajo del lenguaje de programación (sintaxis y  semántica) | Realiza la práctica de forma correcta y completa, demuestra dominio del entorno de trabajo del lenguaje de programación (sintaxis y semántica). |  |  |
| **SOLUCIÓN DE LA PRACTICA**  **40%** | No puede generar las soluciones o programas a los problemas planteados ya que no posee el dominio teórico y práctico del modelado y el lenguaje de programación. | Propone soluciones confusas o programas incompletos a los problemas planteados, ya que carece del dominio del modelado y lenguaje de programación | Genera soluciones con poca profundidad y los programas no están orientados de acuerdo a los problemas solicitados, por lo cual no tiene un dominio profundo de la temática y del Lenguaje. | Genera soluciones con profundidad y los programas son correctos de acuerdo a los problemas planteados, por lo cual demuestra un dominio de la temática y del Lenguaje de Programación. |  |  |
| **ACTITUD DE APRENDER Y COLABORAR EN EQUIPO DE TRABAJO**  **10%** | No posee una actitud proactiva para un aprendizaje autónomo y no le gusta participar y trabajar en equipo. | Posee una actitud propositiva para un aprendizaje autónomo, participa pero no le gusta trabajar en equipo. | Posee una actitud propositiva logrando un aprendizaje autónomo, colaborativo, le gusta trabajar en equipo pero requiere mejorar su participación y portaciones de forma profunda. | Posee una actitud proactiva logrando un aprendizaje autónomo, participa con propuestas concretas y profundas, le gusta trabajar en equipo y asume su responsabilidad dentro para lograr éxito del equipo |  |  |
| **Total**  Equipo 4 | | | | |  |  |

****

**Firma del Alumn@**

**Vo. Bo. Docente**