

	<p>UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p>Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLE-001</p>	<p>Página: 1</p>

## INFORME DE LABORATORIO

### (formato estudiante)

INFORMACIÓN BÁSICA					
ASIGNATURA:	<i>Fundamentos de Programación 2</i>				
TÍTULO DE LA PRÁCTICA:					
NÚMERO DE PRÁCTICA:	<i>5</i>	AÑO LECTIVO:	<i>2024-B</i>	NRO. SEMESTRE:	<i>2</i>
FECHA DE PRESENTACIÓN	<i>18/10/2024</i>	HORA DE PRESENTACIÓN	<i>06:20:00 pm</i>		
INTEGRANTE (s) <i>Jose Manuel Morocco Saico</i>				NOTA (0-20)	<i>Nota colocada por el docente</i>
DOCENTE(s): <i>Mg.Lino José Pinto Oppe</i>					

RESULTADOS Y PRUEBAS	
<p>I. EJERCICIOS RESUELTOS:</p>	

*Clase Soldado:*

```
package EJERCICIOS;
public class Soldado {
    private String nombre;
    private int vida;
    private int fila;
    private int columna;

    public Soldado(String nombre, int puntosVida, int fila, int columna) {
        this.nombre = nombre;
        this.vida = puntosVida;
        this.fila = fila;
        this.columna = columna;
    }

    public String getNombre() {
        return nombre;
    }

    public int getVida() {
        return vida;
    }

    public int getFila() {
        return fila;
    }

    public int getColumna() {
        return columna;
    }

    public String toString() {
        return nombre + " (Vida: " + vida + ", Pos: [" + (fila + 1) + ", " + (columna + 1) + "]);"
    }
}
```

*Clase Main:*

```
package EJERCICIOS;
import java.util.*;
public class VideoJuego2 {
    public static void main(String[] args) {
        Random random = new Random();
        int nSoldados=random.nextInt(10)+1;
        int tTablero=10;
        System.out.println("Se generaran "+nSoldados+" Soldados");

        Soldado[][] tablero = new Soldado[tTablero][tTablero];
        Soldado[] soldadosCreados = new Soldado[nSoldados];

        // Crear soldados y colocarlos en el tablero
        for (int i = 0; i < nSoldados; i++) {
            int fila, columna;
            do {
                fila = random.nextInt(tTablero);
                columna = random.nextInt(tTablero);
            } while (tablero[fila][columna] != null);

            // Generar nivel de vida aleatorio
            int nivelVida = random.nextInt(5) + 1; // Nivel de vida entre 1 y 5
            String nombre = "Soldado" + i;
            Soldado soldado = new Soldado(nombre, nivelVida, fila, columna);
            tablero[fila][columna] = soldado;
            soldadosCreados[i] = soldado;
        }
    }
}
```

```
// Mostrar el tablero
mostrarTablero(tablero);

// Encontrar el soldado con mayor nivel de vida
Soldado soldadoMayorVida = soldadosCreados[0];
int sumaTotalVida = 0;

for (Soldado soldado : soldadosCreados) {
    sumaTotalVida += soldado.getVida();
    if (soldado.getVida() > soldadoMayorVida.getVida()) {
        soldadoMayorVida = soldado;
    }
}

double promedioVida = (double) sumaTotalVida / nSoldados;

System.out.println("\nSoldado con mayor nivel de vida: " + soldadoMayorVida);
System.out.println("Promedio de nivel de vida: " + promedioVida);
System.out.println("Nivel de vida total del ejercito: " + sumaTotalVida);

// Mostrar los soldados en el orden de creación
System.out.println("\nSoldados creados en orden:");
for (Soldado soldado : soldadosCreados) {
    System.out.println(soldado);
}

// Ordenar soldados por nivel de vida (burbuja) y mostrar ranking
ordenarPorVidaBurbuja(soldadosCreados);
System.out.println("\nRanking de soldados (burbuja):");
for (Soldado soldado : soldadosCreados) {
    System.out.println(soldado);
}

// Ordenar soldados por nivel de vida (selección) y mostrar ranking
ordenarPorVidaSeleccion(soldadosCreados);
System.out.println("\nRanking de soldados (seleccion):");
for (Soldado soldado : soldadosCreados) {
    System.out.println(soldado);
}
}
```

```
// Método para mostrar el tablero
public static void mostrarTablero(Soldado[][] tablero) {
    // Mostrar las letras horizontales (A-J) centradas
    System.out.print("    ");
    for (char letra = 'A'; letra <= 'J'; letra++) {
        System.out.print("  " + letra + "  ");
    }
    System.out.println();

    // Mostrar el tablero con numeración vertical (1-10)
    for (int i = 0; i < tablero.length; i++) {
        if (i + 1 < 10) {
            System.out.print("  " + (i + 1) + "  "); // Tres espacios para alinear los números de una cifra
        } else {
            System.out.print((i + 1) + "  "); // Dos espacios para los números de dos cifras
        }
        // Mostrar las celdas del tablero
        for (int j = 0; j < tablero[i].length; j++) {
            if (tablero[i][j] == null) {
                System.out.print("| ____ ");
            } else {
                System.out.print("| S  ");
            }
        }
        System.out.println("|");
    }
}
```

```
// Método para ordenar por vida usando burbuja
public static void ordenarPorVidaBurbuja(Soldado[] soldados) {
    for (int i = 0; i < soldados.length - 1; i++) {
        for (int j = 0; j < soldados.length - i - 1; j++) {
            if (soldados[j].getVida() < soldados[j + 1].getVida()) {
                Soldado temp = soldados[j];
                soldados[j] = soldados[j + 1];
                soldados[j + 1] = temp;
            }
        }
    }
}
```

```
// Método para ordenar por vida usando selección
public static void ordenarPorVidaSeleccion(Soldado[] soldados) {
    for (int i = 0; i < soldados.length - 1; i++) {
        int maxIdx = i;
        for (int j = i + 1; j < soldados.length; j++) {
            if (soldados[j].getVida() > soldados[maxIdx].getVida()) {
                maxIdx = j;
            }
        }
        Soldado temp = soldados[maxIdx];
        soldados[maxIdx] = soldados[i];
        soldados[i] = temp;
    }
}
```

## II.PRUEBAS

### a)Ejecución 1:

Se generaran 6 Soldados

	A	B	C	D	E	F	G	H	I	J
1					S					
2										
3										
4										
5							S			
6								S		
7		S								
8										
9									S	
10		S								

Soldado con mayor nivel de vida: Soldado0 (Vida: 5, Pos: [5,7])

Promedio de nivel de vida: 3.6666666666666665

Nivel de vida total del ejercito: 22

Soldados creados en orden:

Soldado0 (Vida: 5, Pos: [5,7])  
Soldado1 (Vida: 1, Pos: [9,9])  
Soldado2 (Vida: 2, Pos: [1,5])  
Soldado3 (Vida: 5, Pos: [7,2])  
Soldado4 (Vida: 4, Pos: [6,8])  
Soldado5 (Vida: 5, Pos: [10,2])

Ranking de soldados (burbuja):

Soldado0 (Vida: 5, Pos: [5,7])  
Soldado3 (Vida: 5, Pos: [7,2])  
Soldado5 (Vida: 5, Pos: [10,2])  
Soldado4 (Vida: 4, Pos: [6,8])  
Soldado2 (Vida: 2, Pos: [1,5])  
Soldado1 (Vida: 1, Pos: [9,9])

Ranking de soldados (seleccion):

Soldado0 (Vida: 5, Pos: [5,7])  
Soldado3 (Vida: 5, Pos: [7,2])  
Soldado5 (Vida: 5, Pos: [10,2])  
Soldado4 (Vida: 4, Pos: [6,8])  
Soldado2 (Vida: 2, Pos: [1,5])  
Soldado1 (Vida: 1, Pos: [9,9])

## B)Ejecución2:

Se generaran 4 Soldados

	A	B	C	D	E	F	G	H	I	J
1								S		
2										
3										
4										
5										
6									S	
7										
8										
9										
10	S			S						

Soldado con mayor nivel de vida: Soldado0 (Vida: 5, Pos: [10,4])

Promedio de nivel de vida: 3.75

Nivel de vida total del ejercito: 15

Soldados creados en orden:

Soldado0 (Vida: 5, Pos: [10,4])

Soldado1 (Vida: 5, Pos: [1,8])

Soldado2 (Vida: 2, Pos: [10,1])

Soldado3 (Vida: 3, Pos: [6,9])

Ranking de soldados (burbuja):

Soldado0 (Vida: 5, Pos: [10,4])

Soldado1 (Vida: 5, Pos: [1,8])

Soldado3 (Vida: 3, Pos: [6,9])

Soldado2 (Vida: 2, Pos: [10,1])



Ranking de soldados (seleccion):

Soldado0 (Vida: 5, Pos: [10,4])

Soldado1 (Vida: 5, Pos: [1,8])

Soldado3 (Vida: 3, Pos: [6,9])

Soldado2 (Vida: 2, Pos: [10,1])

	<p>UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p>Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLE-001</p>	<p>Página: 8</p>

### *Mi commit al repositorio:*

```

Usuario24B@DESKTOP-6V821ET MINGW64 ~/Downloads/REPOSITORIO_LOCAL (main)
$ git init
Initialized empty Git repository in C:/Users/Usuario24B/Downloads/REPOSITORIO_LOCAL/.git/

Usuario24B@DESKTOP-6V821ET MINGW64 ~/Downloads/REPOSITORIO_LOCAL (master)
$ git add LABORATORIO_05

Usuario24B@DESKTOP-6V821ET MINGW64 ~/Downloads/REPOSITORIO_LOCAL (master)
$ git commit -m "Se agrega el LABORATORIO_05"
[master (root-commit) 2c1b9c8] Se agrega el LABORATORIO_05
 2 files changed, 154 insertions(+)
 create mode 100644 LABORATORIO_05/Soldado.java
 create mode 100644 LABORATORIO_05/VideoJuego2.java

Usuario24B@DESKTOP-6V821ET MINGW64 ~/Downloads/REPOSITORIO_LOCAL (master)
$ git remote add origin https://github.com/JoseMorocco/FP2.git

```

```

Usuario24B@DESKTOP-6V821ET MINGW64 ~/Downloads/REPOSITORIO_LOCAL (master)
$ git branch -m master main

```

```

Usuario24B@DESKTOP-6V821ET MINGW64 ~/Downloads/REPOSITORIO_LOCAL (main)
$ git push -u origin main
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 12 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (5/5), 1.84 KiB | 1.84 MiB/s, done.
Total 5 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), completed with 1 local object.
To https://github.com/JoseMorocco/FP2.git
 84992a7..fcce028  main -> main
branch 'main' set up to track 'origin/main'.

```

*Link a mi repositorio :*

<https://github.com/JoseMorocco/FP2>



## II. RUBRICA:

Contenido y demostración		Puntos	Checklist	Estudiante	Profesor
1. GitHub	Hay enlace URL activo del directorio para el laboratorio hacia su repositorio GitHub con código fuente terminado y fácil de revisar.	2	x	1	
2. Commits	Hay capturas de pantalla de los commits más importantes con sus explicaciones detalladas.  (El profesor puede preguntar para refrendar calificación).	4	x	3	
3. Código fuente	Hay porciones de código fuente importantes con numeración y explicaciones detalladas de sus funciones.	2	x	2	
4. Ejecución	Se incluyen ejecuciones/pruebas del código fuente explicadas gradualmente.	2	x	2	
5. Pregunta	Se responde con completitud a la pregunta formulada en la tarea. (El profesor puede preguntar para refrendar calificación).	2	x	2	
6. Fechas	Las fechas de modificación del código fuente están dentro de los plazos de fecha de entrega establecidos.	2	x	2	
7. Ortografía	El documento no muestra errores ortográficos.	2	x	2	
8. Madurez	El Informe muestra de manera general una evolución de la madurez del código fuente, explicaciones puntuales pero precisas y un acabado impecable. (El profesor puede preguntar para refrendar calificación).	4	x	4	
TOTAL		20		18	

	<p>UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p><b>Formato:</b> Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p><b>Aprobación:</b> 2022/03/01</p>	<p><b>Código:</b> GUIA-PRLE-001</p>	<p><b>Página:</b> 10</p>

## CONCLUSIONES

En este laboratorio, exploramos los arreglos bidimensionales de objetos con el ejercicio Videojuego con el que pudimos comprender más sobre estos además de seguir empleando y practicando los algoritmos de ordenamiento

## METODOLOGÍA DE TRABAJO

- 1.-Elaborar un pequeño pseudocódigo para plantear el programa
- 2.-Elaborar un diagrama de flujo para ver las opciones que quiero que tenga
- 3.-Implementarlas en el programa
- 4.-Corregir errores

## REFERENCIAS Y BIBLIOGRAFÍA