


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

## INFORME DE LABORATORIO

### (formato estudiante)

INFORMACIÓN BÁSICA					
<b>ASIGNATURA:</b>	<i>Fundamentos de Programación 2</i>				
<b>TÍTULO DE LA PRÁCTICA:</b>					
<b>NÚMERO DE PRÁCTICA:</b>	<i>6</i>	<b>AÑO LECTIVO:</b>	<i>2024-B</i>	<b>NRO. SEMESTRE:</b>	<i>2</i>
<b>FECHA DE PRESENTACIÓN</b>	<i>25/10/2024</i>	<b>HORA DE PRESENTACIÓN</b>	<i>05:40:00 pm</i>		
<b>INTEGRANTE (s)</b> <i>Jose Manuel Morocco Saico</i>				<b>NOTA (0-20)</b>	<i>Nota colocada por el docente</i>
<b>DOCENTE(s):</b> <i>Mg.Lino José Pinto Oppe</i>					

RESULTADOS Y PRUEBAS
<p><b>I. EJERCICIOS RESUELTOS:</b></p> <ol style="list-style-type: none"> <li>1. Cree un Proyecto llamado Laboratorio6</li> <li>2. Usted deberá crear las dos clases Soldado.java y VideoJuego3.java. Puede reutilizar lo desarrollado en Laboratorios anteriores.</li> <li>3. Del Soldado nos importa el nombre, puntos de vida, fila y columna (posición en el tablero).</li> <li>4. El juego se desarrollará en el mismo tablero de los laboratorios anteriores. Pero ahora el tablero debe ser un ArrayList bidimensional.</li> <li>5. Tendrá 2 Ejércitos. Inicializar el tablero con n soldados aleatorios entre 1 y 10 para cada Ejército. Cada soldado tendrá un nombre autogenerado: Soldado0X1, Soldado1X1, etc., un valor de puntos de vida autogenerado aleatoriamente [1..5], la fila y columna también autogenerados aleatoriamente (no puede haber 2 soldados en el mismo cuadrado). Se debe mostrar el tablero con todos los soldados creados (distinguir los de un ejército de los del otro ejército). Además de los datos del Soldado con mayor vida de cada ejército, el promedio de puntos de vida de todos los soldados creados por ejército, los datos de todos los soldados por ejército en el orden que fueron creados y un ranking de poder de todos los soldados creados por ejército (del que tiene más nivel de vida al que tiene menos) usando 2 diferentes Marco Aedo López 2 algoritmos de ordenamiento. Finalmente, que muestre qué ejército ganará la batalla (indicar la métrica usada para decidir al ganador de la batalla).</li> </ol>



	<p style="text-align: center;">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p style="text-align: center;"><b>Formato:</b> Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLE-001</p>	<p>Página: 2</p>

*Clase Soldado:*

```

1  package LABORATORIO_06;
2
3  public class Soldado {
4      private String nombre;
5      private int vida;
6      private int fila;
7      private int columna;
8
9      public Soldado(String nombre, int puntosVida, int fila, int columna) {
10         this.nombre = nombre;
11         this.vida = puntosVida;
12         this.fila = fila;
13         this.columna = columna;
14     }
15
16     public String getNombre() {
17         return nombre;
18     }
19
20     public int getVida() {
21         return vida;
22     }
23
24     public int getFila() {
25         return fila;
26     }
27
28     public int getColumna() {
29         return columna;
30     }
31
32     public String toString() {
33         return nombre + " (Vida: " + vida + ", Pos: [" + (fila+1) + "," + (columna+1) + "])";
34     }
35 }

```

	<p style="text-align: center;"><b>UNIVERSIDAD NACIONAL DE SAN AGUSTIN</b>  <b>FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS</b>  <b>ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</b></p>	
<p style="text-align: center;"><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> 3</p>

### Clase VideoJuego3:

```

1 package LABORATORIO_06;
2 import java.util.*;
3 public class VideoJuego3 {
4     public static void main(String[] args) {
5         Random random = new Random();
6         int tTablero = 10;
7
8         // Inicializamos el número de soldados para cada ejército
9         int nSoldados1 = random.nextInt(10) + 1;
10        int nSoldados2 = random.nextInt(10) + 1;
11        System.out.println("Se generarán " + nSoldados1 + " Soldados para el Ejército1");
12        System.out.println("Se generarán " + nSoldados2 + " Soldados para el Ejército2");
13
14        // Creamos el tablero vacío
15        ArrayList<ArrayList<Soldado>> tablero = new ArrayList<>();
16        for (int i = 0; i < tTablero; i++) {
17            ArrayList<Soldado> fila = new ArrayList<>();
18            for (int j = 0; j < tTablero; j++) {
19                fila.add(null); // Agregar null para cada posición de la fila
20            }
21            tablero.add(fila); // Agregar la fila completa al tablero
22        }
23
24        // Inicializamos los ejércitos
25        ArrayList<Soldado> ejercito1 = new ArrayList<>();
26        ArrayList<Soldado> ejercito2 = new ArrayList<>();
27
28        // Colocamos soldados en el tablero
29        ingresarSoldadosTablero(nSoldados1, tablero, ejercito1, 1);
30        ingresarSoldadosTablero(nSoldados2, tablero, ejercito2, 2);
31
32        // Mostramos el tablero con los soldados
33        mostrarTablero(tablero);
34
35        // Mostramos datos de los ejércitos
36        mostrarEstadisticas(ejercito1, "Ejército 1");
37        mostrarEstadisticas(ejercito2, "Ejército 2");
38
39        // Determinamos el ejército ganador
40        String ganador;
41        if (sumaVida(ejercito1) > sumaVida(ejercito2)) {
42            ganador = "Ejército 1";
43        } else {
44            ganador = "Ejército 2";
45        }
46        System.out.println("El ganador de la batalla es (según la suma total de vida de ambos ejércitos): " + ganador);
47    }

```

```
49 // Método para sumar vida total de un ejército
50 public static int sumaVida(ArrayList<Soldado> ejercito) {
51     int suma = 0;
52     for (Soldado soldado : ejercito) {
53         suma += soldado.getVida();
54     }
55     return suma;
56 }
57
58 // Método para ingresar soldados al tablero y al ejército
59 public static void ingresarSoldadosTablero(int nSoldados, ArrayList<ArrayList<Soldado>> tablero,
60     ArrayList<Soldado> ejercito, int idEjercito) {
61     Random random = new Random();
62     int tTablero = tablero.size();
63
64     for (int i = 0; i < nSoldados; i++) {
65         int fila, columna;
66         do {
67             fila = random.nextInt(tTablero);
68             columna = random.nextInt(tTablero);
69         } while (tablero.get(fila).get(columna) != null);
70
71         int vida = random.nextInt(5) + 1;
72         String nombre = "Soldado" + i + "X" + idEjercito;
73         Soldado soldado = new Soldado(nombre, vida, fila, columna);
74         tablero.get(fila).set(columna, soldado);
75         ejercito.add(soldado);
76     }
77 }
78
79 // Método para mostrar el tablero
80 public static void mostrarTablero(ArrayList<ArrayList<Soldado>> tablero) {
81     // Mostrar las letras horizontales (A-J)
82     System.out.print(" ");
83     for (char letra = 'A'; letra <= 'J'; letra++) {
84         System.out.print(" " + letra + " ");
85     }
86     System.out.println();
87
88     // Mostrar el tablero con numeración vertical (1-10)
89     for (int i = 0; i < tablero.size(); i++) {
90         if (i + 1 < 10) {
91             System.out.print(" " + (i + 1) + " "); // Tres espacios para alinear los números de una cifra
92         } else {
93             System.out.print((i + 1) + " "); // Dos espacios para los números de dos cifras
94         }
95         // Mostrar las celdas del tablero
96         for (int j = 0; j < tablero.get(i).size(); j++) {
97             Soldado soldado = tablero.get(i).get(j);
98             if (soldado == null) {
99                 System.out.print("| ____ ");
100             } else if (soldado.getNombre().contains("X1")) {
101                 System.out.print("| S1 ");
102             } else {
103                 System.out.print("| S2 ");
104             }
105         }
106         System.out.println("|");
107     }
108 }
109 }
```

```
110 // Método para mostrar estadísticas del ejército
111 public static void mostrarEstadisticas(ArrayList<Soldado> ejercito, String nombreEjercito) {
112     System.out.println("\nEstadísticas de " + nombreEjercito + ":");
113
114     // Encontrar soldado con mayor vida
115     Soldado soldadoMayorVida = null;
116     int sumaVida = 0;
117
118     for (Soldado soldado : ejercito) {
119         sumaVida += soldado.getVida();
120
121         if (soldadoMayorVida == null) {
122             soldadoMayorVida = soldado;
123         } else if (soldado.getVida() > soldadoMayorVida.getVida()) {
124             soldadoMayorVida = soldado;
125         }
126     }
127
128     // Calcular promedio de vida
129     double promedioVida;
130     if (ejercito.isEmpty()) {
131         promedioVida = 0;
132     } else {
133         promedioVida = (double) sumaVida / ejercito.size();
134         promedioVida = Math.round(promedioVida * 100.0) / 100.0;
135     }
136
137     System.out.println("Soldado con mayor vida: " + soldadoMayorVida);
138     System.out.println("Promedio de vida: " + promedioVida);
139
140     // Mostrar soldados en orden de creación
141     System.out.println("\nSoldados en orden de creación:");
142     for (Soldado soldado : ejercito) {
143         System.out.println(soldado);
144     }
145
146     // Ranking con burbuja
147     ArrayList<Soldado> ejercitoOrdenadoBurbuja = new ArrayList<>(ejercito);
148     ordenarPorVidaBurbuja(ejercitoOrdenadoBurbuja);
149     System.out.println("\nRanking de soldados (burbuja):");
150     for (Soldado soldado : ejercitoOrdenadoBurbuja) {
151         System.out.println(soldado);
152     }
153
154     // Ranking con selección
155     ArrayList<Soldado> ejercitoOrdenadoSeleccion = new ArrayList<>(ejercito);
156     ordenarPorVidaSeleccion(ejercitoOrdenadoSeleccion);
157     System.out.println("\nRanking de soldados (selección):");
158     for (Soldado soldado : ejercitoOrdenadoSeleccion) {
159         System.out.println(soldado);
160     }
161 }
```

```
165 // Método para ordenar por vida usando burbuja
166 public static void ordenarPorVidaBurbuja(ArrayList<Soldado>soldados) {
167     for (int i = 0; i < soldados.size() - 1; i++) {
168         for (int j = 0; j < soldados.size() - i - 1; j++) {
169             if (soldados.get(j).getVida() < soldados.get(j+1).getVida()) {
170                 Soldado temp = soldados.get(j);
171                 soldados.set(j,soldados.get(j+1));
172                 soldados.set(j+1, temp);
173             }
174         }
175     }
176 }
177
178 // Método para ordenar por vida usando selección
179 public static void ordenarPorVidaSeleccion(ArrayList<Soldado>soldados) {
180     for (int i = 0; i < soldados.size() - 1; i++) {
181         int maxIdx = i;
182         for (int j = i + 1; j < soldados.size(); j++) {
183             if (soldados.get(j).getVida() > soldados.get(maxIdx).getVida()) {
184                 maxIdx = j;
185             }
186         }
187         Soldado temp = soldados.get(maxIdx);
188         soldados.set(maxIdx,soldados.get(i));
189         soldados.set(i, temp);
190     }
191 }
192 }
```

## II. PRUEBAS

### Ejecución :

Se generarán 10 Soldados para el Ejército1

Se generarán 6 Soldados para el Ejército2

	A	B	C	D	E	F	G	H	I	J
1		S1		S1						
2						S1				
3										
4	S2			S2			S1			
5								S1		S1
6	S2									
7						S1				
8		S1		S2					S2	
9										
10							S1	S1		S2

Estadísticas de Ejército 1:

Soldado con mayor vida: Soldado2X1 (Vida: 3, Pos: [5,8])

Promedio de vida: 2.0

Soldados en orden de creación:

Soldado0X1 (Vida: 1, Pos: [10,8])

Soldado1X1 (Vida: 2, Pos: [7,6])

Soldado2X1 (Vida: 3, Pos: [5,8])

Soldado3X1 (Vida: 2, Pos: [4,7])

Soldado4X1 (Vida: 1, Pos: [8,2])

Soldado5X1 (Vida: 2, Pos: [1,4])

Soldado6X1 (Vida: 2, Pos: [2,6])

Soldado7X1 (Vida: 3, Pos: [5,10])

Soldado8X1 (Vida: 1, Pos: [1,2])

Soldado9X1 (Vida: 3, Pos: [10,7])

Ranking de soldados (burbuja):

Soldado2X1 (Vida: 3, Pos: [5,8])

Soldado7X1 (Vida: 3, Pos: [5,10])

Soldado9X1 (Vida: 3, Pos: [10,7])

Soldado1X1 (Vida: 2, Pos: [7,6])

Soldado3X1 (Vida: 2, Pos: [4,7])

Soldado5X1 (Vida: 2, Pos: [1,4])

Soldado6X1 (Vida: 2, Pos: [2,6])

Soldado0X1 (Vida: 1, Pos: [10,8])

Soldado4X1 (Vida: 1, Pos: [8,2])

Soldado8X1 (Vida: 1, Pos: [1,2])

**Ranking de soldados (selección):**

Soldado2X1 (Vida: 3, Pos: [5,8])  
Soldado7X1 (Vida: 3, Pos: [5,10])  
Soldado9X1 (Vida: 3, Pos: [10,7])  
Soldado3X1 (Vida: 2, Pos: [4,7])  
Soldado5X1 (Vida: 2, Pos: [1,4])  
Soldado6X1 (Vida: 2, Pos: [2,6])  
Soldado1X1 (Vida: 2, Pos: [7,6])  
Soldado4X1 (Vida: 1, Pos: [8,2])  
Soldado8X1 (Vida: 1, Pos: [1,2])  
Soldado0X1 (Vida: 1, Pos: [10,8])

**Estadísticas de Ejército 2:**

Soldado con mayor vida: Soldado1X2 (Vida: 5, Pos: [4,1])  
Promedio de vida: 3.83

**Soldados en orden de creación:**

Soldado0X2 (Vida: 4, Pos: [8,4])  
Soldado1X2 (Vida: 5, Pos: [4,1])  
Soldado2X2 (Vida: 4, Pos: [8,9])  
Soldado3X2 (Vida: 3, Pos: [4,4])  
Soldado4X2 (Vida: 3, Pos: [10,10])  
Soldado5X2 (Vida: 4, Pos: [6,1])

**Ranking de soldados (burbuja):**



Soldado1X2 (Vida: 5, Pos: [4,1])  
Soldado0X2 (Vida: 4, Pos: [8,4])  
Soldado2X2 (Vida: 4, Pos: [8,9])  
Soldado5X2 (Vida: 4, Pos: [6,1])  
Soldado3X2 (Vida: 3, Pos: [4,4])  
Soldado4X2 (Vida: 3, Pos: [10,10])

**Ranking de soldados (selección):**

Soldado1X2 (Vida: 5, Pos: [4,1])  
Soldado0X2 (Vida: 4, Pos: [8,4])  
Soldado2X2 (Vida: 4, Pos: [8,9])  
Soldado5X2 (Vida: 4, Pos: [6,1])  
Soldado4X2 (Vida: 3, Pos: [10,10])  
Soldado3X2 (Vida: 3, Pos: [4,4])

El ganador de la batalla es (según la suma total de vida de ambos ejércitos): Ejército 2



	<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: 9</p>

Mi commit al repositorio:

```
Windows@DESKTOP-C32KACO MINGW64 ~/Desktop/RepositorioLocal/FP2 (main)
$ git init
Reinitialized existing Git repository in C:/Users/Windows/Desktop/RepositorioLocal/FP2/.git/

Windows@DESKTOP-C32KACO MINGW64 ~/Desktop/RepositorioLocal/FP2 (main)
$ git add LABORATORIO_06

Windows@DESKTOP-C32KACO MINGW64 ~/Desktop/RepositorioLocal/FP2 (main)
$ git commit -m "Se agrega el LABORATORIO_06"
[main 0b6ce52] Se agrega el LABORATORIO_06
2 files changed, 232 insertions(+)
create mode 100644 LABORATORIO_06/Soldado.java
create mode 100644 LABORATORIO_06/VideoJuego3.java

Windows@DESKTOP-C32KACO MINGW64 ~/Desktop/RepositorioLocal/FP2 (main)
$ git push origin main
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 8 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (5/5), 2.39 KiB | 2.39 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
fcce028..0b6ce52 main -> main



Windows@DESKTOP-C32KACO MINGW64 ~/Desktop/RepositorioLocal/FP2 (main)
$ git log
commit 0b6ce524783667908afe503986d7dbe89fbd64da (HEAD -> main, origin/main, origin/HEAD)
Author: JoseMorocco <jmoroccosa@unsa.edu.pe>
Date: Fri Oct 25 15:31:16 2024 -0500

    Se agrega el LABORATORIO_06
```

**Link a mi repositorio: <https://github.com/JoseMorocco/FP2>**

### III. RUBRICA:

Contenido y demostración		Puntos	Checklis t	Estudiant e	Profeso r
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	2	
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	1	
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	3	
TOTAL		20		17	

	<p style="text-align: center;"><b>UNIVERSIDAD NACIONAL DE SAN AGUSTIN</b>  <b>FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS</b>  <b>ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</b></p>	
<p style="text-align: center;"><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> 11</p>

## CONCLUSIONES

En este laboratorio, exploramos el uso de ArrayList en un contexto bidimensional de objetos. He notado que una de sus principales ventajas es su capacidad de redimensionarse dinámicamente, lo que facilita agregar o eliminar elementos sin preocupaciones por el tamaño fijo de un array. Además, ArrayList ofrece métodos útiles para manipular y acceder a los datos, simplificando la gestión de colecciones en proyectos más complejos. Podemos concluir que ArrayList mejora la eficiencia y flexibilidad del manejo de datos, haciendo que el desarrollo de programas sea más organizado y efectivo como en el manejo de objetos propios.

## METODOLOGÍA DE TRABAJO

- 1.-Con la base de los laboratorios anteriores analizar los cambios y adiciones que hare
- 2.-Elaborar un pequeño pseudocódigo para plantear el programa nuevo
- 3.-Implementarlas en el programa
- 4.-Corregir errores

## REFERENCIAS Y BIBLIOGRAFÍA

<https://puntocomnoesunlenguaje.blogspot.com/2013/02/arraylist-de-objetos-en-java.html>  
<https://parzibyte.me/blog/2020/08/30/java-ordenamiento-seleccion/>  
<https://parzibyte.me/blog/2019/12/26/ordenamiento-burbuja-java/>