
	<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		
<b>Aprobación:</b> 2022/03/01	<b>Código:</b> GUIA-PRLE-001	<b>Página:</b> 1

## INFORME DE LABORATORIO

### (formato estudiante)

INFORMACIÓN BÁSICA					
ASIGNATURA:	Fundamentos de la programación 2				
TÍTULO DE LA PRÁCTICA:	Arreglos de objetos				
NÚMERO DE PRÁCTICA:	3	AÑO LECTIVO:	2024	NRO. SEMESTRE:	//
FECHA DE PRESENTACIÓN	10/05/2024	HORA DE PRESENTACIÓN	23:59:59		
INTEGRANTE (s) Layme Salas Rodrigo Fabricio				NOTA (0-20)	
DOCENTE(s): Lino Jose Pinto Oppe					

RESULTADOS Y PRUEBAS	
<b>I. EJERCICIOS RESUELTOS:</b> <b>CÓDIGO DEL EJERCICIO 1:</b> <b>CLASE NAVE:</b>	
<pre> 1 public class Nave { 2     private String nombre; 3     private int fila; 4     private String columna; 5     private boolean estado; 6     private int puntos; 7     // Metodos mutadores 8     public void setNombre(String n) { 9         nombre = n; 10    } 11    public void setFila(int f) { 12        fila = f; 13    } 14    public void setColumna(String c) { 15        columna = c; 16    } 17    public void setEstado(boolean e) { 18        estado = e; 19    } 20    public void setPuntos(int p) { 21        puntos = p; 22    } 23    // Metodos accesorios 24    public String getNombre() { 25        return nombre; 26    } 27    public int getFila() { 28        return fila; 29    } 30    public String getColumna() { 31        return columna; 32    } 33    public boolean getEstado() { 34        return estado; 35    } 36    public int getPuntos() { 37        return puntos; 38    } // Completar con otros métodos necesarios 39    public String toString() { </pre>	<pre>         return "Nombre: " + nombre + "\nFila: " + fila + "\nColumna: "         + columna + "\nEstado: " + estado + "\nPuntos: "         + puntos;     } </pre>

```
1  /*Propósito: Simular una flota de naves en un juego DemoBatalla*/
2  import java.util.*;
3  public class DemoBatalla {
4      public static Scanner sc = new Scanner(System.in); //hacer el scanner una variable global
5      public static void main(String[] args) {
6          Nave[] misNaves = new Nave[10];
7          String nomb, col;
8          int fil, punt;
9          boolean est;
10         for (int i = 0; i < misNaves.length; i++) { //Inicialización de naves
11             System.out.println("NAVE " + (i + 1) + ":");
12             System.out.print("Nombre: ");
13             nomb = sc.next();
14             System.out.println("Fila ");
15             fil = sc.nextInt();
16             System.out.print("Columna: ");
17             col = sc.next();
18             System.out.print("Estado: ");
19             est = sc.nextBoolean();
20             System.out.print("Puntos: ");
21             punt = sc.nextInt();
22             misNaves[i] = new Nave(); // Se crea un objeto Nave y se asigna su referencia a misNaves
23             misNaves[i].setNombre(nomb);
24             misNaves[i].setFila(fil);
25             misNaves[i].setColumna(col);
26             misNaves[i].setEstado(est);
27             misNaves[i].setPuntos(punt);
28         }
29         System.out.println("\nNaves creadas:");
30         mostrarNaves(misNaves); // Método para mostrar todas las naves en su orden ingresado
31         mostrarPorNombre(misNaves); // Método para mostrar todas las naves de un nombre que se pide por teclado
32         mostrarPorPuntos(misNaves); // Método para mostrar todas las naves con un número de puntos inferior o igual al número de puntos que
33         // se pide por teclado
34         System.out.println("\nNave con mayor número de puntos: \n" + mostrarMayorPuntos(misNaves)); // Método que devuelve la Nave con mayor
35         // número de Puntos
36         Nave[] misNavesDesordenadas = navesDesordenadas(misNaves, misNaves); // Método que copia el array original a otro de forma desordenado
37         System.out.println("NAVES DESORDENADAS: ");
38         mostrarNaves(misNavesDesordenadas); // Método para mostrar ahora las naves desordenadas
```

```

37 }
38 public static void mostrarNaves(Nave[] flota) {
39     for(int i = 0; i<flota.length; i++){
40         System.out.println("Nave " + (i+1) + ":");
41         System.out.println(flota[i].toString());
42     }
43 }
44 public static void mostrarPorNombre(Nave[] flota) {
45     System.out.println("Ingrese el nombre de las naves que desea mostrar:");
46     String nombreNave = sc.next();
47     for(int i = 0; i<flota.length; i++)
48         if(flota[i].getNombre().equals(nombreNave)){
49             System.out.println("Nave " + (i+1) + ":");
50             System.out.println(flota[i].toString());
51         }
52 }
53 public static void mostrarPorPuntos(Nave[] flota) {
54     System.out.println("Ingrese una cantidad de puntos, se mostrará las naves que tengan menor o igual puntaje: ");
55     int puntosNave = sc.nextInt();
56     for(int i = 0; i<flota.length; i++){
57         if(flota[i].getPuntos() == puntosNave || flota[i].getPuntos() < puntosNave){ // Condición para mostrar naves
58             System.out.println("Nave " + (i+1) + ":");
59             System.out.println(flota[i].toString());
60         }
61     }
62 public static Nave mostrarMayorPuntos(Nave[] flota) {
63     Nave naveMayorPuntaje = flota[0];
64     for(int i = 1; i<flota.length; i++)
65         if(flota[i].getPuntos() > naveMayorPuntaje.getPuntos())
66             naveMayorPuntaje = flota[i];
67     return naveMayorPuntaje;
68 }
69 public static Nave[] navesDesordenadas(Nave[] flota, Nave[] misNavesDesordenadas){
70     System.arraycopy(flota, 0, misNavesDesordenadas, 0, flota.length); // No quiero afectar al array original así que creo una copia
71     for (int indexOriginal = misNavesDesordenadas.length - 1; indexOriginal > 0; indexOriginal--){
72         int indexIntercambio = (int)(Math.random()*(indexOriginal + 1));
73         Nave temp = misNavesDesordenadas[indexOriginal]; // Variable temporal para completar el intercambio de posiciones aleatorio
74         misNavesDesordenadas[indexOriginal] = misNavesDesordenadas[indexIntercambio];
75         misNavesDesordenadas[indexIntercambio] = temp;
76     }
77     return misNavesDesordenadas;
78 }
79 }

```



## PRUEBAS DEL EJERCICIO 1:

```
NAVE 1:  
Nombre: a  
Fila  
1  
Columna: 2  
Estado: true  
Puntos: 12  
NAVE 2:  
Nombre: b  
Fila  
2  
Columna: 3  
Estado: false  
Puntos: 11  
NAVE 3:  
Nombre: c  
Fila  
1
```

```
Ingrese el nombre de las naves que desea mostrar:  
a  
Nave 1:  
Nombre: a  
Fila: 1  
Columna: 2  
Estado: true  
Puntos: 12
```

```
Ingrese una cantidad de puntos, se mostrará las naves que tengan menor o igual puntaje:  
5  
Nave 3:  
Nombre: c  
Fila: 1  
Columna: 0  
Estado: true  
Puntos: 3  
Nave 7:  
Nombre: g  
Fila: 5  
Columna: 8  
Estado: false  
Puntos: 2
```

```
Nave con mayor número de puntos:  
Nombre: i  
Fila: 6  
Columna: 7  
Estado: false  
Puntos: 100
```

	<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>	
<p align="center"><b>Formato:</b> Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p align="center">Código: GUIA-PRLE-001</p>	<p align="right">Página: 5</p>

#### NAVES DESORDENADAS:

```
Nave 1:
Nombre: e
Fila: 7
Columna: 3
Estado: false
Puntos: 50
Nave 2:
Nombre: a
Fila: 1
Columna: 2
Estado: true
Puntos: 12
Nave 3:
Nombre: g
Fila: 5
Columna: 8
Estado: false
```

*Ya no comienza por la nave a*



#### CÓDIGO DEL EJERCICIO 2:

##### CLASE SOLDADO:

```
1 public class Soldado {
2     private String nombre;
3     private int vida;
4     public void setNombre(String nombre) {
5         this.nombre = nombre;
6     }
7     public void setVida(int vida) {
8         this.vida = vida;
9     }
10    public String getNombre() {
11        return nombre;
12    }
13    public int getVida() {
14        return vida;
15    }
16    public String toString() {
17        return "Nombre: " + nombre + "\nNivel de vida: " + vida;
18    }
19 }
```

```
1  /*Propósito: Inicializar soldados */
2  import java.util.*;
3  public class Ejercicio4PC1v2 {
4      Run | Debug
5      public static void main(String[] args) {
6          @SuppressWarnings("resource")
7          Scanner scan = new Scanner(System.in);
8          Soldado[] soldados = new Soldado[5];
9          for(int i = 0; i<soldados.length; i++){
10              Soldado persona = new Soldado(); // Creación de objetos dentro del array
11              System.out.println("Ingrese el nombre del soldado " + (i+1));
12              persona.setNombre(scan.next());
13              System.out.println("Ingrese el nivel de vida ");
14              persona.setVida(scan.nextInt());
15              soldados[i] = persona; // Completar la inicialización
16          }
17          imprimirSoldados(soldados);
18      }
19      public static void imprimirSoldados(Soldado[] soldados){ // Imprime los datos de los soldados
20          for(int i = 0; i<soldados.length; i++){
21              System.out.println("Soldado "+ (i+1) + ":");
22              System.out.println(soldados[i].toString());
23          }
24      }
25  }
```

**PRUEBAS DEL EJERCICIO 2:**

	<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> 7</p>

```

Ingrese el nombre del soldado 1
Mau
Ingrese el nivel de vida
5
Ingrese el nombre del soldado 2
Ema
Ingrese el nivel de vida
4
Ingrese el nombre del soldado 3
Ger
Ingrese el nivel de vida
6
Ingrese el nombre del soldado 4
San
Ingrese el nivel de vida
8
Ingrese el nombre del soldado 5
Misael
Ingrese el nivel de vida
100
Soldado 1:
Nombre: Mau
Nivel de vida: 5
Soldado 2:
Nombre: Ema
Nivel de vida: 4
Soldado 3:
Nombre: Ger
Nivel de vida: 6
Soldado 4:
Nombre: San
Nivel de vida: 8
Soldado 5:
Nombre: Misael
Nivel de vida: 100

```



### CÓDIGO DEL EJERCICIO 3:

```

1  /*Propósito: Simular una guerra entre 2 ejércitos */
2  public class Ejercicio5PC2v2 {
3      Run | Debug
4      public static void main(String[] args) {
5          String[] ejercito1 = inicializarEjercito((int) (Math.random() * 5 + 1)); // Inicializa con el parámetro de cantidad de soldados
6          String[] ejercito2 = inicializarEjercito((int) (Math.random() * 5 + 1));
7          mostrarEjercito(ejercito1, tipo:1);
8          mostrarEjercito(ejercito2, tipo:2);
9          mostrarGanador(ejercito1, ejercito2);
10     }
11     public static String[] inicializarEjercito(int cantidad) { // Inicializa los soldados con su nombre
12         String[] ejercito = new String[cantidad];
13         for (int i = 0; i < ejercito.length; i++)
14             ejercito[i] = "Soldado" + i;
15         return ejercito;
16     }
17     public static void mostrarEjercito(String[] ejercito, int tipo) { // Muestra los soldados de un ejército
18         System.out.println("\nEjercito " + tipo + ": ");
19         for (int i = 0; i < ejercito.length; i++)
20             System.out.println(ejercito[i]);
21     }
22     public static void mostrarGanador(String[] ejercito1, String[] ejercito2) { // Muestra los resultados del juego
23         if (ejercito1.length > ejercito2.length)
24             System.out.println("\n¡¡Gana el ejercito 1 con " + ejercito1.length + " soldados!!!");
25         else if (ejercito1.length < ejercito2.length)
26             System.out.println("\n¡¡Gana el ejercito 2 con " + ejercito2.length + " soldados!!!");
27         else
28             System.out.println("\n¡¡Empatan los ejércitos con " + ejercito1.length + " soldados!!!");
29     }
30 }

```

### PRUEBAS DEL EJERCICIO 2:

	<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		
<b>Aprobación:</b> 2022/03/01	<b>Código:</b> GUIA-PRLE-001	<b>Página:</b> 8

```
Ejercito 1:
Soldado0
Soldado1
Soldado2

Ejercito 2:
Soldado0
Soldado1
Soldado2
Soldado3

¡¡¡Gana el ejercito 2 con 4 soldados!!!
```

```
Ejercito 1:
Soldado0
Soldado1
Soldado2

Ejercito 2:
Soldado0
Soldado1
Soldado2

¡¡¡Empatan los ejercitos con 3 soldados!!!
```

## II. PRUEBAS

*¿Con que valores comprobaste que tu práctica estuviera correcta?*

*Con valores String, int, booleanos dentro de los objetos creados. Fueron correctos desde el primer commit.*

*¿Qué resultado esperabas obtener para cada valor de entrada?*

*Esperaba obtener la modificación exitosa dentro del objeto, tuve unos problemas al momento de trasladar las entradas con el ArrayCopy pero pronto lo solucioné.*

*¿Qué valor o comportamiento obtuviste para cada valor de entrada?*



*Obtuve la salida correcta con el toString de los objetos y sus valores dentro de cada variable.*

## III. CUESTIONARIO:

*Las pruebas de los ejercicios están arriba.*

**PRUEBAS COMMIT VÍA GIT BASH:**



	<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>

```

USER@DESKTOP-BDOAPPE MINGW64 ~/LAB3 (main)
$ git init
Initialized empty Git repository in C:/Users/USER/LAB3/.git/

USER@DESKTOP-BDOAPPE MINGW64 ~/LAB3 (master)
$ git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
  LAYME_SALAS_LABORATORIO_03/

nothing added to commit but untracked files present (use "git add" to track)

USER@DESKTOP-BDOAPPE MINGW64 ~/LAB3 (master)
$ git add LAYME_SALAS_LABORATORIO_03/

USER@DESKTOP-BDOAPPE MINGW64 ~/LAB3 (master)
$ git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   LAYME_SALAS_LABORATORIO_03/DemoBatalla.class
    new file:   LAYME_SALAS_LABORATORIO_03/DemoBatalla.java
    new file:   LAYME_SALAS_LABORATORIO_03/Ejercicio4PC1v2.java
    new file:   LAYME_SALAS_LABORATORIO_03/Ejercicio5PC2v2.java
    new file:   LAYME_SALAS_LABORATORIO_03/Nave.class
    new file:   LAYME_SALAS_LABORATORIO_03/Nave.java
    new file:   LAYME_SALAS_LABORATORIO_03/Soldado.java

```

*La imagen de arriba inicialice mi repositorio local y añadí mi trabajo para subirlo a github.*

```



USER@DESKTOP-BDOAPPE MINGW64 ~/LAB3 (master)
$ git branch -M main

USER@DESKTOP-BDOAPPE MINGW64 ~/LAB3 (main)
$ git remote add origin https://github.com/F4brici0L4yme/PF2.git

USER@DESKTOP-BDOAPPE MINGW64 ~/LAB3 (main)
$ git commit -m "lab3 terminado"
[main (root-commit) f4b7542] lab3 terminado
7 files changed, 113 insertions(+)
create mode 100644 LAYME_SALAS_LABORATORIO_03/DemoBatalla.class
create mode 100644 LAYME_SALAS_LABORATORIO_03/DemoBatalla.java
create mode 100644 LAYME_SALAS_LABORATORIO_03/Ejercicio4PC1v2.java
create mode 100644 LAYME_SALAS_LABORATORIO_03/Ejercicio5PC2v2.java
create mode 100644 LAYME_SALAS_LABORATORIO_03/Nave.class
create mode 100644 LAYME_SALAS_LABORATORIO_03/Nave.java
create mode 100644 LAYME_SALAS_LABORATORIO_03/Soldado.java

```

Aquí cambio de rama al Main, para no subir a la rama Master que viene por defecto e hice el commit con el mensaje de terminado.

	<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;">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: 10</p>

```

USER@DESKTOP-BDOAPPE MINGW64 ~/LAB3 (main)
$ git pull origin main --rebase
remote: Enumerating objects: 12, done.
remote: Counting objects: 100% (12/12), done.
remote: Compressing objects: 100% (11/11), done.
remote: Total 12 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)

Unpacking objects: 100% (12/12), 5.95 KiB | 13.00 KiB/s, done.
From https://github.com/F4briciOL4yme/PF2
* branch          main          -> FETCH_HEAD
* [new branch]     main          -> origin/main
Successfully rebased and updated refs/heads/main.

USER@DESKTOP-BDOAPPE MINGW64 ~/LAB3 (main)
$ git push origin main
Enumerating objects: 11, done.
Counting objects: 100% (11/11), done.
Delta compression using up to 4 threads
Compressing objects: 100% (9/9), done.
Writing objects: 100% (10/10), 4.51 KiB | 1.50 MiB/s, done.
Total 10 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/F4briciOL4yme/PF2.git
 cdf7958..d018a7f  main -> main

```

**// En el aula del laboratorio teníamos el problema de cambiar de rama, la solución era hacer un git pull origin main --rebase, esto para traer los cambios que ya hice en git hub a mi repositorio local y que no exista errores de estado entre los estados del repositorio local y remoto.**

## CONCLUSIONES

*Concluyo que usar objetos dentro de Arrays da un código más limpio en el main, no debo de almacenar todas las variables dentro del mismo archivo, sino que estos están en otro.*



## METODOLOGÍA DE TRABAJO

*Mi metodología consiste en recordar problemas pasados similares y ver si puedo aplicar lo mismo o que podría cambiar para implementarlo de una forma directa. Además de comentar bloques de código para ir probando de poco en poco mi programa y ver si funciona correctamente.*



## REFERENCIAS Y BIBLIOGRAFÍA

[1] M. W. Aedo Lopez, *Fundamentos de programación*, 1st ed., vol. 1. Calle Paucarpata, Puerta 5, Área de Ingenierías: EDITORIAL UNSA, 2019, p. 100.

## RUBRICA DE CALIFICACIÓN DE LABORATORIO

	<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>	
<b>Formato:</b> Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		
<b>Aprobación:</b> 2022/03/01	<b>Código:</b> GUIA-PRLE-001	<b>Página:</b> 11

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	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	4	
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	4	X	3	

	<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>	
<p align="center"><b>Formato:</b> Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p><b>Aprobación:</b> 2022/03/01</p>	<p align="center"><b>Código:</b> GUIA-PRLE-001</p>	<p align="right"><b>Página:</b> 12</p>

	<p>evolución de la madurez del código fuente, explicaciones puntuales pero precisas y un acabado impecable. (El profesor puede preguntar para refrendar calificación).</p>				
<p align="center"><b>TOTAL</b></p>		<p align="center">20</p>		<p align="center">19</p>	