



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

## INFORME DE LABORATORIO

### (formato estudiante)

INFORMACIÓN BÁSICA					
ASIGNATURA:	Fundamentos de la programación II				
TÍTULO DE LA PRÁCTICA:	Arreglos de Objetos				
NÚMERO DE PRÁCTICA:	03	AÑO LECTIVO:	2024	NRO. SEMESTRE:	02
FECHA DE PRESENTACIÓN	06/10/2024	HORA DE PRESENTACIÓN	00/22/00		
INTEGRANTE (s) José León Enrique Hatches Curo				NOTA (0-20)	Nota colocada por el docente
DOCENTE(s): Ing. Lino Pinto Oppe					

RESULTADOS Y PRUEBAS
<p><b>I. EJERCICIOS RESUELTOS:</b></p> <p>El estudiante coloca la evidencia de los ejercicios propuestos realizados en la sesión de laboratorio, en el tiempo o duración indicado por el docente.</p> <p>El docente debe colocar la retroalimentación por cada ejercicio que el estudiante/grupo ha presentado.</p> <p>Repositorio en GitHub:</p> <p>1ER COMMIT (en main): <a href="https://github.com/LeonHatches/Laboratorio_03">Laboratorio_03/Laboratorio03 at main · LeonHatches/Laboratorio_03 (github.com)</a></p> <p>2DO COMMIT (en master): <a href="https://github.com/LeonHatches/Laboratorio_03">Laboratorio_03/Laboratorio03 at master · LeonHatches/Laboratorio_03 (github.com)</a></p>

	<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><b>Aprobación:</b> 2022/03/01</p>	<p><b>Código:</b> GUIA-PRLE-001</p>	<p><b>Página:</b> 2</p>

## ARREGLOS DE OBJETOS

### ENUNCIADO 01:

*Analice, complete y pruebe el Código de la clase DemoBatalla.*

- *Primero, en este método hace que se muestren todas las naves por orden de ingreso y por igual nombre ingresado.*

```
// Método para mostrar todas las naves
public static void mostrarNaves (Nave [] flota)
{
    for (int i = 0 ; i < flota.length ; i++)
        System.out.println("\t| NAVE "+(i+1)+" | \n"+flota[i]+" \n");
}

// Método para mostrar todas las naves de un nombre que se pide por teclado
public static void mostrarPorNombre (Nave [] flota)
{
    System.out.println ("| TODAS LAS NAVES POR NOMBRE INGRESADO |");
    System.out.print ("Ingrese nombre: ");
    String nombre = sc.next();

    for (int i = 0 ; i < flota.length ; i++)
    {
        if ( flota[i].getNombre().equals( nombre ) )
            System.out.println("\t| NAVE ENCONTRADA | \n"+flota[i]+" \n");
    }

    System.out.println ();
}
```

- *Luego, este método hace que se muestre la nave con mayor puntaje.*

```
// Método que devuelve la Nave con mayor número de Puntos
public static Nave mostrarMayorPuntos(Nave [] flota)
{
    int index = 0;
    for (int i = 1 ; i < flota.length ; i++)
    {
        if (flota[index].getPuntos() < flota[i].getPuntos())
            index = i;
    }
    return flota[index];
}
```

- Por último, estos métodos en conjunto hacen que se muestre el arreglo de naves al azar con otro arreglo de números que funcionan como el orden de los índices del arreglo principal. Con el "ContieneIndex" revisa si el número se repite y el "AleatorioFalta", muestra si falta llenar de aleatorios el arreglo de números.

```
// Método que devuelva un nuevo arreglo de objetos con todos los objetos
// previamente ingresados pero aleatoriamente desordenados
public static void mostrarAleatorio (Nave [] flota)
{
    int contador = 0, aleatorio [] = {-1,-1,-1,-1,-1,-1,-1,-1,-1,-1};

    System.out.println("\n\t NAVES AL AZAR |");

    while ( aleatorioFalta(aleatorio) )
    {
        int random = (int) (Math.random() * 10);

        if ( !ContieneIndex (aleatorio, random) )
        {
            aleatorio[contador] = random;
            contador++;
        }
    }

    for (int i = 0 ; i < flota.length ; i++)
        System.out.println("\t NAVE "+(i+1)+" |\n"+flota[ aleatorio[i] ]+"\n");
}

// Metodo que revisa si el numero aleatorio ya esta en el arreglo
public static boolean ContieneIndex (int [] aleatorio, int numero)
{
    for (int azar : aleatorio)
        if (azar == numero) return true;

    return false;
}

// Metodo que revisa si falta llenar algun numero aleatorio
public static boolean aleatorioFalta (int [] aleatorio)
{
    for (int i = 0 ; i < aleatorio.length ; i++)
    {
        if (aleatorio[i] == -1) return true;
    }
    return false;
}
```

**ENUNCIADO 02:**

*Solucionar la Actividad 4 de la Práctica 1 pero usando arreglo de objetos.*

*Primero, se muestra la creación del arreglo de objetos de la clase Soldado.*

```
public static void main (String [] args)
{
    Soldado [] soldados = new Soldado [5];
}
```

- *Lo más importante es la creación del objeto para cada elemento del arreglo.*

```
public static void IngresarNombres (Soldado [] soldados)
{
    Scanner sc = new Scanner (System.in);

    for (int i = 0 ; i < soldados.length ; i++)
    {
        soldados[i] = new Soldado ();
    }
}
```

- *Por último, este método muestra lo que pide el ejercicio.*

```
public static void Mostrar (Soldado [] soldados)
{
    System.out.println("\nEl nombre de sus soldados son:");

    for (int i = 0 ; i < soldados.length ; i++)
        System.out.println
            ("- " + soldados[i].getName() + " ----- Vida: " + soldados[i].getVida());
}
```

	<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> 5</p>

### ENUNCIADO 03:

*Solucionar la Actividad 5 de la Práctica 1 pero usando arreglos de objetos.*

- *Primero, se crean los arreglos de la clase Soldado.*

```
public static void main (String [] args)
{
    Soldado [] soldadosA = new Soldado [Random ()];
    Soldado [] soldadosB = new Soldado [Random ()];
}
```

- *Por último, lo más importante es la creación del objeto para cada elemento del arreglo.*

```
public static void InicializarSoldados (Soldado [] soldados)
{
    for (int i = 0 ; i < soldados.length ; i++)
    {
        soldados[i] = new Soldado ();
        soldados[i].setName("Soldado"+i);
    }
}
```

	<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: 6</p>

## II. PRUEBAS

### EJECUCIÓN:

#### GIT:

- Aquí inicié mi repositorio local.

```
Equipo@DESKTOP-VTEJTT1 MINGW32 ~/git/Repositorios/laboratorio03
$ git init
Initialized empty Git repository in C:/Users/Equipo/git/Repositorios/laboratorio03/.git/
```

- Aquí añadiré mi laboratorio.

```
Equipo@DESKTOP-VTEJTT1 MINGW32 ~/git/Repositorios/laboratorio03 (master)
$ git add Laboratorio03/
```

- Aquí hice mi segundo commit que ya está con todos los ejercicios finalizados, al contrario del primero, que solo tiene "Nave.java" y "DemoBatalla.java".

```
Equipo@DESKTOP-VTEJTT1 MINGW32 ~/git/Repositorios/laboratorio03 (master)
$ git commit -m "Segundo Commit - Lab completo"
[master (root-commit) 9e23b15] Segundo Commit - Lab completo
5 files changed, 342 insertions(+)
create mode 100644 Laboratorio03/DemoBatalla.java
create mode 100644 Laboratorio03/Enunciado02.java
create mode 100644 Laboratorio03/Enunciado03.java
create mode 100644 Laboratorio03/Nave.java
create mode 100644 Laboratorio03/Soldado.java
```

- Conecté mi repositorio.

```
Equipo@DESKTOP-VTEJTT1 MINGW32 ~/git/Repositorios/laboratorio03 (master)
$ git remote add origin https://github.com/LeonHatches/Laboratorio_03.git
```

- Hice "push" para añadirlo a la rama "master" de mi repositorio, ya que en main no me lo permitía.

```
Equipo@DESKTOP-VTEJTT1 MINGW32 ~/git/Repositorios/laboratorio03 (master)
$ git push -u origin master
Enumerating objects: 8, done.
Counting objects: 100% (8/8), done.
Delta compression using up to 4 threads
Compressing objects: 100% (7/7), done.
Writing objects: 100% (8/8), 3.10 KiB | 211.00 KiB/s, done.
Total 8 (delta 1), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (1/1), done.
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:   https://github.com/LeonHatches/Laboratorio_03/pull/new/master
remote:
To https://github.com/LeonHatches/Laboratorio_03.git
 * [new branch]      master -> master
branch 'master' set up to track 'origin/master'.
```

**ENUNCIADO 01:**

- Esta es la parte inicial del código donde se ingresan los datos.

```
| NAVE 1|
Nombre: a
Fila: 1
Columna: 1
Estado: false
Puntos: 1

| NAVE 2|
Nombre: b
Fila: 1
Columna: 1
Estado: false
Puntos: 2

| NAVE 3|
Nombre: c
Fila: 1
Columna: 1
Estado: false
Puntos: 3

| NAVE 4|
```

- Luego se muestran todas las naves.

```
| NAVES CREADAS |
| NAVE 1 |
NOMBRE:      a
FILA:        1
COLUMNA:     1
ESTADO:      false
PUNTOS:      1

| NAVE 2 |
NOMBRE:      b
FILA:        1
COLUMNA:     1
ESTADO:      false
PUNTOS:      2

| NAVE 3 |
NOMBRE:      c
FILA:        1
COLUMNA:     1
ESTADO:      false
PUNTOS:      3
```

- Aquí se buscan las naves por nombre.

```
| TODAS LAS NAVES POR NOMBRE INGRESADO |
Ingrese nombre: a
| NAVE ENCONTRADA |
NOMBRE:      a
FILA:        1
COLUMNA:     1
ESTADO:      false
PUNTOS:      1
```

- Aquí se muestran según un máximo de puntos.

```
| TODAS LAS NAVES SEGUN PUNTOS |
Ingrese un maximo de puntos: 3
| NAVE 1 |
NOMBRE:      a
FILA:        1
COLUMNA:     1
ESTADO:      false
PUNTOS:      1

| NAVE 2 |
NOMBRE:      b
FILA:        1
COLUMNA:     1
ESTADO:      false
PUNTOS:      2

| NAVE 3 |
NOMBRE:      c
FILA:        1
COLUMNA:     1
ESTADO:      false
PUNTOS:      3
```

- Aquí se muestra solo la nave con mayor puntaje.

```
Nave con mayor numero de puntos:
NOMBRE:      e
FILA:        1
COLUMNA:     1
ESTADO:      false
PUNTOS:      5
```



- Por último, se muestran las naves en un orden al azar.

```
| NAVES AL AZAR |
| NAVE 1 |
NOMBRE:      a
FILA:        1
COLUMNA:     1
ESTADO:      false
PUNTOS:      1

| NAVE 2 |
NOMBRE:      c
FILA:        1
COLUMNA:     1
ESTADO:      false
PUNTOS:      3

| NAVE 3 |
NOMBRE:      e
FILA:        1
COLUMNA:     1
ESTADO:      false
PUNTOS:      5

| NAVE 4 |
NOMBRE:      b
```

#### ENUNCIADO 02:

- En esta captura, se ve como se ingresan y muestran los soldados con su respectiva vida.

```
SOLDADOS EN GUERRA

Ingrese nombre del soldado: jose
Ingrese nombre del soldado: aron
Ingrese nombre del soldado: leon
Ingrese nombre del soldado: lian
Ingrese nombre del soldado: juan

El nombre de sus soldados son:
- jose ----- Vida: 4
- aron ----- Vida: 4
- leon ----- Vida: 4
- lian ----- Vida: 4
- juan ----- Vida: 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>	
<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> 10

### ENUNCIADO 03:

- En esta captura, se muestra la ejecución del ganador de la batalla según número de soldados.

```

SOLDADOS EN GUERRA

      EJERCITO A
El nombre de sus soldados son:
- Soldado0
- Soldado1
- Soldado2
- Soldado3
- Soldado4

      EJERCITO B
El nombre de sus soldados son:
- Soldado0
- Soldado1

      GANADOR: EJERCITO A
El nombre de sus soldados son:
- Soldado0
- Soldado1
- Soldado2
- Soldado3
- Soldado4

```

### III. CUESTIONARIO:

Colocar la evidencia de las respuestas realizadas al cuestionario enunciado en la guía práctica de laboratorio.

Preguntas:

¿Con qué valores comprobaste que tu práctica estuviera correcta?

- Con valores de entrada tipo String (cadena), que son representadas por nombres; por valores booleanos (boolean) como estados o con valores de entrada tipo entero (int) como las filas, columnas y puntos.

¿Qué resultado esperabas obtener para cada valor de entrada?

- Esperaba resultados de salida del arreglo que dependen de cada ejercicio.

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

- Obtuve los valores que deseaba para satisfacer cada enunciado.

	<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 conclusión, para que los códigos de los enunciados puedan tener un buen funcionamiento, el uso de arreglos será necesario de implementar en los datos de las naves como soldados, así también para el uso de números al azar y evitar su repetición, así mostremos gran optimización, accesibilidad y orden al momento de programar. Así también, la importancia de los métodos al instante de trabajar y tener un código que tenga una facilidad de observación al momento de las correcciones del código.*

## METODOLOGÍA DE TRABAJO

*Colocar la metodología de trabajo que ha utilizado el estudiante o el grupo para resolver la práctica, es decir el procedimiento/secuencia de pasos en forma general.*

- *La secuencia que utilicé fue, primero, analizar el código base ya proporcionado o que había realizado y tratar de darle parte de la solución en mi mente. Segundo, me dediqué a armar la estructura de los algoritmos faltantes, los distintos métodos y clases Soldado - Nave. Tercero, analizar errores y formas de optimizar el código para que se visualice de una manera entendible. Cuarto y último, compilar o correr el código para visualizar su funcionamiento o en dado caso no funcione, volver al paso 3 para corregir.*

## REFERENCIAS Y BIBLIOGRAFÍA

*Ninguna referencia externa.*

## RÚBRICA PARA EL CONTENIDO DEL INFORME Y DEMOSTRACIÓN

El alumno debe marcar o dejar en blanco en celdas de la columna Checklist si cumplió con el ítem correspondiente.

Si un alumno supera la fecha de entrega, su calificación será sobre la nota mínima aprobada, siempre y cuando cumpla con todos los ítems.

El alumno debe autocalificarse en la columna Estudiante de acuerdo a la siguiente tabla:

Tabla 1: Niveles de desempeño

Nivel				
Puntos	Insatisfactorio 25 %	En Proceso 50 %	Satisfactorio 75 %	Sobresaliente 100 %
2.0	0.5	1.0	1.5	2.0
4.0	1.0	2.0	3.0	4.0

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	0	
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		16	

Tabla 2: Rúbrica para contenido del Informe y demostración