

	<p align="center">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
Formato: 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					
ASIGNATURA:	LABORATORIO DE FUNDAMENTOS DE PROGRAMACIÓN 2				
TÍTULO DE LA PRÁCTICA:					
NÚMERO DE PRÁCTICA:	4	AÑO LECTIVO:	2024	NRO. SEMESTRE:	II
FECHA DE PRESENTACIÓN	18/10/2024	HORA DE PRESENTACIÓN			
INTEGRANTE (s)	Diego Aristides Cervantes Apaza			NOTA (0-20)	
DOCENTE(s):				Lino Jose Pinto Oppe	

RESULTADOS Y PRUEBAS
<p>I. EJERCICIOS RESUELTOS:</p> <p>Ejercicio 01.</p> <ul style="list-style-type: none"> - Cree un Proyecto llamado Laboratorio4 - Usted podrá reutilizar las dos clases Nave.java y DemoBatalla.java. creadas en Laboratorio - Completar el Código de la clase DemoBatalla. <div data-bbox="363 1541 1289 1886">  </div>

CÓDIGO:

```
1 //Autor: Diego Aristides Cervantes Apaza
2 //Problema: Utilizando el laboratorio 03 utilizar métodos de ordenamiento
3 package laboratorio04;
4
5 import java.util.ArrayList;
6 import java.util.Scanner;
7 import java.util.Random;
8
9 public class DemoBatalla {
10     public static void main(String[] args) {
11         Nave[] misNaves = new Nave[10];
12         Scanner sc = new Scanner(System.in);
13         String nomb, col;
14         int fil, punt;
15         boolean est;
16
17         // Captura de datos para cada nave
18         for (int i = 0; i < misNaves.length; i++) {
19             System.out.println("Nave " + (i + 1));
20             System.out.print("Nombre: ");
21             nomb = sc.next();
22             System.out.print("Fila (1-10): ");
23             fil = sc.nextInt();
24             System.out.print("Columna (A-J): ");
25             col = sc.next().toUpperCase();
26             System.out.print("Estado (true/false): ");
27             est = sc.nextBoolean();
28             System.out.print("Puntos: ");
29             punt = sc.nextInt();
30
31             // Se crea un objeto Nave y se asigna su referencia a misNaves
32             misNaves[i] = new Nave();
33             misNaves[i].setNombre(nomb);
34             misNaves[i].setFila(fil);
35             misNaves[i].setColumna(col);
36             misNaves[i].setEstado(est);
```

```

37         misNaves[i].setPuntos(punt);
38     }
39
40     // Mostrar naves
41     System.out.println("\nNaves creadas:");
42     mostrarNaves(misNaves);
43
44     // Búsqueda lineal
45     System.out.println("\nIngrese el nombre a buscar con búsqueda lineal:");
46     String nombreBuscado = sc.next();
47     int index = busquedaLinealNombre(misNaves, nombreBuscado);
48     if (index != -1) {
49         System.out.println("Nave encontrada: " + misNaves[index].getNombre() + " con " + misNaves[index].getPuntos() + " puntos.");
50     } else {
51         System.out.println("Nave no encontrada.");
52     }
53
54     // Ordenar por puntos usando burbuja
55     System.out.println("\nNaves ordenadas por puntos (Burbuja):");
56     ordenarPorPuntosBurbuja(misNaves);
57     mostrarNaves(misNaves);
58
59     // Ordenar por nombre de A a Z usando selección
60     System.out.println("\nNaves ordenadas por nombre (Selección):");
61     ordenarPorNombreSeleccion(misNaves);
62     mostrarNaves(misNaves);
63
64     // Búsqueda binaria
65     System.out.println("\nIngrese el nombre a buscar con búsqueda binaria:");
66     nombreBuscado = sc.next();
67     ordenarPorNombreBurbuja(misNaves); // Ordenar antes de usar búsqueda binaria
68     int binIndex = busquedaBinariaNombre(misNaves, nombreBuscado);
69     if (binIndex != -1) {
70         System.out.println("Nave encontrada: " + misNaves[binIndex].getNombre() + " con " + misNaves[binIndex].getPuntos() + " puntos.");
71     } else {
72         System.out.println("Nave no encontrada.");
73     }
74 }
75
76 // Método para mostrar todas las naves
77 public static void mostrarNaves(Nave[] flota) {
78     for (Nave nave : flota) {
79         System.out.println(nave.getNombre() + " | Fila: " + nave.getFila() + " | Columna: " + nave.getColumna() +
80             " | Estado: " + nave.getEstado() + " | Puntos: " + nave.getPuntos());
81     }
82 }
83
84 // Método para buscar la primera nave con un nombre que se pidió por teclado
85 public static int busquedaLinealNombre(Nave[] flota, String s) {
86     for (int i = 0; i < flota.length; i++) {
87         if (flota[i].getNombre().equalsIgnoreCase(s)) {
88             return i;
89         }
90     }
91     return -1; // No encontrado
92 }
93
94 // Método que ordena por número de puntos de menor a mayor (Burbuja)
95 public static void ordenarPorPuntosBurbuja(Nave[] flota) {
96     int n = flota.length;
97     for (int i = 0; i < n - 1; i++) {
98         for (int j = 0; j < n - 1 - i; j++) {
99             if (flota[j].getPuntos() > flota[j + 1].getPuntos()) {
100                 // Intercambiar
101                 Nave temp = flota[j];
102                 flota[j] = flota[j + 1];
103                 flota[j + 1] = temp;
104             }
105         }
106     }
107 }

```



```
109 // Método que ordena por nombre de A a Z (Selección)
110 public static void ordenarPorNombreSeleccion(Nave[] flota) {
111     int n = flota.length;
112     for (int i = 0; i < n - 1; i++) {
113         int minIndex = i;
114         for (int j = i + 1; j < n; j++) {
115             if (flota[j].getNombre().compareToIgnoreCase(flota[minIndex].getNombre()) < 0) {
116                 minIndex = j;
117             }
118         }
119         // Intercambiar
120         Nave temp = flota[minIndex];
121         flota[minIndex] = flota[i];
122         flota[i] = temp;
123     }
124 }
125
126 // Método que ordena por nombre de A a Z (Burbuja)
127 public static void ordenarPorNombreBurbuja(Nave[] flota) {
128     int n = flota.length;
129     for (int i = 0; i < n - 1; i++) {
130         for (int j = 0; j < n - 1 - i; j++) {
131             if (flota[j].getNombre().compareToIgnoreCase(flota[j + 1].getNombre()) > 0) {
132                 // Intercambiar
133                 Nave temp = flota[j];
134                 flota[j] = flota[j + 1];
135                 flota[j + 1] = temp;
136             }
137         }
138     }
139 }
140
141 // Método para buscar la primera nave con un nombre que se pidió por teclado (Búsqueda Binaria)
142 public static int busquedaBinariaNombre(Nave[] flota, String s) {
143     int inicio = 0;
144     int fin = flota.length - 1;
145     while (inicio <= fin) {
```

```

146         int comparacion = flota[medio].getNombre().compareToIgnoreCase(s);
147         if (comparacion == 0) {
148             return medio;
149         } else if (comparacion < 0) {
150             inicio = medio + 1;
151         } else {
152             fin = medio - 1;
153         }
154     }
155     return -1; // No encontrado
156 }
157
158 // Método que ordena por puntos de mayor a menor (Inserción)
159 public static void ordenarPorPuntosInsercion(Nave[] flota) {
160     for (int i = 1; i < flota.length; i++) {
161         Nave actual = flota[i];
162         int j = i - 1;
163         while (j >= 0 && flota[j].getPuntos() < actual.getPuntos()) {
164             flota[j + 1] = flota[j];
165             j--;
166         }
167         flota[j + 1] = actual;
168     }
169 }
170
171 // Método que ordena por nombre de Z a A (Inserción)
172 public static void ordenarPorNombreInsercion(Nave[] flota) {
173     for (int i = 1; i < flota.length; i++) {
174         Nave actual = flota[i];
175         int j = i - 1;
176         while (j >= 0 && flota[j].getNombre().compareToIgnoreCase(actual.getNombre()) < 0) {
177             flota[j + 1] = flota[j];
178             j--;
179         }
180         flota[j + 1] = actual;
181     }
182 }
183 }

```

II. PRUEBAS

	<p align="center">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		
Aprobación: 2022/03/01	Código: GUIA-PRLE-001	Página: 6

Ejercicio 01.



- Cree un Proyecto llamado Laboratorio4
- Usted podrá reutilizar las dos clases Nave.java y DemoBatalla.java. creadas en Laboratorio
- Completar el Código de la clase DemoBatalla.

DIGITANDO NAVES Y SUS ATRIBUTO

```
run:
Nave 1
Nombre: aberca
Fila (1-10): 3
Columna (A-J): j
Estado (true/false): true
Puntos: 10
Nave 2
Nombre: loli
Fila (1-10): 4
Columna (A-J): a
Estado (true/false): false
Puntos: 11
Nave 3
Nombre: minerva
Fila (1-10): 4
Columna (A-J): i
Estado (true/false): false
Puntos: 2
Nave 4
Nombre: babian
Fila (1-10): 7
Columna (A-J): c
Estado (true/false): false
Puntos: 12
Nave 5
Nombre: berlawski
Fila (1-10): 4
Columna (A-J): c
Estado (true/false): true
Puntos: 20
Nave 6
Nombre: iker
Fila (1-10): 6
Columna (A-J): h
Estado (true/false): false
Puntos: 5
```



```
Nave 7
Nombre: ilal
Fila (1-10): 6
Columna (A-J): g
Estado (true/false): true
Puntos: 4
Nave 8
Nombre: guysensei
Fila (1-10): 2
Columna (A-J): d
Estado (true/false): true
Puntos: 15
Nave 9
Nombre: oliv
Fila (1-10): 8
Columna (A-J): e
Estado (true/false): true
Puntos: 17
Nave 10
Nombre: galvan
Fila (1-10): 9
Columna (A-J): f
Estado (true/false): true
Puntos: 1
```

	<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: 7</p>

MUESTRA DE LISTAS CREADAS

Naves creadas:

```

aberca | Fila: 3 | Columna: J | Estado: true | Puntos: 10
loli | Fila: 4 | Columna: A | Estado: false | Puntos: 11
minerva | Fila: 4 | Columna: I | Estado: false | Puntos: 2
babian | Fila: 7 | Columna: C | Estado: false | Puntos: 12
berlawski | Fila: 4 | Columna: C | Estado: true | Puntos: 20
iker | Fila: 6 | Columna: H | Estado: false | Puntos: 5
ilal | Fila: 6 | Columna: G | Estado: true | Puntos: 4
guysensei | Fila: 2 | Columna: D | Estado: true | Puntos: 15
oliv | Fila: 8 | Columna: E | Estado: true | Puntos: 17
galvan | Fila: 9 | Columna: F | Estado: true | Puntos: 1

```

BÚSQUEDA LINEAL A PARTIR DEL NOMBRE

Ingrese el nombre a buscar con búsqueda lineal:

loli

Nave encontrada: loli con 11 puntos.



BÚSQUEDA BURBUJA POR PUNTOS MENOR A MAYOR

Naves ordenadas por puntos (Burbuja):

```

galvan | Fila: 9 | Columna: F | Estado: true | Puntos: 1
minerva | Fila: 4 | Columna: I | Estado: false | Puntos: 2
ilal | Fila: 6 | Columna: G | Estado: true | Puntos: 4
iker | Fila: 6 | Columna: H | Estado: false | Puntos: 5
aberca | Fila: 3 | Columna: J | Estado: true | Puntos: 10
loli | Fila: 4 | Columna: A | Estado: false | Puntos: 11
babian | Fila: 7 | Columna: C | Estado: false | Puntos: 12
guysensei | Fila: 2 | Columna: D | Estado: true | Puntos: 15
oliv | Fila: 8 | Columna: E | Estado: true | Puntos: 17
berlawski | Fila: 4 | Columna: C | Estado: true | Puntos: 20

```

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

BÚSQUEDA ORDENADA POR NOMBRE(SELECCIÓN):

```
Naves ordenadas por nombre (Selección):
aberca | Fila: 3 | Columna: J | Estado: true | Puntos: 10
babian | Fila: 7 | Columna: C | Estado: false | Puntos: 12
berlawski | Fila: 4 | Columna: C | Estado: true | Puntos: 20
galvan | Fila: 9 | Columna: F | Estado: true | Puntos: 1
guysensei | Fila: 2 | Columna: D | Estado: true | Puntos: 15
iker | Fila: 6 | Columna: H | Estado: false | Puntos: 5
ilal | Fila: 6 | Columna: G | Estado: true | Puntos: 4
loli | Fila: 4 | Columna: A | Estado: false | Puntos: 11
minerva | Fila: 4 | Columna: I | Estado: false | Puntos: 2
oliv | Fila: 8 | Columna: E | Estado: true | Puntos: 17
```



BÚSQUEDA BINARIA POR NOMBRE:

```
Ingrese el nombre a buscar con búsqueda binaria:
babian
Nave encontrada: babian con 12 puntos.
BUILD SUCCESSFUL (total time: 3 minutes 32 seconds)
```

III. CUESTIONARIO

LINK DEL REPOSITORIO EN GITHUB:

<https://github.com/Aristides-1/AVANCESLABSDIEGOCERVANTES>

	<p align="center">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		
Aprobación: 2022/03/01	Código: GUIA-PRLE-001	Página: 9

CONCLUSIONES

A través de la presente hemos hecho varias clases que se crean para la resolución final del ejercicio, que conlleva un paradigma de creación de objetos estándares, arrays, uso de métodos , utilización de atributos correspondientes y algoritmos para el enfoque el cual está destinado el ejercicio: la creación de un programa que simule creación de naves con datos y estos mismos destinados a “jugar entre ellos mismos” y explorar distintas facetas tanto útiles como interesantes una vez con datos ordenados y a disposición.



METODOLOGÍA DE TRABAJO

Se ha trabajado en NetBeans distribución de Apache, a través de la prueba y error, cambiando códigos, digitando a mano posibles códigos y maneras, luego ir a ejecución, hallando finalmente la manera corecta

REFERENCIAS Y BIBLIOGRAFÍA

RÚBRICA DEL ESTUDIANTE:

¡OJO! Anexar a su informe

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

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. Commits	Hay capturas de pantalla de los commits más importantes con sus explicaciones detalladas. (El profesor puede preguntar para refrendar calificación).	4		0	
3. Código fuente	Hay porciones de código fuente importantes con numeración y explicaciones detalladas de sus funciones.	2		1	
4. Ejecución	Se incluyen ejecuciones/pruebas del código fuente explicadas gradualmente.	2		2	
5. Pregunta	Se responde con completitud a la pregunta formulada en la tarea. (El profesor puede preguntar para refrendar calificación).	2		1	
6. Fechas	Las fechas de modificación del código fuente están dentro de los plazos de fecha de entrega establecidos.	2		0	
7. Ortografía	El documento no muestra errores ortográficos.	2		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		2	
TOTAL		20		10	