
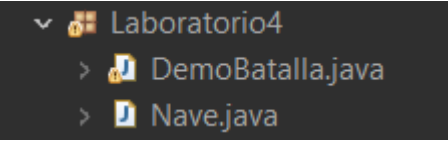
	<p align="center">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		
Aprobación: 2022/03/01	Código: GUIA-PRLE-001	Página: 1

INFORME DE LABORATORIO

(formato estudiante)

INFORMACIÓN BÁSICA					
ASIGNATURA:	Fundamentos de la programación 2				
TÍTULO DE LA PRÁCTICA:	Arreglos de Objetos, Búsquedas y Ordenamientos				
NÚMERO DE PRÁCTICA:	04	AÑO LECTIVO:	2024	NRO. SEMESTRE:	II
FECHA DE PRESENTACIÓN	13/10/2024	HORA DE PRESENTACIÓN	17/00/00		
INTEGRANTE (s) Hilacondo Begazo, Emanuel David				NOTA (0-20)	
DOCENTE(s): Pinto Oppe, Lino José					

RESULTADOS Y PRUEBAS
<p>I. EJERCICIOS RESUELTOS:</p> <p><i>El estudiante coloca la evidencia de los ejercicios propuestos realizados en la sesión de laboratorio, en el tiempo o duración indicado por el docente.</i></p> <p><i>El docente debe colocar la retroalimentación por cada ejercicio que el estudiante/grupo ha presentado</i></p> <p>EJERCICIO 1:</p> <p>DEMO BATALLA:</p> <p>1. Cree un Proyecto llamado Laboratorio4.</p> <div data-bbox="145 1496 595 1635">  </div> <p>2. Usted podrá reutilizar las dos clases Nave.java y DemoBatalla.java. creadas en Laboratorio 3.</p> <p>3. Completar el Código de la clase DemoBatalla.</p>

PROGRAMA PRINCIPAL:

En el main se crean las naves con sus diferentes atributos, también se probará con métodos los diferentes tipos de búsqueda y ordenamientos vistos en clase.

```
1 package Laboratorio4;
2 import java.util.*;
3 /*Laboratorio Nr4 - Ejercicio1
4 *Autor: Hilacondo Begazo,Emanuel David
5 *Colaboró: con nadie
6 *Tiempo: --
7 */
8
9 public class DemoBatalla {
10     public static void main(String[] args) {
11         Nave[] misNaves = new Nave[8];
12         Scanner sc = new Scanner(System.in);
13         String nomb, col;
14         int fil, punt;
15         boolean est;
16
17         for (int i = 0; i < misNaves.length; i++) {
18             System.out.println("\nNave " + (i + 1));
19             System.out.print("Nombre: ");
20             nomb = sc.next();
21             System.out.print("Fila: ");
22             fil = sc.nextInt();
23             System.out.print("Columna: ");
24             col = sc.next();
25             System.out.print("Estado: ");
26             est = sc.nextBoolean();
27             System.out.print("Puntos: ");
28             punt = sc.nextInt();
29
30             misNaves[i] = new Nave(); // Se crea un objeto Nave y se asigna su referencia a misNaves
31             misNaves[i].setNombre(nomb);
32             misNaves[i].setFila(fil);
33             misNaves[i].setColumna(col);
34             misNaves[i].setEstado(est);
35             misNaves[i].setPuntos(punt);
36         }
37
38         System.out.println("\nNaves creadas:");
39         mostrarNaves(misNaves);
40         mostrarPorNombre(misNaves);
41         mostrarPorPuntos(misNaves);
42         System.out.println("\nNave con mayor número de puntos: " + mostrarMayorPuntos(misNaves));
43     }
44 }
```

```

43
44 // Mostrar los datos de la nave con dicho nombre, mensaje de "no encontrado" en caso contrario
45 System.out.println("Ingrese un nombre para buscar:");
46 nomb = sc.next();
47 int pos = busquedaLinealNombre(misNaves, nomb); // Asumiendo 'nombre' debe ser 'nomb'
48 mostrarBusqueda(pos, misNaves);
49
50 System.out.println("\nOrdenamiento burbuja por puntos:");
51 ordenarPorPuntosBurbuja(misNaves);
52 mostrarNaves(misNaves);
53
54 System.out.println("\nOrdenamiento burbuja por nombres:");
55 ordenarPorNombreBurbuja(misNaves);
56 mostrarNaves(misNaves);
57
58 // Mostrar los datos de la nave con dicho nombre, mensaje de "no encontrado" en caso contrario
59 System.out.println("Ingrese un nombre para buscar:");
60 nomb = sc.next();
61 pos = busquedaBinariaNombre(misNaves, nomb); // Asumiendo 'nombre' debe ser 'nomb'
62 mostrarBusqueda(pos, misNaves);
63
64 System.out.println("\nOrdenamiento por seleccion por puntos:");
65 ordenarPorPuntosSeleccion(misNaves);
66 mostrarNaves(misNaves);
67
68 System.out.println("\nOrdenamiento por seleccion por nombre:");
69 ordenarPorNombreSeleccion(misNaves);
70 mostrarNaves(misNaves);
71
72 System.out.println("\nOrdenamiento por insercion por puntos:");
73 ordenarPorPuntosInsercion(misNaves);
74 mostrarNaves(misNaves);
75
76 System.out.println("\nOrdenamiento por insecion por nombres:");
77 ordenarPorNombreInsercion(misNaves);
78 mostrarNaves(misNaves);
79 }
80

```

```



81 // Método para mostrar todas las naves
82 public static void mostrarNaves(Nave[] flota) {
83     for(Nave n: flota)
84         System.out.println(n+"\n");
85 }
86
87 // Método para mostrar todas las naves de un nombre que se pide por teclado
88 public static void mostrarPorNombre(Nave[] flota) {
89     Scanner sc=new Scanner(System.in);
90     System.out.print("Ingrese el nombre de la flota: ");
91     String nomBuscado=sc.next();
92     for(Nave n: flota) {
93         if( n.getNombre().equals(nomBuscado) )
94             System.out.println(n+"\n");
95     }
96 }
97
98 // Método para mostrar todas las naves con un número de puntos inferior o igual
99 // al número de puntos que se pide por teclado
100 public static void mostrarPorPuntos(Nave[] flota) {
101     Scanner sc=new Scanner(System.in);
102     System.out.print("Ingrese un numero de puntos: ");
103     int cantPunt=sc.nextInt();
104     for(Nave n: flota)
105         if(n.getPuntos()<=cantPunt)
106             System.out.println(n+"\n");
107 }

```

```
108
109 // Método que devuelve la Nave con mayor número de Puntos
110● public static Nave mostrarMayorPuntos(Nave[] flota) {
111     int pos=0;
112     int mayorPunt=flota[pos].getPuntos();
113     for(int i=1; i<flota.length; i++) {
114         if(mayorPunt<flota[i].getPuntos()) {
115             mayorPunt=flota[i].getPuntos();
116             pos=i;
117         }
118     }
119     return flota[pos];
120 }
121
122 // Método para buscar la primera nave con un nombre que se pidió por teclado
123● public static int busquedaLinealNombre(Nave[] flota, String s) {
124     for(int i=0; i<flota.length; i++) {
125         if( flota[i].getNombre().equals(s) )
126             return i;
127     }
128     return -1; // Placeholder
129 }
130
131 //Método que muestra la nave buscada
132● public static void mostrarBusqueda(int pos, Nave[] n) {
133     if(pos == -1) {
134         System.out.println("No existe esa nave");
135     } else {
136         System.out.println("La nave es: "+ n[pos]);
137     }
138 }
139
140 // Método que ordena por número de puntos de menor a mayor
141● public static void ordenarPorPuntosBurbuja(Nave[] flota) {
142     for(int i=1; i<flota.length; i++) {
143         for(int j=0; j<flota.length-i; j++){
144             if(flota[j].getPuntos() > flota[j+1].getPuntos())
145                 intercambiar(flota, j, j+1);
146         }
147     }
148 }
```

```
149 //Método que intercambia las posiciones de las naves
150 public static void intercambiar(Nave[] f, int pos1, int pos2) {
151     Nave nave = f[pos2];
152     f[pos2] = f[pos1];
153     f[pos1] = nave;
154 }
155
156 // Método que ordena por nombre de A a Z
157 public static void ordenarPorNombreBurbuja(Nave[] flota) {
158     for(int i = 1; i < flota.length; i++) {
159         for(int j = 0; j < flota.length-i; j++){
160             if(flota[j].getNombre().charAt(0) > flota[j+1].getNombre().charAt(0))
161                 intercambiar(flota, j, j+1);
162         }
163     }
164 }
165
166 // Método para buscar la primera nave con un nombre que se pidió por teclado
167 public static int busquedaBinariaNombre(Nave[] flota, String s) {
168     int alta = flota.length-1, baja = 0, media;
169     while(baja <= alta) {
170         media = (alta+baja)/2;
171         if( flota[media].getNombre().equals(s) )
172             return media;
173         else if( s.charAt(0) < flota[media].getNombre().charAt(0) )
174             alta = media-1;
175         else
176             baja = media+1;
177     }
178     return -1; // Placeholder
179 }
180
181 // Método que ordena por número de puntos de menor a mayor
182 public static void ordenarPorPuntosSeleccion(Nave[] flota) {
183     for(int rango=0; rango<flota.length-1; rango++) {
184         int pos=puntosMenor(flota,rango);
185         intercambiar(flota, pos, rango);
186     }
187 }
188
189 //Método que regresa al menor por puntos
190 public static int puntosMenor(Nave[] f, int r) {
191     int pos=r, menor=f[r].getPuntos();
192     for(int i=r+1; i<f.length; i++) {
193         if(menor>f[i].getPuntos()) {
194             menor = f[i].getPuntos();
195             pos=i;
196         }
197     }
198     return pos;
199 }
```

```
200
201 // Método que regresa el menor por nombre de A a Z
202 public static void ordenarPorNombreSeleccion(Nave[] flota) {
203     for(int rango=0; rango<flota.length-1; rango++) {
204         int pos=nombreMenor(flota,rango);
205         intercambiar(flota, pos, rango);
206     }
207 }
208
209 // Método que haya la nave por el nombre de A a Z
210 public static int nombreMenor(Nave[] f, int r) {
211     int pos=r;
212     char menor=f[r].getNombre().charAt(0);
213     for(int i=r+1; i<f.length; i++) {
214         if(menor>f[i].getNombre().charAt(0)) {
215             menor = f[i].getNombre().charAt(0);
216             pos=i;
217         }
218     }
219     return pos;
220 }
221
222 // Método que muestra las naves ordenadas por número de puntos de mayor a menor
223 public static void ordenarPorPuntosInsercion(Nave[] flota) {
224     for(int i=1; i<flota.length; i++) {
225         for(int j=i; j>=1; j--) {
226             if(flota[j-1].getPuntos() < flota[j].getPuntos())
227                 intercambiar(flota, j, j-1);
228         }
229     }
230 }
231
232 // Método que muestra las naves ordenadas por nombre de Z a A
233 public static void ordenarPorNombreInsercion(Nave[] flota) {
234     for(int i=1; i<flota.length; i++) {
235         for(int j=i; j>=1; j--) {
236             if(flota[j-1].getNombre().charAt(0) < flota[j].getNombre().charAt(0))
237                 intercambiar(flota, j, j-1);
238         }
239     }
240 }
241 }
```

	<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: 7</p>



CLASE NAVE:

Esta clase de encarga de darle los diferentes atributos a las naves y a la vez llamarlos por medio de métodos.



```

1 package Laboratorio4;
2 /*Laboratorio Nr4 - Ejercicio1
3  *Autor: Hilacondo Begazo,Emanuel David
4  *Colaboró: con nadie
5  *Tiempo: --
6  */
7
8 public class Nave {
9     private String nombre;
10    private int fila;
11    private String columna;
12    private boolean estado;
13    private int puntos;
14
15    // Métodos mutadores
16    public void setNombre(String n) {
17        nombre = n;
18    }
19
20    public void setFila(int f) {
21        fila = f;
22    }
23
24    public void setColumna(String c) {
25        columna = c;
26    }
27
28    public void setEstado(boolean e) {
29        estado = e;
30    }
31
32    public void setPuntos(int p) {
33        puntos = p;
34    }
35
36    // Métodos accesoros
37    public String getNombre() {
38        return nombre;
39    }
40
41    public int getFila() {
42        return fila;
43    }
44

```

	<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: 8</p>

<pre> 44 45 public String getColumna() { 46 return columna; 47 } 48 49 public boolean getEstado() { 50 return estado; 51 } 52 53 public int getPuntos() { 54 return puntos; 55 } 56 57 //Imprimir la nave 58 public String toString() { 59 return ("\nNombre: "+getNombre()+"\nFila: "+getFila()+"\nColumna: " + 60 getColumna()+"\nEstado: "+getEstado()+"\nPuntos: "+getPuntos()); 61 } 62 } </pre>	<h2>II. PRUEBAS</h2> <p><i>¿Con que valores comprobaste que tu práctica estuviera correcta?</i></p> <p>Se logró gracias a los diferentes testeos realizados al código:</p> <ul style="list-style-type: none"> • Ingreso correcto de “String”, “int”, “boolean” en las variables establecidas. • Correcto funcionamiento de cada método del ejercicio. • Imprimió los resultados esperados de acuerdo al problema. <p><i>¿Qué resultado esperabas obtener para cada valor de entrada?</i></p> <p>Esperaba que cada elemento del arreglo se almacenará con sus respectivos atributos para que se pueda desarrollar de manera correcta los diferentes métodos de búsqueda y ordenamiento.</p> <p><i>¿Qué valor o comportamiento obtuviste para cada valor de entrada?</i></p> <p>Se almacenó de manera correcta los datos de entrada para cada nave, los métodos funcionaron de manera correcta y se imprimió el resultado de los problemas propuestos.</p>
--	---

	<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>

PRUEBA DEL EJERCICIO 1: DEMO BATALLA

1. Ingresamos los atributos de las 8 naves.

```
Nave 1
Nombre: A1
Fila: 1
Columna: A
Estado: true
Puntos: 100
```

```
Nave 2
Nombre: B1
Fila: 6
Columna: X
Estado: true
Puntos: 12
```

```
Nave 3
Nombre: C1
Fila: 9
Columna: E
Estado: false
Puntos: 0
```

```
Nave 4
Nombre: C1
Fila: 4
Columna: T
Estado: true
Puntos: 35
```

```
Nave 5
Nombre: T1
Fila: 10
Columna: D
Estado: true
Puntos: 29
```

```
Nave 6
Nombre: M1
Fila: 8
Columna: B
Estado: true
Puntos: 75
```

- Después de ingresar los datos, se imprime todas las naves para mostrar el primer orden establecido, el de ingreso.

Naves creadas:

Nombre: A1
Fila: 1
Columna: A
Estado: true
Puntos: 100

Nombre: B1
Fila: 6
Columna: X
Estado: true
Puntos: 12

Nombre: C1
Fila: 9
Columna: E
Estado: false
Puntos: 0

Nombre: C1
Fila: 4
Columna: T
Estado: true
Puntos: 35

Nombre: T1
Fila: 10
Columna: D
Estado: true
Puntos: 29

Nombre: M1
Fila: 8
Columna: B
Estado: true
Puntos: 75

3. Imprime las naves que posean ese nombre.

```

Ingrese el nombre de la flota: C1

Nombre: C1
Fila: 9
Columna: E
Estado: false
Puntos: 0

Nombre: C1
Fila: 4
Columna: T
Estado: true
Puntos: 35

```

4. Imprime las naves que tengan esa cantidad de puntos o menor.

```

Ingrese un numero de puntos: 50

Nombre: B1
Fila: 6
Columna: X
Estado: true
Puntos: 12

Nombre: C1
Fila: 9
Columna: E
Estado: false
Puntos: 0

Nombre: C1
Fila: 4
Columna: T
Estado: true
Puntos: 35

Nombre: T1
Fila: 10
Columna: D
Estado: true
Puntos: 29

Nombre: M1
Fila: 20
Columna: Z
Estado: false
Puntos: 1

```

5. Imprime la nave que tenga más puntos.

```
Nave con mayor número de puntos:
Nombre: A1
Fila: 1
Columna: A
Estado: true
Puntos: 100
```

6. Imprime la primera nave que posea ese nombre usando búsqueda lineal.

```
Nave con mayor número de puntos:
Nombre: A1
Fila: 1
Columna: A
Estado: true
Puntos: 100
```

7. Ordena el arreglo de menor a mayor de acuerdo a la cantidad de puntos usando el método burbuja y se imprime.

Ordenamiento burbuja por puntos:



```
Nombre: C1
Fila: 9
Columna: E
Estado: false
Puntos: 0
```

```
Nombre: M1
Fila: 20
Columna: Z
Estado: false
Puntos: 1
```

```
Nombre: B1
Fila: 6
Columna: X
Estado: true
Puntos: 12
```

```
Nombre: T1
Fila: 10
Columna: D
Estado: true
Puntos: 29
```

```
Nombre: C1
Fila: 4
Columna: T
Estado: true
Puntos: 35
```

	<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: 13</p>

8. Ordenamiento Burbuja, pero por nombres de A a Z.

```
Ordenamiento burbuja por nombres:

Nombre: A1
Fila: 1
Columna: A
Estado: true
Puntos: 100

Nombre: B1
Fila: 6
Columna: X
Estado: true
Puntos: 12

Nombre: C1
Fila: 9
Columna: E
Estado: false
Puntos: 0



Nombre: C1
Fila: 4
Columna: T
Estado: true
Puntos: 35

Nombre: M1
Fila: 20
Columna: Z
Estado: false
Puntos: 1

Nombre: M1
Fila: 8
Columna: B
```

9. Imprime el primero que tenga ese nombre por búsqueda binaria:

```
Ingrese un nombre para buscar:
A1
La nave es:
Nombre: A1
Fila: 1
Columna: A
Estado: true
Puntos: 100
```

	<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: 14</p>

10. Usa el ordenamiento por selección de acuerdo a la cantidad de puntos, menor a mayor.

Ordenamiento por seleccion por puntos:

Nombre: C1
Fila: 9
Columna: E
Estado: false
Puntos: 0

Nombre: M1
Fila: 20
Columna: Z
Estado: false
Puntos: 1

Nombre: B1
Fila: 6
Columna: X
Estado: true
Puntos: 12

Nombre: T1
Fila: 10
Columna: D
Estado: true
Puntos: 29

Nombre: C1
Fila: 4
Columna: T
Estado: true
Puntos: 35



Nombre: V1
Fila: 6
Columna: P

11. Ordenamiento por selección usando nombres, de A a Z.

Ordenamiento por seleccion por nombre:

Nombre: A1
Fila: 1
Columna: A
Estado: true
Puntos: 100

Nombre: B1
Fila: 6
Columna: X
Estado: true
Puntos: 12

	<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: 15</p>

```
Nombre: C1
Fila: 4
Columna: T
Estado: true
Puntos: 35
```

```
Nombre: C1
Fila: 9
Columna: E
Estado: false
Puntos: 0
```

```
Nombre: M1
Fila: 20
Columna: Z
Estado: false
Puntos: 1
```

```
Nombre: M1
Fila: 8
Columna: B
Estado: true
Puntos: 75
```

```
Nombre: T1
Fila: 10
Columna: D
Estado: true
Puntos: 29
```



```
Nombre: V1
Fila: 6
Columna: P
Estado: true
Puntos: 61
```

12. Ordenamiento por inserción usando puntos de mayor a menor.

Ordenamiento por insercion por puntos:

```
Nombre: A1
Fila: 1
Columna: A
Estado: true
Puntos: 100
```

```
Nombre: M1
Fila: 8
Columna: B
Estado: true
Puntos: 75
```

	<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: 16</p>

```
Nombre: V1
Fila: 6
Columna: P
Estado: true
Puntos: 61
```

```
Nombre: C1
Fila: 4
Columna: T
Estado: true
Puntos: 35
```

```
Nombre: T1
Fila: 10
Columna: D
Estado: true
Puntos: 29
```

```
Nombre: B1
Fila: 6
Columna: X
Estado: true
Puntos: 12
```

```
Nombre: M1
Fila: 20
Columna: Z
Estado: false
Puntos: 1
```



```
Nombre: C1
Fila: 9
Columna: E
Estado: false
Puntos: 0
```

13. Ordenamiento por inserción, pero con nombres de Z a A.

```
Ordenamiento por insercion por nombres:
```

```
Nombre: V1
Fila: 6
Columna: P
Estado: true
Puntos: 61
```

```
Nombre: T1
Fila: 10
Columna: D
Estado: true
Puntos: 29
```


	<p align="center">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p align="center">Formato: 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: 17</p>

```
Nombre: M1
Fila: 8
Columna: B
Estado: true
Puntos: 75
```

```
Nombre: M1
Fila: 20
Columna: Z
Estado: false
Puntos: 1
```

```
Nombre: C1
Fila: 4
Columna: T
Estado: true
Puntos: 35
```

```
Nombre: C1
Fila: 9
Columna: E
Estado: false
Puntos: 0
```

```
Nombre: B1
Fila: 6
Columna: X
Estado: true
Puntos: 12
```

```
Nombre: A1
Fila: 1
Columna: A
Estado: true
Puntos: 100
```

COMMITTS:

CANTIDAD TOTAL DE COMMITS EN GITHUB:





Calihn1 last commit of my code

08ae37b · 3 hours ago



17 Commits

	<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: 18</p>

COMMITTS MÁS IMPORTANTES Y FECHAS DESDE GIT:

Primer avance realizado en clase de Laboratorio, se estableció una rama llamada Laboratorio4 para subir las diferentes versiones. Complete los primeros 5 métodos

```
commit a6b6f27040eaf5bcbd2fdd5067a09d94aaf447f7
Author: Calihn1 <ehilacondob@unsa.edu.pe>
Date: Fri Oct 11 18:28:16 2024 -0500
.
```

Segundo avance realizado del código. Complete 2 métodos más.

```
commit 544c92d3d8848466db933a2ea1a9eda8559e2562
Author: Calihn1 <ehilacondob@unsa.edu.pe>
Date: Fri Oct 11 23:16:49 2024 -0500

second commit de Nave

commit f7fc9b3bea14045fcf68d707da0330ba3b50ee11
Author: Calihn1 <ehilacondob@unsa.edu.pe>
Date: Fri Oct 11 23:16:31 2024 -0500



second commit de DemoBatalla
```

Commits on Oct 11, 2024		
second commit de Nave	544c92d	<>
Calihn1 committed 2 days ago		
second commit de DemoBatalla	f7fc9b3	<>
Calihn1 committed 2 days ago		
First commit de Nave	a6b6f27	<>
Calihn1 committed 2 days ago		
First commit de DemoBatalla	4c63263	<>
Calihn1 committed 2 days ago		

Tercer y último avance realizado del código. Complete los demás métodos y correcciones del código,

```
commit 08ae37bcc2f69f2fe6492b9d850a583a03f1f71d (HEAD -> Laboratorio4, origin/Laboratorio4)
Author: Calihn1 <ehilacondob@unsa.edu.pe>
Date: Sun Oct 13 14:06:05 2024 -0500

last commit of my code
```

	<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: 19

last commit of my code

Calihn1 committed 3 hours ago

08ae37b

<>

Laboratorio4

3 Branches

0 Tags

Go to file

Add file

<> Code

This branch is 11 commits ahead of, 6 commits behind main

Contribute

Calihn1 last commit of my code

08ae37b · 4 hours ago

17 Commits

Laboratorio2	Es mi codigo	2 weeks ago
Laboratorio3	Mi progreso	last week
Laboratorio4	last commit of my code	4 hours ago

Publicación de la versión final en el main de mi repositorio, tras las diferentes pruebas con resultado exitoso.

Commits on Oct 13, 2024

Merge branch 'main' of https://github.com/Calihn1/REPOSITORIO_FP2_LAB

Calihn1 committed 3 hours ago

2ff85d7

<>

main

3 Branches

0 Tags

Go to file

Add file

<> Code

Calihn1 Merge branch 'main' of https://github.com/Calihn1/REPOSITORIO_FP2_LAB

2ff85d7 · 3 hours ago

12 Commits

Laboratorio2	Es mi codigo	2 weeks ago
Laboratorio3	Mi progreso	last week
Laboratorio4	Version oficial LAB4	3 hours ago

LINK A LA RAMA LABORATORIO4:

https://github.com/Calihn1/REPOSITORIO_FP2_LAB/tree/Laboratorio4

III. RÚBRICA:

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	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 evolución de la madurez del código fuente, explicaciones puntuales pero precisas y un acabado	4	XX	2	

	impecable. (El profesor puede preguntar para refrendar calificación).				
TOTAL		20		18	

Colocar la evidencia de las respuestas realizadas al cuestionario enunciado en la guía práctica de laboratorio.
 Tabla 2: Rúbrica para contenido del Informe y demostración

CONCLUSIONES

Colocar las conclusiones, apreciaciones reflexivas, opiniones finales a cerca de los resultados obtenidos de la sesión de laboratorio.

CONCLUSIÓN:

Me ayudó a comprender de mejor manera como trabajar los arreglos de objetos, al ser un código con bastantes líneas fue un reto a superar, pero gracias a ello mejore al momento de probar cada método y desarrollar el código en diferentes fragmentos.
 Además de ello aprendí diferentes métodos de ordenamientos y búsqueda que facilita a la máquina en el proceso.

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.

PASOS:

- Primero entender el problema que plantea el ejercicio.
- Diseñar el algoritmo más corto y eficiente para solucionar el problema que tenían algunos métodos.
- Escribir el código en el lenguaje de programación java para ejecutar el algoritmo.
- Recibir los valores esperados del código realizado.

REFERENCIAS Y BIBLIOGRAFÍA

Colocare las referencias utilizadas para el desarrollo de la práctica en formato IEEE

M. Aedo López, Práctica de Laboratorio 4: Arreglos de Objetos, búsqueda y ordenamientos, Universidad Nacional de San Agustín, 2023