


	<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: 1</p>

INFORME DE LABORATORIO

INFORMACIÓN BÁSICA					
ASIGNATURA:	<i>Fundamentos de la Programación 02</i>				
TÍTULO DE LA PRÁCTICA:					
NÚMERO DE PRÁCTICA:	<i>04</i>	AÑO LECTIVO:	<i>2024</i>	NRO. SEMESTRE:	<i>2</i>
FECHA DE PRESENTACIÓN	<i>13/10/2024</i>	HORA DE PRESENTACIÓN	<i>12/30/00 PM</i>		
INTEGRANTE (s) <i>German Arturo Chipana Jerónimo</i>				NOTA (0-20)	
DOCENTE(s): <i>Pinto Oppe Lino José</i>					

RESULTADOS Y PRUEBAS	
I. EJERCICIOS RESUELTOS: <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> <i>El docente debe colocar la retroalimentación por cada ejercicio que el estudiante/grupo ha presentado</i>	
EJERCICIO 01:	
DEMO BATALLA	
1. Usted podrá reutilizar las dos clases Nave.java y DemoBatalla.java. creadas en Laboratorio 3. 2. Completar el Código de la clase DemoBatalla.	
<u>MAIN:</u>	

```
Start Page x Laboratorio_04.java x Nave.java x
Source History
1  /*
2     Autor: Chipana Jeronimo German Arturo
3     Proposito: Laboratorio 04
4  */
5  package laboratorio_04;
6
7  import java.util.*;
8  public class Laboratorio_04 {
9
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("Nave " + (i+1));
19             System.out.print("Nombre: ");
20             nomb = sc.next();
21             System.out.println("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             misNaves[i] = new Nave(); //Se crea un objeto Nave y se asigna su referencia a misNaves
30             misNaves[i].setNombre(nomb);
31             misNaves[i].setFila(fil);
32             misNaves[i].setColumna(col);
33             misNaves[i].setEstado(est);
34             misNaves[i].setPuntos(punt);
35         }
36         System.out.println("\nNaves creadas:");
37         mostrarNaves(misNaves);
38         mostrarPorNombre(misNaves);
39         mostrarPorPuntos(misNaves);
40         System.out.println("\nNave con mayor número de puntos: " + mostrarMayorPuntos(misNaves));
41         //leer un nombre
42         System.out.println("Ingrese nombre de la nave a buscar: ");
43         String nombre=sc.next();
44         //mostrar los datos de la nave con dicho nombre, mensaje de "no encontrado" en caso contrario
45         int pos=busquedaLinealNombre(misNaves,nombre);
46         if(pos!=-1) {
47             // Si la nave es encontrada, mostrarla
48             System.out.println("Nave encontrada por Busqueda Lineal: "+misNaves[pos].toString());
49         }
50         else{
51             System.out.println("Nave no encontrada.");
52         }
53         System.out.println("Ordenando naves por puntos de menor a mayor por ORDENAMIENTO BURBUJA");
54         ordenarPorPuntosBurbuja(misNaves);
55         mostrarNaves(misNaves);
56     }
57 }
```

	<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: 3</p>

<div>55</div> <div>56</div> <div>57</div> <div>58</div> <div>59</div> <div>60</div> <div>61</div> <div>62</div> <div>63</div> <div>64</div> <div>65</div> <div>66</div> <div>67</div> <div>68</div> <div>69</div> <div>70</div> <div>71</div> <div>72</div> <div>73</div> <div>74</div> <div>75</div> <div>76</div> <div>77</div> <div>78</div> <div>79</div> <div>80</div> <div>81</div> <div>82</div> <div>83</div> <div>84</div> <div>85</div> <div>86</div> <div>87</div> <div>88</div> <div>89</div> <div>90</div> <div>91</div> <div>92</div> <div>93</div> <div>94</div> <div>95</div> <div>96</div> <div>97</div> <div>98</div> <div>99</div> <div>100</div> <div>101</div> <div>102</div>	<pre> mostrarNaves(misNaves); System.out.println("Ordenando naves por nombre de A a Z por ORDENAMIENTO BURBUJA"); ordenarPorNombreBurbuja(misNaves); mostrarNaves(misNaves); //mostrar los datos de la nave con dicho nombre, mensaje de "no encontrado" en caso contrario pos=busquedaBinariaNombre(misNaves,nombre); if(pos!=-1) { // Si la nave es encontrada, mostrarla System.out.println("Nave encontrada por Búsqueda Binaria: "+misNaves[pos].toString()); } else{ System.out.println("Nave no encontrada."); } System.out.println("Ordenando naves por puntos de menor a mayor por ORDENAMIENTO SELECCION"); ordenarPorPuntosSeleccion(misNaves); mostrarNaves(misNaves); System.out.println("Ordenando naves por nombre de A a Z por ORDENAMIENTO SELECCION"); ordenarPorNombreSeleccion(misNaves); mostrarNaves(misNaves); System.out.println("Ordenando naves por puntos de mayor a menor por ORDENAMIENTO INSERCIÓN"); ordenarPorPuntosInsercion(misNaves); mostrarNaves(misNaves); System.out.println("Ordenando naves por nombres de Z a A por ORDENAMIENTO INSERCIÓN"); ordenarPorNombreInsercion(misNaves); mostrarNaves(misNaves); } //Método para mostrar todas las naves public static void mostrarNaves(Nave[] flota){ //REUTILIZAR for(Nave nave : flota){ System.out.println(nave.toString()); } } //Método para mostrar todas las naves de un nombre que se pide por teclado public static void mostrarPorNombre(Nave[] flota){ //REUTILIZAR Scanner scan=new Scanner(System.in); System.out.println("Ingrese nombre de la nave a buscar: "); String naveBuscar=scan.next(); System.out.println("Naves encontradas con el nombre "+naveBuscar+" : "); for(Nave nave : flota){ if(nave.getNombre().equals(naveBuscar)){ System.out.println(nave.toString()); } } } </pre>
--	--

```
103 //Método para mostrar todas las naves con un número de puntos inferior o igual
104 //al número de puntos que se pide por teclado
105 public static void mostrarPorPuntos(Nave[] flota){
106     //REUTILIZAR
107     Scanner scan=new Scanner(System.in);
108     System.out.println("Ingrese el numero max de puntos: ");
109     int maxPuntos=scan.nextInt();
110     System.out.println("Naves con puntos inferiores a "+maxPuntos);
111     for(Nave nave : flota){
112         if(nave.getPuntos()<=maxPuntos){
113             System.out.println(nave.toString());
114         }
115     }
116 }
117
118 //Método que devuelve la Nave con mayor número de Puntos
119 public static Nave mostrarMayorPuntos(Nave[] flota){
120     //REUTILIZAR
121     Nave naveMaxPuntos=flota[0];
122     for(Nave nave : flota){
123         if(nave.getPuntos()>naveMaxPuntos.getPuntos()){
124             naveMaxPuntos=nave;
125         }
126     }
127     return naveMaxPuntos;
128 }
129
130 //Método para buscar la primera nave con un nombre que se pidió por teclado
131 public static int busquedaLinealNombre(Nave[] flota, String s){
132     for(int i=0;i<flota.length;i++){
133         if(flota[i].getNombre().equals(s)){
134             return i;
135         }
136     }
137     return -1;
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 }
150
```

```

151 //Método que ordena por nombre de A a Z
152 public static void ordenarPorNombreBurbuja(Nave[] flota){
153     for(int i=1;i<flota.length;i++){
154         for(int j=0;j<flota.length-i;j++){
155             // Comparar las letras iniciales de los nombres
156             char letraInicial1=flota[j].getNombre().charAt(0);
157             char letraInicial2=flota[j+1].getNombre().charAt(0);
158             if (letraInicial1>letraInicial2) {
159                 // Intercambiar las naves si están en el orden incorrecto
160                 intercambiar(flota,j,j+1);
161             }
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 baja=0;
169     int alta=flota.length-1;
170     while(baja<=alta) {
171         int media=(alta+baja)/2;
172         // Comparar usando compareTo
173         int comparacion = flota[media].getNombre().compareTo(s);
174         if (comparacion==0) {
175             return media; // Se encontró la nave
176         } else if (comparacion>0) {
177             alta=media-1; // Buscar en la mitad inferior
178         } else {
179             baja=media+1; // Buscar en la mitad superior
180         }
181     }
182     return -1; // No se encontró la nave
183 }
184
185 // Método que ordena por número de puntos de menor a mayor (Selección)
186 public static void ordenarPorPuntosSeleccion(Nave[] flota){
187     for(int i=0;i<flota.length-1;i++){
188         int indiceMinimo=i;
189         for(int j=i+1;j<flota.length;j++){
190             if(flota[j].getPuntos()<flota[indiceMinimo].getPuntos()){
191                 indiceMinimo=j; // Encontramos el nuevo mínimo
192             }
193         }
194         // Solo se intercambia una vez al final de la iteración
195         if (indiceMinimo!=i) {
196             intercambiar(flota,i,indiceMinimo);
197         }
198     }
199 }
200

```



```

201 //Método que ordena por nombre de A a Z
202 public static void ordenarPorNombreSeleccion(Nave[] flota){
203     for(int i=0;i<flota.length;i++) {
204         int indiceMinimo=i;
205         for(int j=i+1;j<flota.length;j++) {
206             char letraInicial1 = flota[j].getNombre().charAt(0);
207             char letraInicial2 = flota[indiceMinimo].getNombre().charAt(0);
208             if(letraInicial1<letraInicial2) {
209                 indiceMinimo = j; // Actualiza el índice mínimo
210             }
211         }
212         // Intercambiar el elemento en la posición i con el mínimo encontrado
213         if(indiceMinimo!=i) {
214             intercambiar(flota,i,indiceMinimo);
215         }
216     }
217 }
218
219 //Método que muestra las naves ordenadas por número de puntos de mayor a menor
220 public static void ordenarPorPuntosInsercion(Nave[] flota){
221     for(int i=1;i<flota.length;i++){
222         Nave clave=flota[i];
223         int j=i-1;
224         while(j>=0 && flota[j].getPuntos()<clave.getPuntos()){
225             flota[j+1]=flota[j];
226             j=j-1;
227         }
228         flota[j+1]=clave;
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         Nave clave=flota[i]; // Guardamos el objeto Nave
236         int j=i-1;
237         // Mover las naves que tienen un nombre menor que clave (para ordenar de Z a A)
238         while (j>=0 && flota[j].getNombre().compareTo(clave.getNombre())<0) {
239             flota[j+1]=flota[j];
240             j=j-1;
241         }
242         // Colocar la clave en su posición correcta
243         flota[j+1]=clave;
244     }
245 }
246
247 //Método que intercambia naves
248 public static void intercambiar(Nave[] flota,int i, int j){
249     Nave temp;
250     temp=flota[i];
251     flota[i]=flota[j];
252     flota[j]=temp;
253 }
254 }

```

CLASE NAVE:

```
Start Page x Laboratorio_04.java [-/A] x Nave.java [-/A] x
Source History
1  /*
2   * Clase Nave
3   */
4   package laboratorio_04;
5
6   public class Nave {
7       private String nombre;
8       private int fila;
9       private String columna;
10      private boolean estado;
11      private int puntos;
12      // Metodos mutadores
13      public void setNombre( String n){
14          nombre = n;
15      }
16      public void setFila(int f){
17          fila = f;
18      }
19      public void setColumna(String c){
20          columna = c;
21      }
22      public void setEstado(boolean e){
23          estado = e;
24      }
25      public void setPuntos(int p){
26          puntos = p;
27      }
28      // Metodos accesorios
29      public String getNombre(){
30          return nombre;
31      }
32      public int getFila(){
33          return fila;
34      }
35      public String getColumna(){
36          return columna;
37      }
38      public boolean getEstado(){
39          return estado;
40      }
41      public int getPuntos(){
42          return puntos;
43      }
44      // Completar con otros métodos necesarios
45      public String toString(){
46          return "Nave{" +
47              "nombre=" + nombre +
48              ", fila=" + fila +
49              ", columna=" + columna +
50              ", estado=" + estado +
51              ", puntos=" + puntos +
52              '}';
53      }
54  }
```

	<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>

II. PRUEBAS

¿Con que valores comprobaste que tu práctica estuviera correcta?

Para comprobar que mi práctica estaba correcta, utilicé los siguientes valores de entrada para las naves:

- Nave{nombre= Galactus, fila=1, columna=A, estado=true, puntos=10}
- Nave{nombre= Lacia, fila=2, columna=B, estado=false, puntos=20}
- Nave{nombre= Lactea, fila=5, columna=D, estado=true, puntos=35}
- Nave{nombre= Roro, fila=6, columna=G, estado=true, puntos=33}
- Nave{nombre= Tryce, fila=6, columna=C, estado=false, puntos=25}
- Nave{nombre= Feka, fila=7, columna=H, estado=true, puntos=61}
- Nave{nombre= Barca, fila=1, columna=E, estado=true, puntos=56}
- Nave{nombre= Vardrid, fila=3, columna=C, estado=false, puntos=23}

Siendo fila, puntos de tipo int; columna, nombre de tipo String; estado de tipo boolean.

¿Qué resultado esperabas obtener para cada valor de entrada?



Esperaba obtener los siguientes resultados para cada valor de entrada:

- Al mostrar todas las naves, deberían aparecer en el orden en que se introdujeron.
- Al buscar un nombre existente, como "Roro", esperaba encontrar y mostrar la información de esa nave. Para un nombre inexistente, como "NaveX", esperaba el mensaje "Nave no encontrada".
- Al introducir un número de puntos, como 30, esperaba que se mostraran las naves con puntos menores o iguales a este número.
- La nave con el mayor número de puntos debería ser "Feka" con 61 puntos.
- Al ordenar las naves por puntos, esperaba que aparecieran en orden ascendente.
- Al ordenar por nombre, esperaba que se mostraran de A a Z.

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

Los resultados obtenidos fueron consistentes con mis expectativas:

- Al mostrar todas las naves, aparecieron en el orden correcto.
- Al buscar "Roro", se encontró y mostró correctamente; al buscar "NaveX", se devolvió el mensaje adecuado.
- Al solicitar naves con puntos menores o iguales a 30, se mostraron correctamente.
- La nave con el mayor número de puntos, "Feka", fue correctamente identificada.
- Después de ordenar por puntos y por nombre, las naves se presentaron en el orden esperado.

	<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>

DEMO BATALLA

1.Se ingresaron datos de las naves.



```
Puntos: 56
Nave 8
Nombre: Vardrid
Fila
3
Columna: C
Estado: false
Puntos: 23
```

2.Se mostro a todas las naves creadas.

```
Naves creadas:
Nave{nombre= Galactus, fila=1, columna=A, estado=true, puntos=10}
Nave{nombre= Lacia, fila=2, columna=B, estado=false, puntos=20}
Nave{nombre= Lactea, fila=5, columna=D, estado=true, puntos=35}
Nave{nombre= Roro, fila=6, columna=G, estado=true, puntos=33}
Nave{nombre= Tryce, fila=6, columna=C, estado=false, puntos=25}
Nave{nombre= Feka, fila=7, columna=H, estado=true, puntos=61}
Nave{nombre= Barca, fila=1, columna=E, estado=true, puntos=56}
Nave{nombre= Vardrid, fila=3, columna=C, estado=false, puntos=23}
```

3.Se pidió el nombre de una nave y se mostraron sus datos.

```
Ingresa nombre de la nave a buscar:
Roro
Naves encontradas con el nombre Roro :
Nave{nombre= Roro, fila=6, columna=G, estado=true, puntos=33}
```

	<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 style="text-align: center;">Código: GUIA-PRLE-001</p>	<p style="text-align: right;">Página: 10</p>

4. Se pidió ingresar el número máximo de puntos para mostrar las naves con puntos inferiores.

Ingrese el numero max de puntos:

30

Naves con puntos inferiores a 30

Nave{nombre= Galactus, fila=1, columna=A, estado=true, puntos=10}

Nave{nombre= Lacia, fila=2, columna=B, estado=false, puntos=20}

Nave{nombre= Tryce, fila=6, columna=C, estado=false, puntos=25}

Nave{nombre= Vardrid, fila=3, columna=C, estado=false, puntos=23}

5. Se mostró los datos de la nave con mayores puntos.

Nave con mayor número de puntos: Nave{nombre= Feka, fila=7, columna=H, estado=true, puntos=61}

6. Se pidió el nombre de una nave a buscar y se mostro la nave por búsqueda lineal.

Ingrese nombre de la nave a buscar:

Feka

Nave encontrada por Búsqueda Lineal: Nave{nombre= Feka, fila=7, columna=H, estado=true, puntos=61}

7. Se ordenaron las naves por puntos de menor a mayor por ORDENAMIENTO BURBUJA.

Ordenando naves por puntos de menor a mayor por ORDENAMIENTO BURBUJA

Nave{nombre= Galactus, fila=1, columna=A, estado=true, puntos=10}

Nave{nombre= Lacia, fila=2, columna=B, estado=false, puntos=20}

Nave{nombre= Vardrid, fila=3, columna=C, estado=false, puntos=23}



Nave{nombre= Tryce, fila=6, columna=C, estado=false, puntos=25}

Nave{nombre= Roro, fila=6, columna=G, estado=true, puntos=33}

Nave{nombre= Lactea, fila=5, columna=D, estado=true, puntos=35}

Nave{nombre= Barca, fila=1, columna=E, estado=true, puntos=56}

Nave{nombre= Feka, fila=7, columna=H, estado=true, puntos=61}

	<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: 11</p>

8. Se ordenaron las naves por nombre de A a Z por ORDENAMIENTO BURBUJA.



```
Ordenando naves por nombre de A a Z por ORDENAMIENTO BURBUJA
Nave{nombre= Barca, fila=1, columna=E, estado=true, puntos=56}
Nave{nombre= Feka, fila=7, columna=H, estado=true, puntos=61}
Nave{nombre= Galactus, fila=1, columna=A, estado=true, puntos=10}
Nave{nombre= Lacia, fila=2, columna=B, estado=false, puntos=20}
Nave{nombre= Lactea, fila=5, columna=D, estado=true, puntos=35}
Nave{nombre= Roro, fila=6, columna=G, estado=true, puntos=33}
Nave{nombre= Tryce, fila=6, columna=C, estado=false, puntos=25}
Nave{nombre= Vardrid, fila=3, columna=C, estado=false, puntos=23}
```

9. Se mostro los datos de la nave que se pidió anteriormente, esta vez por búsqueda binaria.

```
Nave encontrada por Busqueda Binaria: Nave{nombre= Feka, fila=7, columna=H, estado=true, puntos=61}
```

10. Se ordenaron las naves por puntos de menor a mayor por ORDENAMIENTO SELECCIÓN.

```
Ordenando naves por puntos de menor a mayor por ORDENAMIENTO SELECCION
Nave{nombre= Galactus, fila=1, columna=A, estado=true, puntos=10}
Nave{nombre= Lacia, fila=2, columna=B, estado=false, puntos=20}
Nave{nombre= Vardrid, fila=3, columna=C, estado=false, puntos=23}
Nave{nombre= Tryce, fila=6, columna=C, estado=false, puntos=25}
Nave{nombre= Roro, fila=6, columna=G, estado=true, puntos=33}
Nave{nombre= Lactea, fila=5, columna=D, estado=true, puntos=35}
Nave{nombre= Barca, fila=1, columna=E, estado=true, puntos=56}
Nave{nombre= Feka, fila=7, columna=H, estado=true, puntos=61}
```



	<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 style="text-align: center;">Código: GUIA-PRLE-001</p>	<p style="text-align: right;">Página: 12</p>

11. Se ordenaron las naves por nombres de A a Z por ORDENAMIENTO SELECCIÓN.

```
Ordenando naves por nombre de A a Z por ORDENAMIENTO SELECCION
Nave{nombre= Barca, fila=1, columna=E, estado=true, puntos=56}
Nave{nombre= Feka, fila=7, columna=H, estado=true, puntos=61}
Nave{nombre= Galactus, fila=1, columna=A, estado=true, puntos=10}
Nave{nombre= Lactea, fila=5, columna=D, estado=true, puntos=35}
Nave{nombre= Lacia, fila=2, columna=B, estado=false, puntos=20}
Nave{nombre= Roro, fila=6, columna=G, estado=true, puntos=33}
Nave{nombre= Tryce, fila=6, columna=C, estado=false, puntos=25}
Nave{nombre= Vardrid, fila=3, columna=C, estado=false, puntos=23}
```

12. Se ordenaron las naves por puntos de mayor a menor por ORDENAMIENTO INSERCIÓN.

```
Ordenando naves por puntos de mayor a menor por ORDENAMIENTO INSERCCION
Nave{nombre= Feka, fila=7, columna=H, estado=true, puntos=61}
Nave{nombre= Barca, fila=1, columna=E, estado=true, puntos=56}
Nave{nombre= Lactea, fila=5, columna=D, estado=true, puntos=35}
Nave{nombre= Roro, fila=6, columna=G, estado=true, puntos=33}
Nave{nombre= Tryce, fila=6, columna=C, estado=false, puntos=25}
Nave{nombre= Vardrid, fila=3, columna=C, estado=false, puntos=23}
Nave{nombre= Lacia, fila=2, columna=B, estado=false, puntos=20}
Nave{nombre= Galactus, fila=1, columna=A, estado=true, puntos=10}
```

	<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>

13. Se ordenaron las naves por nombres de Z a A por ORDENAMIENTO INSERCIÓN.

```
Ordenando naves por nombres de Z a A por ORDENAMIENTO INSERCIÓN
Nave{nombre= Vardrid, fila=3, columna=C, estado=false, puntos=23}
Nave{nombre= Tryce, fila=6, columna=C, estado=false, puntos=25}
Nave{nombre= Roro, fila=6, columna=G, estado=true, puntos=33}
Nave{nombre= Lactea, fila=5, columna=D, estado=true, puntos=35}
Nave{nombre= Lacia, fila=2, columna=B, estado=false, puntos=20}
Nave{nombre= Galactus, fila=1, columna=A, estado=true, puntos=10}
Nave{nombre= Feka, fila=7, columna=H, estado=true, puntos=61}
Nave{nombre= Barca, fila=1, columna=E, estado=true, puntos=56}
BUILD SUCCESSFUL (total time: 4 minutes 16 seconds)
```

MIS COMMITS:

PRIMERA VERSION:

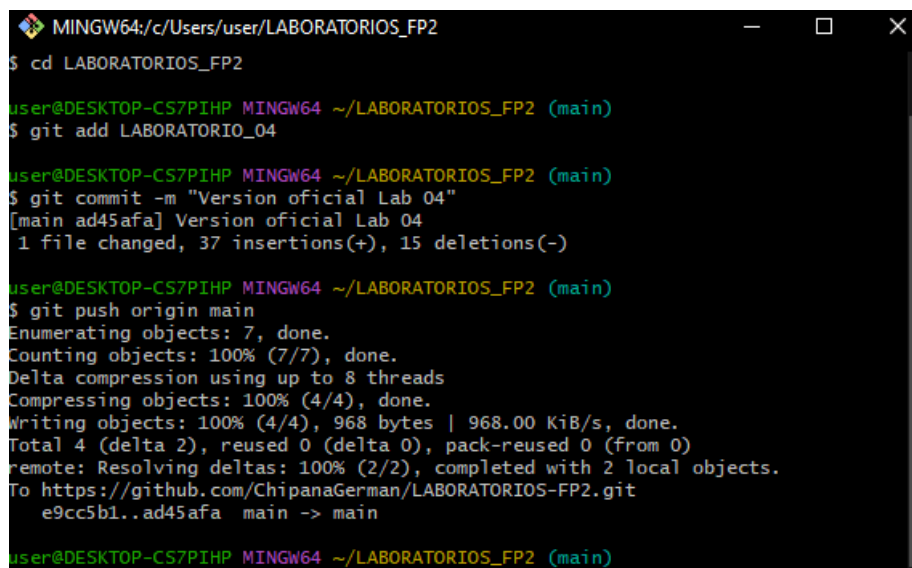
```
user@DESKTOP-CS7PIHP MINGW64 ~/LABORATORIOS_FP2 (main)
$ git add LABORATORIO_04

user@DESKTOP-CS7PIHP MINGW64 ~/LABORATORIOS_FP2 (main)
$ git commit -m "Primera Version Lab 04"
[main e9cc5b1] Primera Version Lab 04
2 files changed, 286 insertions(+)
create mode 100644 LABORATORIO_04/Laboratorio_04.java
create mode 100644 LABORATORIO_04/Nave.java

user@DESKTOP-CS7PIHP MINGW64 ~/LABORATORIOS_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), 2.64 KiB | 2.64 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/ChipanaGerman/LABORATORIOS-FP2.git
   af6b7eb..e9cc5b1  main -> main
```

Se agregaron los archivos Laboratorio_04.java y Nave.java al repositorio de GitHub.

SEGUNDA VERSION:



```

MINGW64:/c:/Users/user/LABORATORIOS_FP2
$ cd LABORATORIOS_FP2

user@DESKTOP-CS7PIHP MINGW64 ~/LABORATORIOS_FP2 (main)
$ git add LABORATORIO_04

user@DESKTOP-CS7PIHP MINGW64 ~/LABORATORIOS_FP2 (main)
$ git commit -m "Version oficial Lab 04"
[main ad45afa] Version oficial Lab 04
1 file changed, 37 insertions(+), 15 deletions(-)

user@DESKTOP-CS7PIHP MINGW64 ~/LABORATORIOS_FP2 (main)
$ git push origin main
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 8 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 968 bytes | 968.00 KiB/s, done.
Total 4 (delta 2), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/ChipanaGerman/LABORATORIOS-FP2.git
   e9cc5b1..ad45afa  main -> main

```

Se actualizaron los archivos Laboratorio_04.java y Nave.java y se subieron a GitHub.

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	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	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	4	
TOTAL		20		18	



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

CONCLUSIONES

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

CONCLUSIÓN:

En esta práctica de laboratorio, se implementaron y comprendieron los conceptos de arreglos de objetos, búsquedas y ordenamientos en Java. Se logró manipular datos de manera eficiente, aplicando diferentes algoritmos de búsqueda y ordenamiento (como búsqueda lineal y binaria, ordenamientos por burbuja, selección e inserción). Este ejercicio permitió reforzar el manejo de estructuras de datos y mejorar las habilidades en programación orientada a objetos.

	<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>

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.

- Se analizaron los ejercicios propuestos y se comprendió la lógica del manejo de arreglos de objetos.
- Se implementaron los métodos necesarios para realizar las búsquedas y ordenamientos solicitados.
- Se realizaron pruebas con distintos valores de entrada para verificar el correcto funcionamiento del código.
- Se depuraron los errores encontrados durante las pruebas.
- Se documentó el código y se subieron las versiones a GitHub.

REFERENCIAS Y BIBLIOGRAFÍA

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

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

<https://github.com/ChipanaGerman/LABORATORIOS-FP2>