



Clase Nave:

```
1 package tarea06;
2 public class Nave {
3     private String nombre;
4     private int fila;
5     private String columna;
6     private boolean estado;
7     private int puntos;
8
9     // Métodos mutadores
10    public void setNombre(String n) {
11        nombre = n;
12    }
13    public void setFila(int f) {
14        fila = f;
15    }
16    public void setColumna(String c) {
17        columna = c;
18    }
19    public void setEstado(boolean e) {
20        estado = e;
21    }
22    public void setPuntos(int p) {
23        puntos = p;
24    }
25
26    // Métodos accesorios
27    public String getNombre() {
28        return nombre;
29    }
30    public int getFila() {
31        return fila;
32    }
33    public String getColumna() {
34        return columna;
35    }
36    public boolean getEstado() {
37        return estado;
38    }
39    public int getPuntos() {
40        return puntos;
41    }
42 }
```



	<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 style="text-align: center;">Aprobación: 2022/03/01</p>	<p style="text-align: center;">Código: GUIA-PRLE-001</p>	<p style="text-align: right;">Página: 3</p>

Clase Main:

```

1 package LABORATORIO_04;
2 import java.util.*;
3 public class DemoBatalla {
4     public static void main(String[] args) {
5         Nave[] misNaves = new Nave[8];
6         Scanner sc = new Scanner(System.in);
7         String nomb, col;
8         int fil, punt;
9         boolean est;
10
11         for (int i = 0; i < misNaves.length; i++) {
12             System.out.println("Nave " + (i + 1));
13             System.out.print("Nombre: ");
14             nomb = sc.nextLine();
15             System.out.print("Fila: ");
16             fil = sc.nextInt();
17             sc.nextLine();
18             System.out.print("Columna: ");
19             col = sc.nextLine();
20             System.out.print("Estado (true/false): ");
21             est = sc.nextBoolean();
22             System.out.print("Puntos: ");
23             punt = sc.nextInt();
24             sc.nextLine();
25
26             misNaves[i] = new Nave();
27
28             misNaves[i].setNombre(nomb);
29             misNaves[i].setFila(fil);
30             misNaves[i].setColumna(col);
31             misNaves[i].setEstado(est);
32             misNaves[i].setPuntos(punt);
33         }
34
35         System.out.println("\nNaves creadas:");
36         mostrarNaves(misNaves);
37         mostrarPorNombre(misNaves);
38         mostrarPorPuntos(misNaves);
39         System.out.println("\nNave con mayor número de puntos: " + mostrarMayorPuntos(misNaves).getNombre());
40
41         System.out.println("Ingrese el nombre que desea buscar(Usando busqueda Lineal): ");
42         String nombre = sc.nextLine();
43
44         int pos = busquedaLinealNombre(misNaves, nombre);
45
46         if (pos != -1) {
47             System.out.println("Nave encontrada en la posicion: "+pos );
48             System.out.print("Nombre: " + misNaves[pos].getNombre() + ", Fila: " + misNaves[pos].getFila() +
49                 ", Columna: " + misNaves[pos].getColumna() + ", Estado: " + misNaves[pos].getEstado() +
50                 ", Puntos: " + misNaves[pos].getPuntos());
51         } else {
52             System.out.println("Nave no encontrada.");
53         }
54         System.out.println("=====");
55
56         System.out.println("Ordenamiento burbuja por puntos(menor a mayor):");
57         ordenarPorPuntosBurbuja(misNaves);
58         mostrarNaves(misNaves);
59         System.out.println("=====");
60
61         System.out.println("Ordenamiento burbuja por nombre(A a Z)");
62         ordenarPorNombreBurbuja(misNaves);
63         mostrarNaves(misNaves);
64         System.out.println("=====");
65

```



	<p style="text-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: 4

```

67      System.out.println("Ingrese el nombre que desea buscar(Usando busqueda Binaria): ");
68      nombre = sc.nextLine();
69      pos = busquedaBinariaNombre(misNaves, nombre);
70      if (pos != -1) {
71          System.out.println("Nave encontrada en la posicion: "+pos );
72          System.out.println("Nombre: " + misNaves[pos].getNombre() + ", Fila: " + misNaves[pos].getFila() +
73              ", Columna: " + misNaves[pos].getColumna() + ", Estado: " + misNaves[pos].getEstado() +
74              ", Puntos: " + misNaves[pos].getPuntos());
75      } else {
76          System.out.println("Nave no encontrada.");
77      }
78      System.out.println("=====");
79
80      System.out.println("Ordenamiento por seleccion según puntos(menor a mayor)");
81      ordenarPorPuntosSeleccion(misNaves);
82      mostrarNaves(misNaves);
83      System.out.println("=====");
84
85      System.out.println("Ordenamiento por seleccion según nombres(A a Z)");
86      ordenarPorNombreSeleccion(misNaves);
87      mostrarNaves(misNaves);
88      System.out.println("=====");
89
90      System.out.println("Ordenamiento por insercion según puntos(mayor a menor)");
91      ordenarPorPuntosInsercion(misNaves);
92      mostrarNaves(misNaves);
93      System.out.println("=====");
94
95      System.out.println("Ordenamiento por insercion según nombres(Z a A)");
96      ordenarPorNombreInsercion(misNaves);
97      mostrarNaves(misNaves);
98  }
99
100  // Método para mostrar todas las naves
101  public static void mostrarNaves(Nave[] flota) {
102      for (Nave nave : flota) {
103          System.out.println("Nombre: " + nave.getNombre() + ", Fila: " + nave.getFila() +
104              ", Columna: " + nave.getColumna() + ", Estado: " + nave.getEstado() +
105              ", Puntos: " + nave.getPuntos());
106      }
107  }
108
109  // Método para mostrar todas las naves por nombre solicitado
110  public static void mostrarPorNombre(Nave[] flota) {
111      Scanner sc = new Scanner(System.in);
112      System.out.print("\nIngrese el nombre de todas las naves a buscar: ");
113      String nombreBuscado = sc.next();
114
115      System.out.println("Naves con nombre '" + nombreBuscado + "':");
116      for (Nave nave : flota) {
117          if (nave.getNombre().equalsIgnoreCase(nombreBuscado)) {
118              System.out.println("Nombre: " + nave.getNombre() + ", Fila: " + nave.getFila() +
119                  ", Columna: " + nave.getColumna() + ", Estado: " + nave.getEstado() +
120                  ", Puntos: " + nave.getPuntos());
121          }
122      }
123  }
124
125  // Método para mostrar todas las naves con puntos menores o iguales al solicitado
126  public static void mostrarPorPuntos(Nave[] flota) {
127      Scanner sc = new Scanner(System.in);
128      System.out.print("\nIngrese el número máximo de puntos: ");
129      int puntosMaximos = sc.nextInt();
130
131      System.out.println("Naves con puntos menores o iguales a " + puntosMaximos + "):");
132      for (Nave nave : flota) {
133          if (nave.getPuntos() <= puntosMaximos) {
134              System.out.println("Nombre: " + nave.getNombre() + ", Fila: " + nave.getFila() +
135                  ", Columna: " + nave.getColumna() + ", Estado: " + nave.getEstado() +
136                  ", Puntos: " + nave.getPuntos());
137          }
138      }
139  }

```



```
141 // Método que devuelve la Nave con mayor número de puntos
142 public static Nave mostrarMayorPuntos(Nave[] flota) {
143     Nave naveMayorPuntos = flota[0];
144     for (Nave nave : flota) {
145         if (nave.getPuntos() > naveMayorPuntos.getPuntos()) {
146             naveMayorPuntos = nave;
147         }
148     }
149     return naveMayorPuntos;
150 }
151
152 // Método de búsqueda lineal por nombre
153 // Recorre cada nave en la flota, compara su nombre con el nombre buscado y
154 // retorna el índice si hay coincidencia.
155 public static int busquedaLinealNombre(Nave[] flota, String s) {
156     for (int i = 0; i < flota.length; i++) {
157         if (flota[i].getNombre().equalsIgnoreCase(s)) {
158             return i;
159         }
160     }
161     return -1; // No se encontró el nombre
162 }
163
164 // Método de búsqueda binaria por nombre
165 // Divide la flota en partes y compara el nombre del centro con el nombre
166 // buscado para reducir el rango de búsqueda.
167 public static int busquedaBinariaNombre(Nave[] flota, String s) {
168     int izquierda = 0, derecha = flota.length - 1;
169
170     while (izquierda <= derecha) {
171         int medio = (izquierda + derecha) / 2;
172         String nombreMedio = flota[medio].getNombre();
173
174         int comparacion = s.compareToIgnoreCase(nombreMedio);
175         if (comparacion == 0) {
176             return medio; // Nombre encontrado
177         } else if (comparacion < 0) {
178             derecha = medio - 1; // Busca en la mitad izquierda
179         } else {
180             izquierda = medio + 1; // Busca en la mitad derecha
181         }
182     }
183     return -1; // No se encontró el nombre
184 }
185
186 // Ordenar por puntos de menor a mayor (Burbuja)
187 // Compara naves adyacentes por puntos y las intercambia si están en orden
188 // incorrecto, repitiendo hasta ordenar.
189 public static void ordenarPorPuntosBurbuja(Nave[] flota) {
190     for (int i = 1; i < flota.length; i++) {
191         for (int j = 0; j < flota.length - i; j++) {
192             if (flota[j].getPuntos() > flota[j + 1].getPuntos()) {
193                 Nave temp = flota[j];
194                 flota[j] = flota[j + 1];
195                 flota[j + 1] = temp;
196             }
197         }
198     }
199 }
```

	<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>

```

201 // Ordenar por nombre de A a Z (Burbuja)
202 // Compara naves adyacentes por nombre y las intercambia si están en orden
203 // incorrecto, repitiendo hasta ordenar.
204 public static void ordenarPorNombreBurbuja(Nave[] flota) {
205     for (int i = 1; i < flota.length; i++) {
206         for (int j = 0; j < flota.length - i; j++) {
207             if (flota[j].getNombre().compareToIgnoreCase(flota[j + 1].getNombre()) > 0) {
208                 Nave temp = flota[j];
209                 flota[j] = flota[j + 1];
210                 flota[j + 1] = temp;
211             }
212         }
213     }
214 }
215
216 // Ordenar por puntos (Selección)
217 // Encuentra la nave con menos puntos en el rango restante y la coloca en su
218 // posición correcta.
219 public static void ordenarPorPuntosSeleccion(Nave[] flota) {
220     for (int i = 0; i < flota.length - 1; i++) {
221         int indiceMin = i;
222         for (int j = i + 1; j < flota.length; j++) {
223             if (flota[j].getPuntos() < flota[indiceMin].getPuntos()) {
224                 indiceMin = j;
225             }
226         }
227         if (indiceMin != i) {
228             Nave temp = flota[i];
229             flota[i] = flota[indiceMin];
230             flota[indiceMin] = temp;
231         }
232     }
233 }
234
235 // Ordenar por nombre (Selección)
236 // Encuentra el nombre alfabéticamente menor en el rango restante y lo coloca en
237 // su posición correcta.
238 public static void ordenarPorNombreSeleccion(Nave[] flota) {
239     for (int i = 0; i < flota.length - 1; i++) {
240         int indiceMin = i;
241         for (int j = i + 1; j < flota.length; j++) {
242             if (flota[j].getNombre().compareToIgnoreCase(flota[indiceMin].getNombre()) < 0) {
243                 indiceMin = j;
244             }
245         }
246         if (indiceMin != i) {
247             Nave temp = flota[i];
248             flota[i] = flota[indiceMin];
249             flota[indiceMin] = temp;
250         }
251     }
252 }
253
254 // Ordenar por puntos (Inserción)
255 // Inserta cada nave en la posición correcta dentro de la lista, comparando los
256 // puntos de mayor a menor.
257 public static void ordenarPorPuntosInsercion(Nave[] flota) {
258     for (int i = 1; i < flota.length; i++) {
259         Nave temp = flota[i];
260         int j = i - 1;
261         while (j >= 0 && flota[j].getPuntos() < temp.getPuntos()) {
262             flota[j + 1] = flota[j];
263             j--;
264         }
265         flota[j + 1] = temp;
266     }
267 }
268



```

	<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>

```

269 // Ordenar por nombre (Inserción)
270 // Inserta cada nave en la posición correcta dentro de la lista, comparando los
271 // nombres de Z a A.
272 public static void ordenarPorNombreInsercion(Nave[] flota) {
273     for (int i = 1; i < flota.length; i++) {
274         Nave temp = flota[i];
275         int j = i - 1;
276         while (j >= 0 && flota[j].getNombre().compareToIgnoreCase(temp.getNombre()) < 0) {
277             flota[j + 1] = flota[j];
278             j--;
279         }
280         flota[j + 1] = temp;
281     }
282 }
283 }

```

	<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: 8</p>



II. PRUEBAS

a)Ejecución ejercicio1:

Insertamos Datos:

Nave 1
Nombre: azul
Fila: 3
Columna: 7
Estado (true/false): true
Puntos: 34
Nave 2
Nombre: verde
Fila: 2
Columna: 7
Estado (true/false): false
Puntos: 67
Nave 3
Nombre: rosa
Fila: 4
Columna: 7
Estado (true/false): true
Puntos: 56
Nave 4
Nombre: azul
Fila: 3
Columna: 7
Estado (true/false): false
Puntos: 39

Nave 5
Nombre: marron
Fila: 2
Columna: 8
Estado (true/false): false
Puntos: 65
Nave 6
Nombre: plateado
Fila: 3
Columna: 7
Estado (true/false): false
Puntos: 58
Nave 7
Nombre: violeta
Fila: 4
Columna: 8
Estado (true/false): true
Puntos: 56
Nave 8
Nombre: negro
Fila: 5
Columna: 8
Estado (true/false): true
Puntos: 67

	<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: 9</p>

Ejecución:

Naves creadas:

Nombre: azul, Fila: 3, Columna: 7, Estado: true, Puntos: 34
Nombre: verde, Fila: 2, Columna: 7, Estado: false, Puntos: 67
Nombre: rosa, Fila: 4, Columna: 7, Estado: true, Puntos: 56
Nombre: azul, Fila: 3, Columna: 7, Estado: false, Puntos: 39
Nombre: marron, Fila: 2, Columna: 8, Estado: false, Puntos: 65
Nombre: plateado, Fila: 3, Columna: 7, Estado: false, Puntos: 58
Nombre: violeta, Fila: 4, Columna: 8, Estado: true, Puntos: 56
Nombre: negro, Fila: 5, Columna: 8, Estado: true, Puntos: 67

Ingrese el nombre de todas las naves a buscar: azul

Naves con nombre 'azul':

Nombre: azul, Fila: 3, Columna: 7, Estado: true, Puntos: 34
Nombre: azul, Fila: 3, Columna: 7, Estado: false, Puntos: 39

Ingrese el número máximo de puntos: 50

Naves con puntos menores o iguales a 50:

Nombre: azul, Fila: 3, Columna: 7, Estado: true, Puntos: 34
Nombre: azul, Fila: 3, Columna: 7, Estado: false, Puntos: 39

Nave con mayor número de puntos: verde

Ingrese el nombre que desea buscar(Usando busqueda Lineal):

azul



Nave encontrada en la posicion: 0

Nombre: azul, Fila: 3, Columna: 7, Estado: true, Puntos: 34

=====

Ordenamiento burbuja por puntos(menor a mayor):

Nombre: azul, Fila: 3, Columna: 7, Estado: true, Puntos: 34
Nombre: azul, Fila: 3, Columna: 7, Estado: false, Puntos: 39
Nombre: rosa, Fila: 4, Columna: 7, Estado: true, Puntos: 56
Nombre: violeta, Fila: 4, Columna: 8, Estado: true, Puntos: 56
Nombre: plateado, Fila: 3, Columna: 7, Estado: false, Puntos: 58
Nombre: marron, Fila: 2, Columna: 8, Estado: false, Puntos: 65
Nombre: verde, Fila: 2, Columna: 7, Estado: false, Puntos: 67
Nombre: negro, Fila: 5, Columna: 8, Estado: true, Puntos: 67
=====

	<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>

Ordenamiento burbuja por nombre(A a Z)

Nombre: azul, Fila: 3, Columna: 7, Estado: true, Puntos: 34
Nombre: azul, Fila: 3, Columna: 7, Estado: false, Puntos: 39
Nombre: marron, Fila: 2, Columna: 8, Estado: false, Puntos: 65
Nombre: negro, Fila: 5, Columna: 8, Estado: true, Puntos: 67
Nombre: plateado, Fila: 3, Columna: 7, Estado: false, Puntos: 58
Nombre: rosa, Fila: 4, Columna: 7, Estado: true, Puntos: 56
Nombre: verde, Fila: 2, Columna: 7, Estado: false, Puntos: 67
Nombre: violeta, Fila: 4, Columna: 8, Estado: true, Puntos: 56

=====

Ingrese el nombre que desea buscar(Usando busqueda Binaria):

blanco

Nave no encontrada.

=====



Ordenamiento por seleccion según puntos(menor a mayor)

Nombre: azul, Fila: 3, Columna: 7, Estado: true, Puntos: 34
Nombre: azul, Fila: 3, Columna: 7, Estado: false, Puntos: 39
Nombre: rosa, Fila: 4, Columna: 7, Estado: true, Puntos: 56
Nombre: violeta, Fila: 4, Columna: 8, Estado: true, Puntos: 56
Nombre: plateado, Fila: 3, Columna: 7, Estado: false, Puntos: 58
Nombre: marron, Fila: 2, Columna: 8, Estado: false, Puntos: 65
Nombre: verde, Fila: 2, Columna: 7, Estado: false, Puntos: 67
Nombre: negro, Fila: 5, Columna: 8, Estado: true, Puntos: 67

=====

Ordenamiento por seleccion según nombres(A a Z)

Nombre: azul, Fila: 3, Columna: 7, Estado: true, Puntos: 34
Nombre: azul, Fila: 3, Columna: 7, Estado: false, Puntos: 39
Nombre: marron, Fila: 2, Columna: 8, Estado: false, Puntos: 65
Nombre: negro, Fila: 5, Columna: 8, Estado: true, Puntos: 67
Nombre: plateado, Fila: 3, Columna: 7, Estado: false, Puntos: 58
Nombre: rosa, Fila: 4, Columna: 7, Estado: true, Puntos: 56
Nombre: verde, Fila: 2, Columna: 7, Estado: false, Puntos: 67
Nombre: violeta, Fila: 4, Columna: 8, Estado: true, Puntos: 56

	<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>

Ordenamiento por insercion según puntos(mayor a menor)

Nombre: negro, Fila: 5, Columna: 8, Estado: true, Puntos: 67

Nombre: verde, Fila: 2, Columna: 7, Estado: false, Puntos: 67

Nombre: marron, Fila: 2, Columna: 8, Estado: false, Puntos: 65

Nombre: plateado, Fila: 3, Columna: 7, Estado: false, Puntos: 58

Nombre: rosa, Fila: 4, Columna: 7, Estado: true, Puntos: 56

Nombre: violeta, Fila: 4, Columna: 8, Estado: true, Puntos: 56

Nombre: azul, Fila: 3, Columna: 7, Estado: false, Puntos: 39

Nombre: azul, Fila: 3, Columna: 7, Estado: true, Puntos: 34

=====

Ordenamiento por insercion según nombres(Z a A)

Nombre: violeta, Fila: 4, Columna: 8, Estado: true, Puntos: 56

Nombre: verde, Fila: 2, Columna: 7, Estado: false, Puntos: 67

Nombre: rosa, Fila: 4, Columna: 7, Estado: true, Puntos: 56



Nombre: plateado, Fila: 3, Columna: 7, Estado: false, Puntos: 58

Nombre: negro, Fila: 5, Columna: 8, Estado: true, Puntos: 67

Nombre: marron, Fila: 2, Columna: 8, Estado: false, Puntos: 65

Nombre: azul, Fila: 3, Columna: 7, Estado: false, Puntos: 39

Nombre: azul, Fila: 3, Columna: 7, Estado: true, Puntos: 34

	<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: 12</p>

Mi commit al repositorio:

```
Windows@DESKTOP-C32KACO MINGW64 ~/Desktop/RepositorioLocal (main)
$ git init
Reinitialized existing Git repository in C:/Users/Windows/Desktop/RepositorioLocal/.git/

Windows@DESKTOP-C32KACO MINGW64 ~/Desktop/RepositorioLocal (main)
$ git add LABORATORIO_04

Windows@DESKTOP-C32KACO MINGW64 ~/Desktop/RepositorioLocal (main)
$ git commit -m "Se sube el LABORATORIO_04"
On branch main
Your branch is ahead of 'origin/main' by 1 commit.
  (use "git push" to publish your local commits)



nothing to commit, working tree clean

Windows@DESKTOP-C32KACO MINGW64 ~/Desktop/RepositorioLocal (main)
$ git remote add origin https://github.com/JoseMorocco/FP2.git
error: remote origin already exists.

Windows@DESKTOP-C32KACO MINGW64 ~/Desktop/RepositorioLocal (main)
$ git push -u 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), 2.62 KiB | 2.62 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/JoseMorocco/FP2.git
   d660882..67ab558  main -> main
branch 'main' set up to track 'origin/main'.
```

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	1	
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		17	

	<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>

CONCLUSIONES

En este laboratorio, exploramos métodos de búsqueda y ordenamiento que son esenciales para optimizar la gestión de datos. Aprendimos a implementar búsqueda lineal y binaria para localizar elementos, así como técnicas de ordenamiento, incluyendo burbuja, selección e inserción. Estos métodos nos ayudarán a manejar colecciones de objetos de manera más eficiente, mejorando el rendimiento y la efectividad de nuestras aplicaciones en el manejo de grandes volúmenes de información.

METODOLOGÍA DE TRABAJO

- 1.-Investigar mejor los métodos de búsqueda y ordenamiento
- 2.-Elaborar un pequeño pseudocódigo para plantear el programa
- 3.-Elaborar un diagrama de flujo para ver las opciones que quiero que tenga
- 4.-Implementarlas en el programa
- 5.-Corregir errores

REFERENCIAS Y BIBLIOGRAFÍA

<https://parzibyte.me/blog/2020/08/30/java-ordenamiento-seleccion/>
<https://parzibyte.me/blog/2019/12/26/ordenamiento-burbuja-java/>