



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>Aprobación: 2022/03/01</p> | <p>Código: GUIA-PRLE-001</p> | <p>Página: 3</p> |

Clase Main:

```

1 package LABORATORIO_03;
2 import java.util.*;
3 public class DemoBatalla {
4     public static void main(String[] args) {
5         Nave[] misNaves = new Nave[10];
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.next();
15             System.out.print("Fila: ");
16             fil = sc.nextInt();
17             System.out.print("Columna: ");
18             col = sc.next();
19             System.out.print("Estado (true/false): ");
20             est = sc.nextBoolean();
21             System.out.print("Puntos: ");
22             punt = sc.nextInt();
23
24             misNaves[i] = new Nave(); // Se crea un objeto Nave y se asigna su referencia a misNaves
25
26             misNaves[i].setNombre(nomb);
27             misNaves[i].setFila(fil);
28             misNaves[i].setColumna(col);
29             misNaves[i].setEstado(est);
30             misNaves[i].setPuntos(punt);
31         }
32
33         System.out.println("\nNaves creadas:");
34         mostrarNaves(misNaves);
35         mostrarPorNombre(misNaves);
36         mostrarPorPuntos(misNaves);
37         System.out.println("\nNave con mayor número de puntos: " + mostrarMayorPuntos(misNaves).getNombre());
38
39         String respuesta;
40         do {
41             System.out.println("Desea desordenar las naves? (s/n)");
42
43             respuesta = sc.nextLine().toLowerCase();
44
45             if (respuesta.equals("s")) {
46                 Nave[] navesDesordenadas = desordenarNaves(misNaves);
47                 System.out.println("\nNaves desordenadas aleatoriamente:");
48                 mostrarNaves(navesDesordenadas);
49             } else if (!respuesta.equals("n")) {
50                 System.out.println("Respuesta no válida. Por favor ingrese 's' o 'n'.");
51             }
52
53         } while (!respuesta.equals("n"));
54     }
55
56     // Método para mostrar todas las naves
57     public static void mostrarNaves(Nave[] flota) {
58         for (Nave nave : flota) {
59             System.out.println("Nombre: " + nave.getNombre() + ", Fila: " + nave.getFila() +
60                 ", Columna: " + nave.getColumna() + ", Estado: " + nave.getEstado() +
61                 ", Puntos: " + nave.getPuntos());
62         }
63     }
64 }

```

```

64
65 // Método para mostrar todas las naves de un nombre que se pide por teclado
66 public static void mostrarPorNombre(Nave[] flota) {
67     Scanner sc = new Scanner(System.in);
68     System.out.print("\nIngrese el nombre de la nave a buscar: ");
69     String nombreBuscado = sc.next();
70
71     System.out.println("Naves con nombre '" + nombreBuscado + "':");
72     for (Nave nave : flota) {
73         if (nave.getNombre().equalsIgnoreCase(nombreBuscado)) {
74             System.out.println("Nombre: " + nave.getNombre() + ", Fila: " + nave.getFila() +
75                 ", Columna: " + nave.getColumna() + ", Estado: " + nave.getEstado() +
76                 ", Puntos: " + nave.getPuntos());
77         }
78     }
79 }
80
81 // Método para mostrar todas las naves con un número de puntos inferior o igual
82 // al número de puntos que se pide por teclado
83 public static void mostrarPorPuntos(Nave[] flota) {
84     Scanner sc = new Scanner(System.in);
85     System.out.print("\nIngrese el número máximo de puntos: ");
86     int puntosMaximos = sc.nextInt();
87
88     System.out.println("Naves con puntos menores o iguales a " + puntosMaximos + ":");
89     for (Nave nave : flota) {
90         if (nave.getPuntos() <= puntosMaximos) {
91             System.out.println("Nombre: " + nave.getNombre() + ", Fila: " + nave.getFila() +
92                 ", Columna: " + nave.getColumna() + ", Estado: " + nave.getEstado() +
93                 ", Puntos: " + nave.getPuntos());
94         }
95     }
96 }
97
98 // Método que devuelve la Nave con mayor número de Puntos
99 public static Nave mostrarMayorPuntos(Nave[] flota) {
100     Nave naveMayorPuntos = flota[0];
101     for (Nave nave : flota) {
102         if (nave.getPuntos() > naveMayorPuntos.getPuntos()) {
103             naveMayorPuntos = nave;
104         }
105     }
106     return naveMayorPuntos;
107 }
108 // Crear un método que devuelva un nuevo arreglo de objetos con todos los objetos
109 // previamente ingresados pero aleatoriamente desordenados
110 public static void desordenarNavesRecursivo(Nave[] flota, int n) {
111     // si n es 1, no hay más elementos para intercambiar
112     if (n == 1) {
113         return;
114     }
115
116     // Elegir un índice aleatorio entre 0 y n-1
117     Random aleatorio = new Random();
118     int j = aleatorio.nextInt(n);
119
120     // Intercambiar flota[n-1] con flota[j]
121     Nave temp = flota[n - 1];
122     flota[n - 1] = flota[j];
123     flota[j] = temp;
124
125     // llamada recursiva para desordenar los primeros n-1 elementos
126     desordenarNavesRecursivo(flota, n - 1);
127 }
128 public static Nave[] desordenarNaves(Nave[] flota) {
129     // Crear una copia del arreglo original
130     Nave[] navesDesordenadas = new Nave[flota.length];
131     System.arraycopy(flota, 0, navesDesordenadas, 0, flota.length);
132
133     // Desordenamos el arreglo
134     desordenarNavesRecursivo(navesDesordenadas, flota.length);
135     return navesDesordenadas;
136 }
137 }



```

2.-Solucionar la actividad 4 de la Practica 1 pero usando arreglo de Objetos

```
1 package LABORATORIO_03;
2 import java.util.Scanner;
3 public class Actividad4Practica1 {
4     public static void main(String[] args) {
5         Scanner sc = new Scanner(System.in);
6
7         // Crear un arreglo para 5 objetos Soldado
8         Soldado[] soldados = new Soldado[5];
9
10        // Ingreso de datos
11        for (int i = 0; i < soldados.length; i++) {
12            System.out.println("Ingrese los datos del soldado " + (i + 1) + ":");
13            System.out.print("Nombre: ");
14            String nombre = sc.nextLine();
15            System.out.print("Nivel de vida: ");
16            int nivelDeVida = sc.nextInt();
17            sc.nextLine();
18            //Añadimos los datos arreglo soldado
19            soldados[i] = new Soldado(nombre, nivelDeVida);
20        }
21
22        // Mostrar los datos de los soldados
23        System.out.println("\nDatos de los soldados:");
24        for (int i = 0; i < soldados.length; i++) {
25            System.out.println("Soldado " + (i + 1) + ":"+"Nombre: " + soldados[i].getNombre()
26                + " | Nivel de vida: " + soldados[i].getNivelDeVida());
27        }
28    }
29 }
30 }
```

Clase Soldado:

```
1 package LABORATORIO_03;
2
3 public class Soldado {
4     public String nombre;
5     public int nivelDeVida;
6
7
8     public Soldado(String nombre, int nivelDeVida) {
9         this.nombre = nombre;
10        this.nivelDeVida = nivelDeVida;
11    }
12
13    public void setNombre(String nombre){
14        this.nombre=nombre;
15    }
16
17    public void setNivelDeVida(int nivelDeVida){
18        this.nivelDeVida=nivelDeVida;
19    }
20    public String getNombre() {
21        return nombre;
22    }
23
24    public int getNivelDeVida() {
25        return nivelDeVida;
26    }
27 }
```



| | | |
|---|---|---|
|  | <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> |

3.-Solucionar la actividad 5 de la Practica 1 pero usando arreglo de Objetos

```

1 package LABORATORIO_03;
2 import java.util.Random;
3 public class Actividad5Practica1 {
4
5     public static void main(String[] args) {
6         Random aleatorio = new Random();
7         // Generar el número de soldados para los dos ejércitos
8         int numeroEjercito1 = aleatorio.nextInt(5) + 1;
9         int numeroEjercito2 = aleatorio.nextInt(5) + 1;
10
11         // Inicializar los ejércitos con objetos Soldado
12         Soldado[] ejercito1 = inicializarEjercito(numeroEjercito1);
13         Soldado[] ejercito2 = inicializarEjercito(numeroEjercito2);
14
15         System.out.println("Soldados del primer ejército:");
16         mostrarEjercito(ejercito1);
17
18         System.out.println("\nSoldados del segundo ejército:");
19         mostrarEjercito(ejercito2);
20
21         mostrarEjercitoGanador(numeroEjercito1, numeroEjercito2);
22     }
23
24     // Inicializar un ejército con un número de soldados dado
25     public static Soldado[] inicializarEjercito(int numeroDeSoldados) {
26         Soldado[] ejercito = new Soldado[numeroDeSoldados];
27         Random aleatorio = new Random();
28         // Generar nombre y nivel de vida
29         for (int i = 0; i < ejercito.length; i++) {
30             int nivelDeVida = aleatorio.nextInt(5) + 1; // Generar un número entre 1 y 5
31             ejercito[i] = new Soldado("Soldado " + (i + 1), nivelDeVida);
32         }
33         return ejercito;
34     }
35
36     // Mostrar los datos de cada soldado del ejército
37     public static void mostrarEjercito(Soldado[] ejercito) {
38         for (int i = 0; i < ejercito.length; i++) {
39             System.out.println("Nombre: " + ejercito[i].getNombre() + ", Nivel de vida: " + ejercito[i].getNivelDeVida());
40         }
41     }
42
43     // Mostrar el ejército ganador en función de la cantidad de soldados
44     public static void mostrarEjercitoGanador(int ejercito1, int ejercito2) {
45         System.out.println("\nEl primer ejército tiene " + ejercito1 + " soldados.");
46         System.out.println("El segundo ejército tiene " + ejercito2 + " soldados.");
47         if (ejercito1 > ejercito2) {
48             System.out.println("Ganó el primer ejército.");
49         } else if (ejercito2 > ejercito1) {
50             System.out.println("Ganó el segundo ejército.");
51         } else {
52             System.out.println("Hubo un empate.");
53         }
54     }
55 }

```

| | | |
|---|--|---|
|  | <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> |



II. PRUEBAS

a)Ejecución ejercicio1:

Insertamos Datos:

Nave 1
Nombre: NaveAzul
Fila: 4
Columna: 7
Estado (true/false): true
Puntos: 76
Nave 2
Nombre: NaveVerde
Fila: 7
Columna: 5
Estado (true/false): false
Puntos: 56
Nave 3
Nombre: NaveAzul
Fila: 7
Columna: 3
Estado (true/false): false
Puntos: 78
Nave 4
Nombre: NaveRoja
Fila: 4
Columna: 7
Estado (true/false): true
Puntos: 87
Nave 5
Nombre: NaveAzul
Fila: 2
Columna: 7
Estado (true/false): false
Puntos: 34

Nave 6
Nombre: NaveRoja
Fila: 7
Columna: 4
Estado (true/false): true
Puntos: 65
Nave 7
Nombre: NaveAmarilla
Fila: 65
Columna: 3
Estado (true/false): true
Puntos: 76
Nave 8
Nombre: NaveAmarilla
Fila: 5
Columna: 4
Estado (true/false): true
Puntos: 65
Nave 9
Nombre: NaveAzul
Fila: 8
Columna: 6
Estado (true/false): true
Puntos: 45
Nave 10
Nombre: NaveNegra
Fila: 5
Columna: 8
Estado (true/false): true
Puntos: 98

| | | |
|---|--|---|
|  | <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> |

Ejecución:

Naves creadas:

Nombre: NaveAzul, Fila: 4, Columna: 7, Estado: true, Puntos: 76
Nombre: NaveVerde, Fila: 7, Columna: 5, Estado: false, Puntos: 56
Nombre: NaveAzul, Fila: 7, Columna: 3, Estado: false, Puntos: 78
Nombre: NaveRoja, Fila: 4, Columna: 7, Estado: true, Puntos: 87
Nombre: NaveAzul, Fila: 2, Columna: 7, Estado: false, Puntos: 34
Nombre: NaveRoja, Fila: 7, Columna: 4, Estado: true, Puntos: 65
Nombre: NaveAmarilla, Fila: 65, Columna: 3, Estado: true, Puntos: 76
Nombre: NaveAmarilla, Fila: 5, Columna: 4, Estado: true, Puntos: 65
Nombre: NaveAzul, Fila: 8, Columna: 6, Estado: true, Puntos: 45
Nombre: NaveNegra, Fila: 5, Columna: 8, Estado: true, Puntos: 98

Ingrese el nombre de la nave a buscar: **NaveAzul**



Naves con nombre 'NaveAzul':

Nombre: NaveAzul, Fila: 4, Columna: 7, Estado: true, Puntos: 76
Nombre: NaveAzul, Fila: 7, Columna: 3, Estado: false, Puntos: 78
Nombre: NaveAzul, Fila: 2, Columna: 7, Estado: false, Puntos: 34
Nombre: NaveAzul, Fila: 8, Columna: 6, Estado: true, Puntos: 45

Ingrese el número máximo de puntos: **76**

Naves con puntos menores o iguales a 76:

Nombre: NaveAzul, Fila: 4, Columna: 7, Estado: true, Puntos: 76
Nombre: NaveVerde, Fila: 7, Columna: 5, Estado: false, Puntos: 56
Nombre: NaveAzul, Fila: 2, Columna: 7, Estado: false, Puntos: 34
Nombre: NaveRoja, Fila: 7, Columna: 4, Estado: true, Puntos: 65
Nombre: NaveAmarilla, Fila: 65, Columna: 3, Estado: true, Puntos: 76
Nombre: NaveAmarilla, Fila: 5, Columna: 4, Estado: true, Puntos: 65
Nombre: NaveAzul, Fila: 8, Columna: 6, Estado: true, Puntos: 45

| | | |
|---|--|---|
|  | <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> |

Desea desordenar las naves? (s/n)

s

Naves desordenadas aleatoriamente:



Nombre: NaveVerde, Fila: 7, Columna: 5, Estado: false, Puntos: 56
Nombre: NaveRoja, Fila: 4, Columna: 7, Estado: true, Puntos: 87
Nombre: NaveAzul, Fila: 7, Columna: 3, Estado: false, Puntos: 78
Nombre: NaveNegra, Fila: 5, Columna: 8, Estado: true, Puntos: 98
Nombre: NaveAzul, Fila: 2, Columna: 7, Estado: false, Puntos: 34
Nombre: NaveAzul, Fila: 8, Columna: 6, Estado: true, Puntos: 45
Nombre: NaveAmarilla, Fila: 5, Columna: 4, Estado: true, Puntos: 65
Nombre: NaveRoja, Fila: 7, Columna: 4, Estado: true, Puntos: 65
Nombre: NaveAmarilla, Fila: 65, Columna: 3, Estado: true, Puntos: 76
Nombre: NaveAzul, Fila: 4, Columna: 7, Estado: true, Puntos: 76
Desea desordenar las naves? (s/n)

s

Naves desordenadas aleatoriamente:

Nombre: NaveAzul, Fila: 4, Columna: 7, Estado: true, Puntos: 76
Nombre: NaveAmarilla, Fila: 65, Columna: 3, Estado: true, Puntos: 76
Nombre: NaveAzul, Fila: 2, Columna: 7, Estado: false, Puntos: 34
Nombre: NaveAzul, Fila: 7, Columna: 3, Estado: false, Puntos: 78
Nombre: NaveRoja, Fila: 7, Columna: 4, Estado: true, Puntos: 65
Nombre: NaveRoja, Fila: 4, Columna: 7, Estado: true, Puntos: 87
Nombre: NaveNegra, Fila: 5, Columna: 8, Estado: true, Puntos: 98
Nombre: NaveVerde, Fila: 7, Columna: 5, Estado: false, Puntos: 56
Nombre: NaveAzul, Fila: 8, Columna: 6, Estado: true, Puntos: 45
Nombre: NaveAmarilla, Fila: 5, Columna: 4, Estado: true, Puntos: 65
Desea desordenar las naves? (s/n)

n

| | | |
|--|--|---|
|  | <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: 10</p> |



b)Ejecución Ejercicio2:

```

Ingrese los datos del soldado 1:
Nombre: Sebastian
Nivel de vida: 4
Ingrese los datos del soldado 2:
Nombre: Juan
Nivel de vida: 3
Ingrese los datos del soldado 3:
Nombre: Pepe
Nivel de vida: 5
Ingrese los datos del soldado 4:
Nombre: Carlos
Nivel de vida: 4
Ingrese los datos del soldado 5:
Nombre: Jhon
Nivel de vida: 3

Datos de los soldados:
Soldado 1:Nombre: Sebastian | Nivel de vida: 4
Soldado 2:Nombre: Juan | Nivel de vida: 3
Soldado 3:Nombre: Pepe | Nivel de vida: 5
Soldado 4:Nombre: Carlos | Nivel de vida: 4
Soldado 5:Nombre: Jhon | Nivel de vida: 3

```

| | | |
|--|--|---|
|  | <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: 11</p> |

c)Ejecución ejercicio3:

En caso de ganar el primer ejercito:

Soldados del primer ejército:

Nombre: Soldado 1, Nivel de vida: 2

Nombre: Soldado 2, Nivel de vida: 2

Nombre: Soldado 3, Nivel de vida: 2

Nombre: Soldado 4, Nivel de vida: 1

Nombre: Soldado 5, Nivel de vida: 5

Soldados del segundo ejército:

Nombre: Soldado 1, Nivel de vida: 3

El primer ejército tiene 5 soldados.

El segundo ejército tiene 1 soldados.

Ganó el primer ejército.

En caso de ganar el segundo ejercito:

Soldados del primer ejército:

Nombre: Soldado 1, Nivel de vida: 4

Nombre: Soldado 2, Nivel de vida: 2

Nombre: Soldado 3, Nivel de vida: 2

Nombre: Soldado 4, Nivel de vida: 3

Soldados del segundo ejército:

Nombre: Soldado 1, Nivel de vida: 4

Nombre: Soldado 2, Nivel de vida: 3

Nombre: Soldado 3, Nivel de vida: 4



Nombre: Soldado 4, Nivel de vida: 1

Nombre: Soldado 5, Nivel de vida: 4

El primer ejército tiene 4 soldados.

El segundo ejército tiene 5 soldados.

Ganó el segundo ejército.

| | | |
|---|--|---|
|  | <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> |

En caso de empate:

Soldados del primer ejército:

Nombre: Soldado 1, Nivel de vida: 4

Nombre: Soldado 2, Nivel de vida: 2

Nombre: Soldado 3, Nivel de vida: 1

Soldados del segundo ejército:

Nombre: Soldado 1, Nivel de vida: 5

Nombre: Soldado 2, Nivel de vida: 5

Nombre: Soldado 3, Nivel de vida: 2

El primer ejército tiene 3 soldados.

El segundo ejército tiene 3 soldados.

Hubo un empate.

Mi commit al repositorio:

 MINGW64:c:/Users/Windows/Desktop/RepositorioLocal

```
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_03

Windows@DESKTOP-C32KACO MINGW64 ~/Desktop/RepositorioLocal (main)
$ git commit -m "Agregar LABORATORIO_03"
[main a35ce8c] Agregar LABORATORIO_03
5 files changed, 302 insertions(+)
create mode 100644 LABORATORIO_03/Actividad4Practical.java
create mode 100644 LABORATORIO_03/Actividad5Practical.java
create mode 100644 LABORATORIO_03/DemoBatalla.java
create mode 100644 LABORATORIO_03/Nave.java
create mode 100644 LABORATORIO_03/Soldado.java



Windows@DESKTOP-C32KACO MINGW64 ~/Desktop/RepositorioLocal (main)
$ AC

Windows@DESKTOP-C32KACO MINGW64 ~/Desktop/RepositorioLocal (main)
$ git push origin main
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 8 threads
Compressing objects: 100% (8/8), done.
Writing objects: 100% (8/8), 3.47 KiB | 3.47 MiB/s, done.
Total 8 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/JoseMorocco/FP2.git
4f80326..a35ce8c main -> main

Windows@DESKTOP-C32KACO MINGW64 ~/Desktop/RepositorioLocal (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 | 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 | |

| | | |
|---|--|---|
|  | <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: 14</p> |

CONCLUSIONES

En este laboratorio, pudimos explorar el uso de arreglos de objetos, una técnica que nos permite manejar de manera más eficiente y organizada los datos asociados a entidades complejas, en comparación con el uso de arreglos estándar que pueden resultar menos flexibles. Además de investigar y analizar algoritmos, como en mi caso, para desordenar un conjunto de naves desafío que hasta el momento no me había cruzado.

METODOLOGÍA DE TRABAJO

- 1.-Un pequeño pseudocódigo para plantear el programa
- 2.-Un diagrama de flujo para ver las opciones que quiero que tenga
- 3.-Implementarlas en el programa
- 4.-Corregir errores

REFERENCIAS Y BIBLIOGRAFÍA

<https://www.geeksforgeeks.org/shuffle-a-given-array-using-fisher-yates-shuffle-algorithm/>