
	<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;"><b>Formato:</b> 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

### (formato estudiante)

INFORMACIÓN BÁSICA					
<b>ASIGNATURA:</b>	<i>Fundamentos de la programación 02</i>				
<b>TÍTULO DE LA PRÁCTICA:</b>	<i>ArrayList</i>				
<b>NÚMERO DE PRÁCTICA:</b>	<i>06</i>	<b>AÑO LECTIVO:</b>	<i>2024-B</i>	<b>NRO. SEMESTRE:</b>	<i>//</i>
<b>FECHA DE PRESENTACIÓN</b>	<i>25/10/2024</i>	<b>HORA DE PRESENTACIÓN</b>	<i>18:00:00</i>		
<b>INTEGRANTE (s)</b> <i>Mauro Snayder Sullca Mamani</i>				<b>NOTA (0-20)</b>	
<b>DOCENTE(s):</b> <i>Ing. Lino Jose Pinto Oppe</i>					

RESULTADOS Y PRUEBAS	
<b>I. EJERCICIOS RESUELTOS:</b>	
<pre> 7  /** 8   * 9   * @author Mauro Snayder 10  */ 11  public class Soldado { 12      // Creamos los atributos 13      private String nombre= "          "; 14      private int vida; 15      private int fila; 16      private int columna; 17 18      // Creamos los Set y Get de cada atributo 19      public String getNombre() { 20          return nombre; 21      } 22 23      public void setNombre(String nombre) { 24          this.nombre = nombre; 25      } 26 27      public int getVida() { 28          return vida; 29      } 30 31      public int getFila(){ 32          return fila; 33      } 34 35      public int getColumna(){ 36          return columna; 37      } </pre>	



```
38 //Generamos una posicion aleatoria para el soldado
39 public void aleatorioPosicion(int fila,int columna){
40     this.fila=(int) (Math.random()*fila);
41     this.columna=(int) (Math.random()*columna);
42 }
43 //Generamos la vida del soldado
44 public void aleatorioVida(){
45     this.vida=(int) (Math.random()*5+1);
46 }
47 // Creamos el toString
48 public String toString() {
49     return "Soldado{" + "nombre=" + nombre + ", vida=" + vida + ", fila=" + fila + ", columna=" + columna + "}\n";
50 }
51 }
```

```
8
9 * @author Mauro Snyder
10 */
11 import java.util.*;
12
13 public class Actividad05 {
14     public static void main(String[] args) {
15         int fila=10;//Generamos la fila de la tabla
16         int columna=10;//Generamos la columna de la tabla
17         ArrayList<ArrayList<Soldado>> tabla=new ArrayList<ArrayList<Soldado>>();//Creamos la tabla
18         ArrayList<Soldado> ejercito1=new ArrayList<Soldado>();//Creamos ejercito 1
19         ArrayList<Soldado> ejercito2=new ArrayList<Soldado>();//Creamos ejercito 2
20         for (int i=0;i<fila;i++){//Inicializamos el tablero con valores vacios
21             tabla.add(new ArrayList<Soldado>());
22             for (int j=0;j<columna;j++){
23                 tabla.get(i).add(new Soldado());
24             }
25         }
26         inicializarEjercito(tabla,ejercito1,1);
27         inicializarEjercito(tabla,ejercito2,2);
28         mostrarEjercitoTabla(tabla);
29         System.out.print("\nEl soldado con mayor vida del ejercito 1 es: "+mayorVida(ejercito1).toString());
30         System.out.print("\nEl soldado con mayor vida del ejercito 2 es: "+mayorVida(ejercito2).toString());
31         System.out.println("\nEl promedio de la vida del ejercito 1 es: "+vidaPromedio(ejercito1));
32         System.out.println("\nEl promedio de la vida del ejercito 2 es: "+vidaPromedio(ejercito2));
33         System.out.println("\nLista de los soldados por orden de creacion: ");
34         mostrarEjercitoOrdenCreacion(ejercito1,1);
35         mostrarEjercitoOrdenCreacion(ejercito2,2);
36         System.out.println("\nEl ranking de los soldados es: ");
37         rankingSoldadosV1(ejercito1,1);
38         rankingSoldadosV2(ejercito2,2);
39         ganador(ejercito1,ejercito2);
40     }
41     // Método para inicializar una tabla con cierto numeros de soldados
42     public static void inicializarEjercito(ArrayList<ArrayList<Soldado>> tabla,ArrayList<Soldado> ejercito,int tipo){
43         int numSoldados=(int) (Math.random()*10+1);//Genemos la cantidad de soldados
44         for (int i=0;i<numSoldados;i++){
45             Soldado persona=new Soldado();//creamos "persona" para luego ponerlo dentro del tablero y del Array del ejercito
46             do {
47                 persona.aleatorioPosicion(tabla.size(),tabla.get(0).size());//generamos una posicion
48                 persona.aleatorioVida();//generamos la vida
49                 persona.setNombre("soldado"+persona.getFila()+"X"+persona.getColumna()+" E"+tipo);//Generamos el nombre del soldado
50             }
51             //El bucle se repite si en una posicion aleatoria ya existe un soldado puesto
52             while(!tabla.get(persona.getFila()).get(persona.getColumna()).getNombre().equals(""));
53             ejercito.add(persona);//ponemos el soldado dentro del Array del ejercito
54             tabla.get(persona.getFila()).set(persona.getColumna(),persona);//ponemos el soldado dentro del tablero
55         }
56     }
```

```

57 // Metodo para mostrar la tabla
58 public static void mostrarEjercitoTabla(ArrayList<ArrayList<Soldado>> tabla){
59     System.out.println("
60     for (int i=0;i<tabla.size();i++){
61         System.out.print("
62         for (int j=0;j<tabla.get(i).size();j++){
63             System.out.print(tabla.get(i).get(j).getNombre()+"|");
64             System.out.println();
65             System.out.println("
66     }
67 }
68 //Metodo para determinar el soldado con mayor vida
69 public static Soldado mayorVida(ArrayList<Soldado> ejercito){
70     Soldado mayor=new Soldado();//creamos un objeto para almacenar al soldado mayor
71     for (Soldado persona:ejercito){ //Recorremos todo el Array del ejercito
72         if (persona.getVida()>mayor.getVida())//Buscamos al soldado con mayor vida
73             mayor=persona;
74     }
75     return mayor;
76 }
77 //Metodo para determinar la vida total de todos los soldados
78 public static double vidaPromedio(ArrayList<Soldado> ejercito){
79     double vidaT=0;//vida inicial
80     for (Soldado persona:ejercito){ //Recorremos todo el Array del ejercito
81         vidaT+=persona.getVida();//Sumamos las vidas de todos los soldados
82     }
83     return vidaT/ejercito.size();
84 }
85 //Metodo para ver las lista de los soldados por el orden de creacion
86 public static void mostrarEjercitoOrdenCreacion(ArrayList<Soldado> ejercito,int tipo){
87     System.out.println("Ejercito "+tipo+" : ");
88     for (Soldado persona:ejercito){ //Imprimimos todos los soldados del Array ejercito
89         System.out.print(persona.toString());
90     }
91 }
92 //Metodo para ver el ranking de los soldados version 1(por vida)
93 public static void rankingSoldadosV1(ArrayList<Soldado> ejercito,int tipo){
94     ordenamientoBurbuja(ejercito);//ordenamos el Array Unidimensional
95     System.out.println("Ejercito "+tipo+" : ");
96     for (int i=ejercito.size()-1;i>=0;i--){ //Imprimimos el Array
97         System.out.print(ejercito.get(i).toString());
98     }
99 }
100 //Metodo para ver el ranking de los soldados version 2(por vida)
101 public static void rankingSoldadosV2(ArrayList<Soldado> ejercito,int tipo){
102     ordenamientoInsercion(ejercito);//ordenamos el Array Unidimensional
103     System.out.println("Ejercito "+tipo+" : ");
104     for (int i=ejercito.size()-1;i>=0;i--){ //Imprimimos el Array
105         System.out.print(ejercito.get(i).toString());
106     }
107 }
108 //Metodo de ordenamiento Burbuja para la vida de los soldados
109 public static void ordenamientoBurbuja(ArrayList<Soldado> lista){
110     Soldado cambio;
111     for(int i=0;i<lista.size()-1;i++){
112         for(int j=0;j<lista.size()-1-l;i++){
113             if (lista.get(j).getVida()>lista.get(j+1).getVida()){
114                 cambio=lista.get(j);
115                 lista.set(j,lista.get(j+1));
116                 lista.set(j+1,cambio);
117             }
118         }
119     }
120 }
121 //Metodo de ordenamiento de insercion para la vida de los soldados
122 public static void ordenamientoInsercion(ArrayList<Soldado> lista) {
123     for (int i=1;i<lista.size();i++){
124         Soldado soldadoActual=lista.get(i);
125         int j=i-1;
126         while (j>=0 && lista.get(j).getVida()>soldadoActual.getVida()){
127             lista.set(j+1,lista.get(j));
128             j--;
129         }
130         lista.set(j+1,soldadoActual);
131     }
132 }
133 //Determinar el ganador de la batalla (por la cantidad de soldados que tiene cada ejercito)
134 //Gana el ejercito que tiene mas soldados.
135 public static void ganador(ArrayList<Soldado> ejercito1,ArrayList<Soldado> ejercito2){
136     if (ejercito1.size()>ejercito2.size()){
137         System.out.println("\nGana el ejercito 1.");
138     } else if (ejercito1.size()<ejercito2.size()){
139         System.out.println("\nGana el ejercito 2.");
140     } else
141         System.out.println("\nQuedan empatados los ejercitos.");
142 }
143 }
144

```

	<p style="text-align: center;"><b>UNIVERSIDAD NACIONAL DE SAN AGUSTIN</b>  <b>FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS</b>  <b>ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</b></p>	
<p style="text-align: center;"><b>Formato:</b> Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p><b>Aprobación:</b> 2022/03/01</p>	<p><b>Código:</b> GUIA-PRLE-001</p>	<p><b>Página:</b> 4</p>

## II. PRUEBAS

*¿Con que valores comprobaste que tu práctica estuviera correcta?*

*Para comprobar que mi práctica estaba correcta, utilicé varios valores aleatorios para las posiciones y la vida de los soldados en ambos ejércitos. En particular, dejé que el programa generara de manera aleatoria entre 1 y 10 soldados para cada ejército, y las vidas también se generaron aleatoriamente en un rango de 1 a 5.*

*¿Qué resultado esperabas obtener para cada valor de entrada?*

*Al ejecutar el programa, esperaba que se mostrara una tabla con los soldados ubicados correctamente en posiciones aleatorias dentro de la matriz 10x10. También anticipaba que el programa identificara sin errores el soldado con mayor vida de cada ejército y que calculara adecuadamente el promedio de vida de los soldados. Además, esperaba que se listaran los soldados en el orden de su creación y que se mostrara un ranking de los soldados por vida, utilizando dos métodos de ordenamiento diferentes. Finalmente, esperaba que se determinara un ganador basado en la cantidad de soldados en cada ejército.*

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

*Al ejecutar el programa, obtuve el comportamiento esperado en todos los casos. La tabla se llenó con soldados en posiciones aleatorias y se mostraron nombres como "soldadoXxY E1" o "soldadoXxY E2" correctamente. El programa identificó el soldado con mayor vida en cada ejército, calculó el promedio de vida de los soldados adecuadamente y mostró los soldados en el orden en que fueron creados. Ambos métodos de ordenamiento (burbuja e inserción) mostraron correctamente los soldados ordenados de acuerdo a su vida. Finalmente, el programa determinó el ganador al comparar el número de soldados en cada ejército, mostrando si uno ganó o si empataron.*

La primera ejecución del programa podemos ver que se generó una tabla 10x10 con 6 soldados en total. En el primer ejército se crearon 5 soldados donde el soldado con mayor vida es el soldado0x4 y el promedio de vida es de 3.6, mientras que en el segundo ejército se creó un soldado donde este mismo es el soldado con mayor vida y el promedio de vida es 4. Y por último nos muestra la lista de los soldados por orden de creación, por orden de vida (por ejército) y el ganador de la batalla (por la cantidad de soldados por ejército).

```
Output - Laboratorios (run)

run:
| | | | |soldado0X4 E1| | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | |soldado2X5 E1|soldado2X6 E1| | | |soldado2X9 E2|
| | | | | | | | | | |
| | | | |soldado4X2 E1| | | | | | |
| | | | | | | | | | |
| | | | |soldado6X1 E1| | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |
| | | | | | | | | | |

El soldado con mayor vida del ejercito 1 es: Soldado{nombre=soldado0X4 E1, vida=5, fila=0, columna=4}
El soldado con mayor vida del ejercito 2 es: Soldado{nombre=soldado2X9 E2, vida=4, fila=2, columna=9}

El promedio de la vida del ejercito 1 es: 3.6
El promedio de la vida del ejercito 2 es: 4.0

Lista de los soldados por orden de creacion:
Ejercito 1 :
Soldado{nombre=soldado0X4 E1, vida=5, fila=0, columna=4}
Soldado{nombre=soldado4X2 E1, vida=1, fila=4, columna=2}
Soldado{nombre=soldado2X6 E1, vida=5, fila=2, columna=6}
Soldado{nombre=soldado2X5 E1, vida=5, fila=2, columna=5}
Soldado{nombre=soldado6X1 E1, vida=2, fila=6, columna=1}
Ejercito 2 :
Soldado{nombre=soldado2X9 E2, vida=4, fila=2, columna=9}

El ranking de los soldados es:
Ejercito 1 :
Soldado{nombre=soldado2X5 E1, vida=5, fila=2, columna=5}
Soldado{nombre=soldado2X6 E1, vida=5, fila=2, columna=6}
Soldado{nombre=soldado0X4 E1, vida=5, fila=0, columna=4}
Soldado{nombre=soldado6X1 E1, vida=2, fila=6, columna=1}
Soldado{nombre=soldado4X2 E1, vida=1, fila=4, columna=2}
Ejercito 2 :
Soldado{nombre=soldado2X9 E2, vida=4, fila=2, columna=9}

Gana el ejercito 1.
BUILD SUCCESSFUL (total time: 0 seconds)
```

La primera ejecución del programa podemos ver que se generó una tabla 10x10 con 12 soldados en total. En el primer ejército se crearon 7 soldados donde el soldado con mayor vida es el soldado2X3 y el promedio de vida es de 3.86, mientras que en el segundo ejército se creó 5 soldados donde el soldado con mayor vida es el soldado9X5 y el promedio de vida es 3.2. Y por último nos muestra la lista de los soldados por orden de creación, por orden de vida (por ejército) y el ganador de la batalla (por la cantidad de soldados por ejército).

```

Output - Laboratorios (run)

run:



|  |  |               |               |  |               |               |               |               |  |
|--|--|---------------|---------------|--|---------------|---------------|---------------|---------------|--|
|  |  |               |               |  |               |               |               |               |  |
|  |  |               | soldado1X3 E1 |  |               | soldado1X6 E2 |               |               |  |
|  |  |               | soldado2X3 E1 |  |               |               | soldado2X8 E2 |               |  |
|  |  |               |               |  |               |               | soldado3X8 E1 |               |  |
|  |  |               |               |  |               |               |               |               |  |
|  |  | soldado4X1 E2 |               |  |               |               |               |               |  |
|  |  |               |               |  |               |               | soldado5X8 E1 | soldado5X9 E2 |  |
|  |  |               | soldado6X2 E1 |  |               |               |               |               |  |
|  |  |               |               |  |               | soldado7X6 E1 |               |               |  |
|  |  |               |               |  |               |               |               |               |  |
|  |  |               |               |  | soldado9X4 E1 | soldado9X5 E2 |               |               |  |



El soldado con mayor vida del ejercito 1 es: Soldado{nombre=soldado2X3 E1, vida=5, fila=2, columna=3}
El soldado con mayor vida del ejercito 2 es: Soldado{nombre=soldado9X5 E2, vida=5, fila=9, columna=5}

El promedio de la vida del ejercito 1 es: 3.857142857142857
El promedio de la vida del ejercito 2 es: 3.2

Lista de los soldados por orden de creacion:
Ejercito 1 :
Soldado{nombre=soldado2X3 E1, vida=5, fila=2, columna=3}
Soldado{nombre=soldado5X8 E1, vida=4, fila=5, columna=8}
Soldado{nombre=soldado7X6 E1, vida=1, fila=7, columna=6}
Soldado{nombre=soldado1X3 E1, vida=4, fila=1, columna=3}
Soldado{nombre=soldado3X8 E1, vida=3, fila=3, columna=8}
Soldado{nombre=soldado6X2 E1, vida=5, fila=6, columna=2}
Soldado{nombre=soldado9X4 E1, vida=5, fila=9, columna=4}
Ejercito 2 :
Soldado{nombre=soldado2X8 E2, vida=4, fila=2, columna=8}
Soldado{nombre=soldado9X5 E2, vida=5, fila=9, columna=5}
Soldado{nombre=soldado1X6 E2, vida=1, fila=1, columna=6}
Soldado{nombre=soldado4X1 E2, vida=2, fila=4, columna=1}
Soldado{nombre=soldado5X9 E2, vida=4, fila=5, columna=9}

El ranking de los soldados es:
Ejercito 1 :
Soldado{nombre=soldado9X4 E1, vida=5, fila=9, columna=4}
Soldado{nombre=soldado6X2 E1, vida=5, fila=6, columna=2}
Soldado{nombre=soldado2X3 E1, vida=5, fila=2, columna=3}
Soldado{nombre=soldado1X3 E1, vida=4, fila=1, columna=3}
Soldado{nombre=soldado5X8 E1, vida=4, fila=5, columna=8}
Soldado{nombre=soldado3X8 E1, vida=3, fila=3, columna=8}
Soldado{nombre=soldado7X6 E1, vida=1, fila=7, columna=6}
Ejercito 2 :
Soldado{nombre=soldado9X5 E2, vida=5, fila=9, columna=5}
Soldado{nombre=soldado5X9 E2, vida=4, fila=5, columna=9}
Soldado{nombre=soldado2X8 E2, vida=4, fila=2, columna=8}
Soldado{nombre=soldado4X1 E2, vida=2, fila=4, columna=1}
Soldado{nombre=soldado1X6 E2, vida=1, fila=1, columna=6}

Gana el ejercito 1.
BUILD SUCCESSFUL (total time: 0 seconds)

```

La primera ejecución del programa podemos ver que se generó una tabla 10x10 con 16 soldados en total. En el primer ejército se crearon 6 soldados donde el soldado con mayor vida es el soldado6x9 y el promedio de vida es de 3.66, mientras que en el segundo ejército se creó 10 soldados donde el soldado con mayor vida es el soldado0x0 y el promedio de vida es 3.0. Y por último nos muestra la lista de los soldados por orden de creación, por orden de vida (por ejército) y el ganador de la batalla (por la cantidad de soldados por ejército).

Output - Laboratorios (run)

run:

soldado0X0 E2					soldado0X5 E1				
						soldado1X6 E1	soldado1X7 E2		
						soldado3X6 E2			
	soldado4X1 E2			soldado4X4 E1	soldado4X5 E2	soldado4X6 E2			
	soldado5X1 E1								
					soldado6X5 E2				soldado6X9 E1
							soldado8X7 E2	soldado8X8 E1	soldado8X9 E2
							soldado9X7 E2		

El soldado con mayor vida del ejercito 1 es: Soldado[nombre=soldado6X9 E1, vida=5, fila=6, columna=9]  
El soldado con mayor vida del ejercito 2 es: Soldado[nombre=soldado0X0 E2, vida=4, fila=0, columna=0]

El promedio de la vida del ejercito 1 es: 3.6666666666666665  
El promedio de la vida del ejercito 2 es: 3.0

Lista de los soldados por orden de creacion:

Ejercito 1 :

Soldado[nombre=soldado0X5 E1, vida=4, fila=0, columna=5]  
Soldado[nombre=soldado6X9 E1, vida=5, fila=6, columna=9]  
Soldado[nombre=soldado4X4 E1, vida=4, fila=4, columna=4]  
Soldado[nombre=soldado8X8 E1, vida=4, fila=8, columna=8]  
Soldado[nombre=soldado1X6 E1, vida=1, fila=1, columna=6]  
Soldado[nombre=soldado5X1 E1, vida=4, fila=5, columna=1]

Ejercito 2 :

Soldado[nombre=soldado0X0 E2, vida=4, fila=0, columna=0]  
Soldado[nombre=soldado1X7 E2, vida=2, fila=1, columna=7]  
Soldado[nombre=soldado8X7 E2, vida=3, fila=8, columna=7]  
Soldado[nombre=soldado6X5 E2, vida=2, fila=6, columna=5]  
Soldado[nombre=soldado8X9 E2, vida=4, fila=8, columna=9]  
Soldado[nombre=soldado9X7 E2, vida=3, fila=9, columna=7]  
Soldado[nombre=soldado4X1 E2, vida=1, fila=4, columna=1]  
Soldado[nombre=soldado4X6 E2, vida=4, fila=4, columna=6]  
Soldado[nombre=soldado3X6 E2, vida=4, fila=3, columna=6]  
Soldado[nombre=soldado4X5 E2, vida=3, fila=4, columna=5]

El ranking de los soldados es:

Ejercito 1 :

Soldado[nombre=soldado6X9 E1, vida=5, fila=6, columna=9]  
Soldado[nombre=soldado5X1 E1, vida=4, fila=5, columna=1]  
Soldado[nombre=soldado8X8 E1, vida=4, fila=8, columna=8]  
Soldado[nombre=soldado4X4 E1, vida=4, fila=4, columna=4]  
Soldado[nombre=soldado0X5 E1, vida=4, fila=0, columna=5]  
Soldado[nombre=soldado1X6 E1, vida=1, fila=1, columna=6]



Ejercito 2 :

Soldado[nombre=soldado3X6 E2, vida=4, fila=3, columna=6]  
Soldado[nombre=soldado4X6 E2, vida=4, fila=4, columna=6]  
Soldado[nombre=soldado8X9 E2, vida=4, fila=8, columna=9]  
Soldado[nombre=soldado0X0 E2, vida=4, fila=0, columna=0]  
Soldado[nombre=soldado4X5 E2, vida=3, fila=4, columna=5]  
Soldado[nombre=soldado9X7 E2, vida=3, fila=9, columna=7]  
Soldado[nombre=soldado8X7 E2, vida=3, fila=8, columna=7]  
Soldado[nombre=soldado6X5 E2, vida=2, fila=6, columna=5]  
Soldado[nombre=soldado1X7 E2, vida=2, fila=1, columna=7]  
Soldado[nombre=soldado4X1 E2, vida=1, fila=4, columna=1]

Gana el ejercito 2.

BUILD SUCCESSFUL (total time: 0 seconds)

Output

	<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;"><b>Formato:</b> 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>

### III. COMMITS:

Entramos a nuestra carpeta donde están nuestros archivos y añadimos los cambios.

```

MINGW64:/c/Users/Mauro Snayder/Documents/NetBeansProjects/Laboratorios/src
Mauro Snayder@Mauro MINGW64 ~/Documents/NetBeansProjects/Laboratorios/src (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   Laboratorio_06/Actividad05.java
        modified:   Laboratorio_06/Soldado.java

no changes added to commit (use "git add" and/or "git commit -a")

Mauro Snayder@Mauro MINGW64 ~/Documents/NetBeansProjects/Laboratorios/src (master)
$ git add .

Mauro Snayder@Mauro MINGW64 ~/Documents/NetBeansProjects/Laboratorios/src (master)
$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   Laboratorio_06/Actividad05.java
        modified:   Laboratorio_06/Soldado.java

Mauro Snayder@Mauro MINGW64 ~/Documents/NetBeansProjects/Laboratorios/src (master)
$ |

```

Hacemos un commit

```

Mauro Snayder@Mauro MINGW64 ~/Documents/NetBeansProjects/Laboratorios/src (master)
$ git commit -m "terminado"
[master bf74aca] terminado
 2 files changed, 23 insertions(+), 24 deletions(-)

Mauro Snayder@Mauro MINGW64 ~/Documents/NetBeansProjects/Laboratorios/src (master)
$ |

```



Subimos nuestro commit a nuestro repositorio remoto

```
Mauro Snayder@Mauro MINGW64 ~/Documents/NetBeansProjects/Laboratorios/src (master)
$ git push -u origin master
Enumerating objects: 9, done.
Counting objects: 100% (9/9), done.
Delta compression using up to 20 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (5/5), 830 bytes | 830.00 KiB/s, done.
Total 5 (delta 3), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To https://github.com/MauroSullcaMamani/FDLP2_LAB.git
d3a4de9..bf74aca master -> master
branch 'master' set up to track 'origin/master'.



Mauro Snayder@Mauro MINGW64 ~/Documents/NetBeansProjects/Laboratorios/src (master)
$
```

**Link de mi repositorio:** [https://github.com/MauroSullcaMamani/FDLP2\\_LAB.git](https://github.com/MauroSullcaMamani/FDLP2_LAB.git)

#### IV. 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	✓	2	
2. Commits	Hay capturas de pantalla de los commits más importantes con sus explicaciones detalladas. (El profesor puede preguntar para refrendar calificación).	4	✓	3	
3. Código fuente	Hay porciones de código fuente importantes con numeración y explicaciones detalladas de sus funciones.	2	✓	2	
4. Ejecución	Se incluyen ejecuciones/pruebas del código fuente explicadas gradualmente.	2	✓	2	
5. Pregunta	Se responde con completitud a la pregunta formulada en la tarea. (El profesor puede preguntar para refrendar calificación).	2	✓	2	
6. Fechas	Las fechas de modificación del código fuente están dentro de los plazos de fecha de entrega establecidos.	2	✓	2	
7. Ortografía	El documento no muestra errores ortográficos.	2	✓	2	
8. Madurez	El Informe muestra de manera general una evolución de la madurez del código fuente, explicaciones puntuales pero precisas y un acabado impecable. (El profesor puede preguntar para refrendar calificación).	4	✓	3	
TOTAL		20		18	

Tabla 2: Rúbrica para contenido del Informe y demostración

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

## CONCLUSIONES

*Los ArrayList unidimensionales, como los utilizados para representar los ejércitos, permiten una gestión sencilla y dinámica de los soldados, facilitando operaciones como el cálculo de la vida promedio o la identificación del soldado con mayor vida. Por otro lado, los ArrayList bidimensionales me ofrecen una representación clara del campo de batalla, permitiendo organizar visualmente la posición de los soldados y mantener la lógica del juego. Esta combinación me permite crear un código más organizado y eficiente para simular la batalla.*

## METODOLOGÍA DE TRABAJO

*Lo primero que hice, es leer cada los enunciados de cada ejercicio y también tomar en cuenta las restricciones que nos da para así poder buscar una solución al problema. Después observar el código y entender la funcionalidad de cada uno y completar las partes que están incompletas. Y por último comprobar nuestro código ingresando varias veces valores de prueba para ver que nuestro código está funcionando correctamente.*

## REFERENCIAS Y BIBLIOGRAFÍA

*E. G. Castro Gutiérrez and M. W. Aedo López, Fundamentos de programación 2: tópicos de programación orientada a objetos, 1st ed. Arequipa, Perú: Universidad Nacional de San Agustín, 2021, pp. 170, ISBN 978-612 5035-20-2.*