Informe de Laboratorio 06

Tema: ArrayList

Nota

Estudiante	Escuela	Asignatura
Christian Mestas Zegarra cmestasz@unsa.edu.pe	Escuela Profesional de Ingeniería de Sistemas	Fundamentos de la Programación 2 Semestre: II Código: 1701213

Laboratorio	Tema	Duración
06	ArrayList	04 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2023 - B	Del 11 Octubre 2023	Al 16 Octubre 2023

1. Tarea

- **Actividad 1:** Cree un Proyecto llamado Laboratorio6
- **Actividad 2:** Usted deberá crear las dos clases Soldado.java y VideoJuego3.java. Puede reutilizar lo desarrollado en Laboratorios anteriores.
- **Actividad 3:** Del Soldado nos importa el nombre, puntos de vida, fila y columna (posición en el tablero).
- **Actividad 4:** El juego se desarrollará en el mismo tablero de los laboratorios anteriores. Pero ahora el tablero debe ser un ArrayList bidimensional.
- **Actividad 5:** Tendrá 2 Ejércitos. Inicializar el tablero con n soldados aleatorios entre 1 y 10 para cada Ejército. Cada soldado tendrá un nombre autogenerado: Soldado0X1, Soldado1X1, etc., un valor de puntos de vida autogenerado aleatoriamente [1..5], la fila y columna también autogenerados aleatoriamente (no puede haber 2 soldados en el mismo cuadrado). Se debe mostrar el tablero con todos los soldados creados (distinguir los de un ejército de los del otro ejército). Además de los datos del Soldado con mayor vida de cada ejército, el promedio de puntos de vida de todos los soldados creados por ejército, los datos de todos los soldados por ejército en el orden que fueron creados y un ranking de poder de todos los soldados creados por ejército (del que tiene más nivel de vida al que tiene menos) usando 2 diferentes algoritmos de ordenamiento. Finalmente, que muestre qué ejército ganará la batalla (indicar la métrica usada para decidir al ganador de la batalla).

2. Equipos, materiales y temas utilizados

- Sistema Operativo Microsoft Windows 10 Pro 64 bits
- Visual Studio Code 1.82.2
- Java Development Kit 17.0.1
- Git 2.41.0.windows.1
- Windows PowerShell 5.1.19041.3031
- Cuenta en GitHub con el correo institucional.
- ArrayList bidimensional de objetos.
- Ordenamientos burbuja y por selección.

3. URL de Repositorio Github

- URL del Repositorio GitHub para clonar o recuperar.
- `https://github.com/cmestasz/fp2-23b.git`
- URL para el laboratorio 06 en el Repositorio GitHub.
- `https://github.com/cmestasz/fp2-23b/tree/main/fase01/lab06`

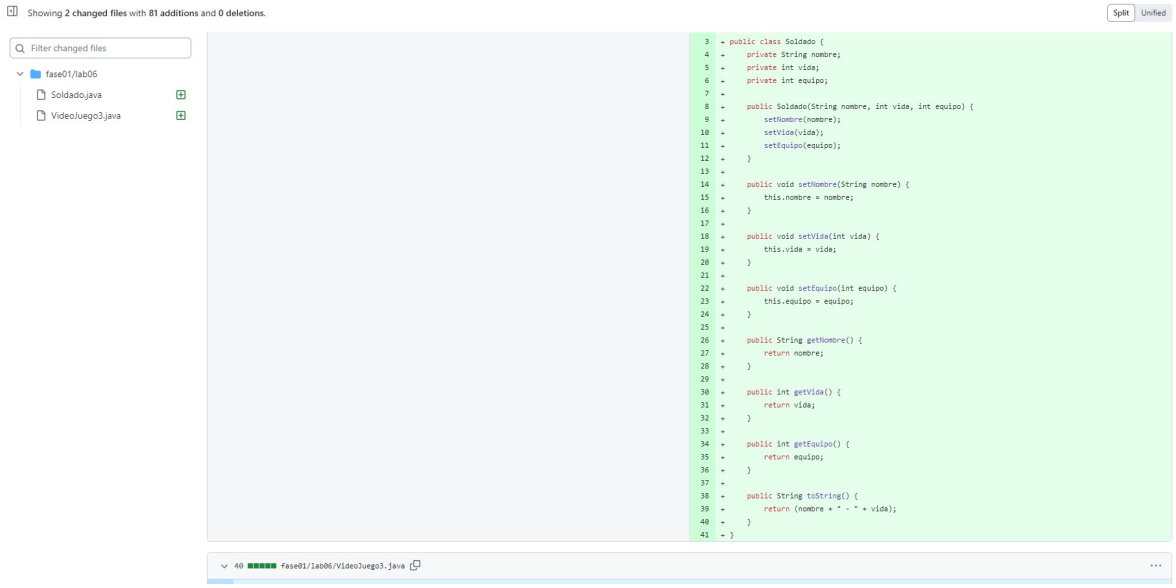
4. Actividades con el repositorio GitHub

Creando plantillas

```
$ mkdir lab05
$ cd lab05
$ code Soldado.java
$ code VideoJuego3.java
```

Primer Commit / Plantillas

```
$ git add .
$ git commit -m "Plantillas de archivos necesarios para resolver las actividades del
    laboratorio"
$ git push
```



Primer Commit.

Actualizando VideoJuego3.java

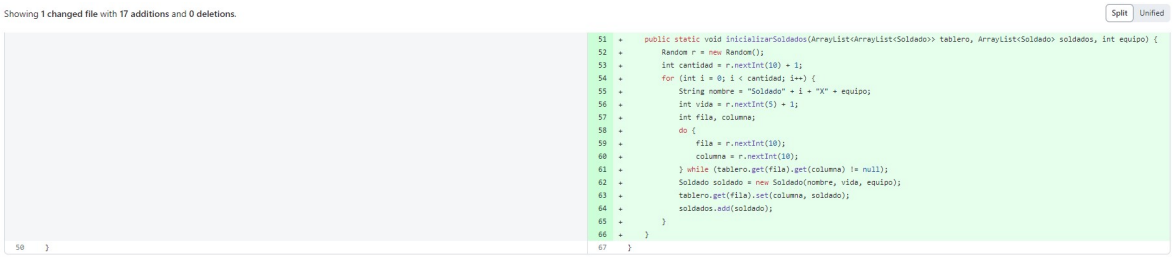
```
$ code VideoJuego3.java
```

Segundo - Decimo Segundo Commit / VideoJuego3.java

```
$ git add VideoJuego3.java
$ git commit -m "Metodo inicializarTablero()"
$ code VideoJuego3.java
$ git add VideoJuego3.java
$ git commit -m "Metodo inicializarSoldados()"
$ code VideoJuego3.java
$ git add VideoJuego3.java
$ git commit -m "Metodo imprimirTablero() y auxiliares"
$ code VideoJuego3.java
$ git add VideoJuego3.java
$ git commit -m "Metodo soldadoMayorVida()"
$ code VideoJuego3.java
$ git add VideoJuego3.java
$ git commit -m "Metodo promedioPuntosVida()"
$ code VideoJuego3.java
$ git add VideoJuego3.java
$ git commit -m "Metodo imprimirSoldados()"
$ code VideoJuego3.java
$ git add VideoJuego3.java
$ git commit -m "Metodo copiarArrayList()"
$ code VideoJuego3.java
$ git add VideoJuego3.java
$ git commit -m "Metodo ordenarSoldadosBurbuja() y auxiliares"
$ code VideoJuego3.java
$ git add VideoJuego3.java
$ git commit -m "Metodo ordenarSoldadosSeleccion()"
```

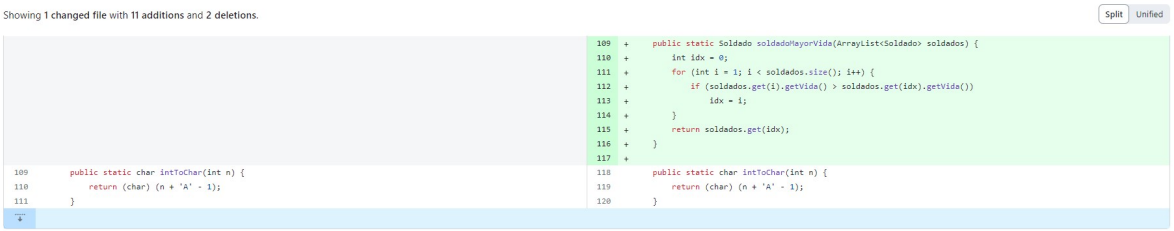
```
$ code VideoJuego3.java
$ git add VideoJuego3.java
$ git commit -m "Metodo obtenerGanador() y auxiliares"
$ code VideoJuego3.java
$ git add VideoJuego3.java
$ git commit -m "Correccion obtenerGanador() por imprimirGanador()"
$ git push
```

Segundo Commit.

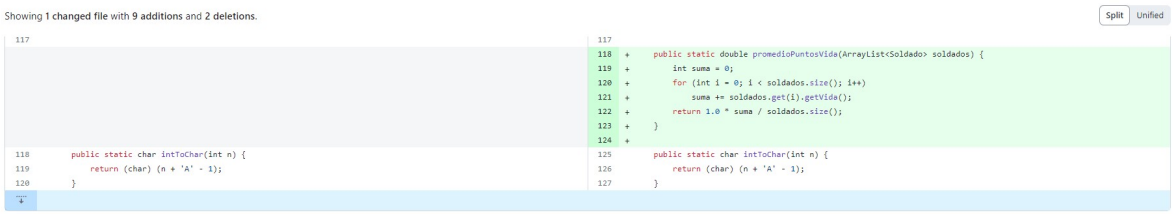


Tercer Commit.

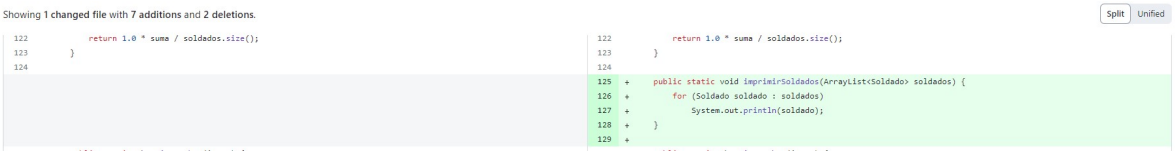
Cuarto Commit.



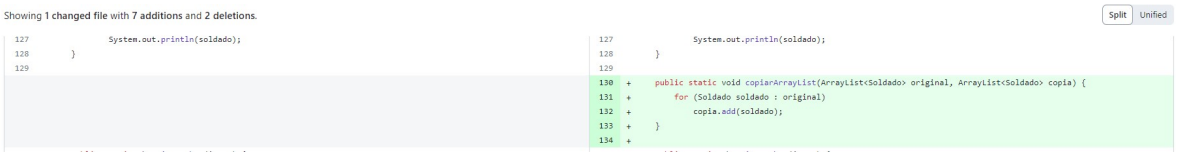
Quinto Commit.



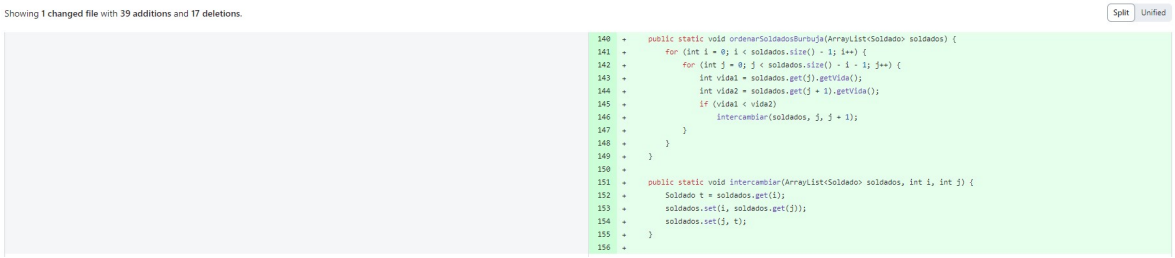
Sexto Commit.



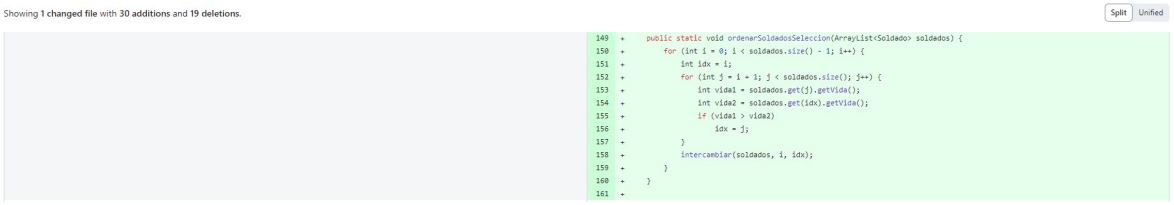
Séptimo Commit.



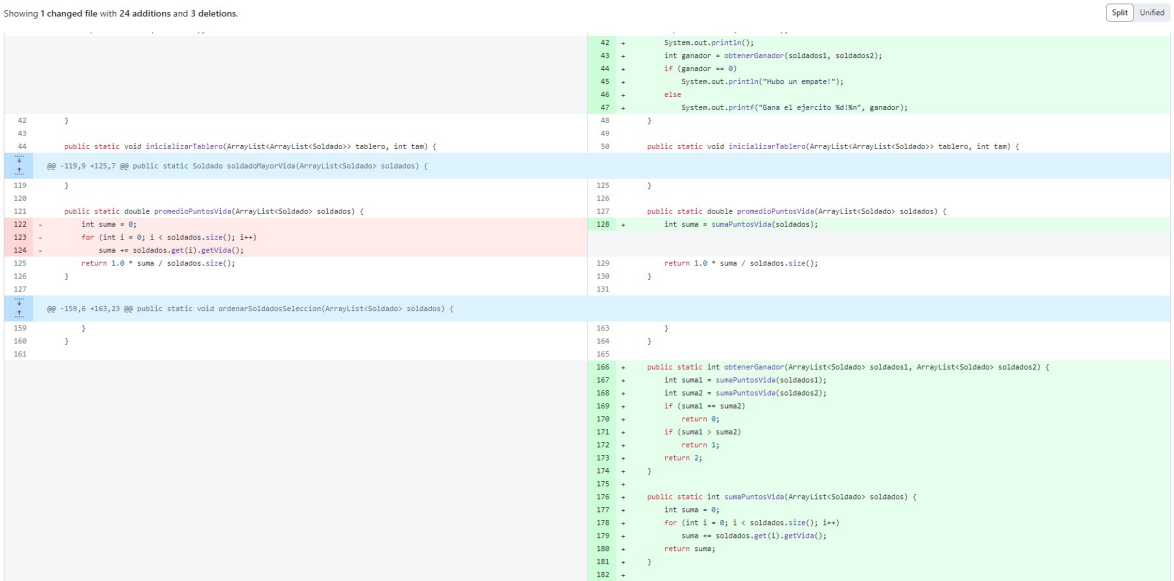
Octavo Commit.



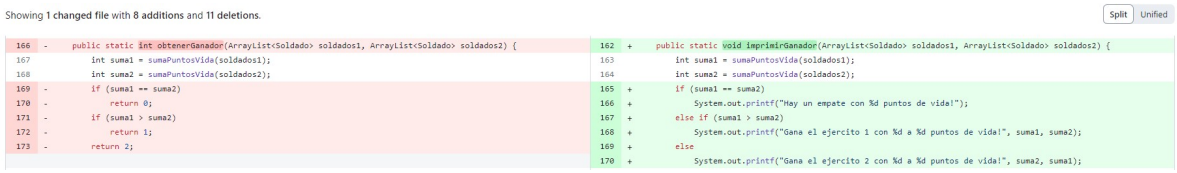
Noveno Commit.



Decimo Commit.



Decimo Primer Commit.



Showing 1 changed file with 8 additions and 11 deletions.

Split | Unified

```
166 -   public static int obtenerGanador(ArrayList<Soldado> soldados1, ArrayList<Soldado> soldados2) {
167             int suma1 = sumaPuntosVida(soldados1);
168             int suma2 = sumaPuntosVida(soldados2);
169 -           if (suma1 == suma2)
170 -               return 0;
171 -           if (suma1 > suma2)
172 -               return 1;
173 -           return 2;
```

```
162 +   public static void imprimirGanador(ArrayList<Soldado> soldados1, ArrayList<Soldado> soldados2) {
163             int suma1 = sumaPuntosVida(soldados1);
164             int suma2 = sumaPuntosVida(soldados2);
165 +           if (suma1 == suma2)
166 +               System.out.printf("Hay un empate con %d puntos de vida!");
167 +           else if (suma1 > suma2)
168 +               System.out.printf("Gana el ejercito 1 con %d a %d puntos de vida!", suma1, suma2);
169 +           else
170 +               System.out.printf("Gana el ejercito 2 con %d a %d puntos de vida!", suma2, suma1);
```

Decimo Segundo Commit.

5. Código desarrollado

Soldado.java

```java
package fase01.lab06;

public class Soldado {
    private String nombre;
    private int vida;
    private int equipo;

    public Soldado(String nombre, int vida, int equipo) {
        setNombre(nombre);
        setVida(vida);
        setEquipo(equipo);
    }

    public void setNombre(String nombre) {
        this.nombre = nombre;
    }

    public void setVida(int vida) {
        this.vida = vida;
    }

    public void setEquipo(int equipo) {
        this.equipo = equipo;
    }

    public String getNombre() {
        return nombre;
    }

    public int getVida() {
        return vida;
    }

    public int getEquipo() {
        return equipo;
    }

    public String toString() {
        return (nombre + " - " + vida);
    }
}
```

- Clase que guarda nombre y vida del soldado.
- Posee tanto setters como getters para todos los atributos.
- Posee el metodo toString() para poder imprimir el objeto.

VideoJuego3.java

```java
package fase01.lab06;

import java.util.Random;
import java.util.ArrayList;

public class VideoJuego3 {
    public static void main(String[] args) {
        ArrayList<ArrayList<Soldado>> tablero = new ArrayList<ArrayList<Soldado>>();
        ArrayList<Soldado> soldados1 = new ArrayList<Soldado>();
        ArrayList<Soldado> soldados2 = new ArrayList<Soldado>();
        inicializarTablero(tablero, 10);
        inicializarSoldados(tablero, soldados1, 1);
        inicializarSoldados(tablero, soldados2, 2);
        imprimirTablero(tablero);
        System.out.printf("Soldado con mayor vida del ejercito 1: %s%n",
            soldadoMayorVida(soldados1));
        System.out.printf("Promedio de puntos de vida del ejercito 1: %f%n",
            promedioPuntosVida(soldados1));
        imprimirSoldados(soldados1);
        ArrayList<Soldado> soldados1a = new ArrayList<Soldado>();
        ArrayList<Soldado> soldados1b = new ArrayList<Soldado>();
        copiarArrayList(soldados1, soldados1a);
        copiarArrayList(soldados1, soldados1b);
        System.out.println();
        ordenarSoldadosBurbuja(soldados1a);
        imprimirSoldados(soldados1a);
        System.out.println();
        ordenarSoldadosSeleccion(soldados1b);
        imprimirSoldados(soldados1b);
        System.out.println();
        System.out.printf("Soldado con mayor vida del ejercito 2: %s%n",
            soldadoMayorVida(soldados2));
        System.out.printf("Promedio de puntos de vida del ejercito 2: %f%n",
            promedioPuntosVida(soldados2));
        imprimirSoldados(soldados2);
        ArrayList<Soldado> soldados2a = new ArrayList<Soldado>();
        ArrayList<Soldado> soldados2b = new ArrayList<Soldado>();
        copiarArrayList(soldados2, soldados2a);
        copiarArrayList(soldados2, soldados2b);
        System.out.println();
        ordenarSoldadosBurbuja(soldados2a);
        imprimirSoldados(soldados2a);
        System.out.println();
        ordenarSoldadosSeleccion(soldados2b);
        imprimirSoldados(soldados2b);
        System.out.println();
        imprimirGanador(soldados1, soldados2);
    }

    public static void inicializarTablero(ArrayList<ArrayList<Soldado>> tablero, int tam) {
        for (int i = 0; i < tam; i++) {
            tablero.add(new ArrayList<Soldado>());
            for (int j = 0; j < tam; j++) {
                tablero.get(i).add(null);
            }
```

```
52          }
53      }
54
55      public static void inicializarSoldados(ArrayList<ArrayList<Soldado>> tablero,
            ArrayList<Soldado> soldados,
56              int equipo) {
57          Random r = new Random();
58          int cantidad = r.nextInt(10) + 1;
59          for (int i = 0; i < cantidad; i++) {
60              String nombre = "Soldado" + i + "X" + equipo;
61              int vida = r.nextInt(5) + 1;
62              int fila, columna;
63              do {
64                  fila = r.nextInt(10);
65                  columna = r.nextInt(10);
66              } while (tablero.get(fila).get(columna) != null);
67              Soldado soldado = new Soldado(nombre, vida, equipo);
68              tablero.get(fila).set(columna, soldado);
69              soldados.add(soldado);
70          }
71      }
72
73      public static void imprimirTablero(ArrayList<ArrayList<Soldado>> tablero) {
74          System.out.print(generarEncabezado(tablero));
75          String separacion = generarSeparacion(tablero);
76          for (int i = 0; i < tablero.size(); i++) {
77              System.out.print(separacion);
78              System.out.print(generarFila(tablero, i));
79          }
80          System.out.print(separacion);
81      }
82
83      public static String generarEncabezado(ArrayList<ArrayList<Soldado>> tablero) {
84          String encabezado = "\t";
85          for (int i = 0; i < tablero.size(); i++)
86              encabezado += (" " + intToChar(i + 1) + " ");
87          encabezado += " \n";
88          return encabezado;
89      }
90
91      public static String generarSeparacion(ArrayList<ArrayList<Soldado>> tablero) {
92          String fila = "\t";
93          for (int i = 0; i < tablero.get(0).size(); i++)
94              fila += "------";
95          fila += "-\n";
96          return fila;
97      }
98
99      public static String generarFila(ArrayList<ArrayList<Soldado>> tablero, int f) {
100         String fila = (f + 1) + "\t";
101         for (int i = 0; i < tablero.get(f).size(); i++) {
102             fila += "| ";
103             Soldado soldado = tablero.get(f).get(i);
104             if (soldado != null)
105                 fila += soldado.getNombre().substring(soldado.getNombre().length() - 3);
106             else
```

```
107             fila += "  ";
108         fila += " ";
109     }
110     fila += "|\n";
111     return fila;
112 }
113 
114 public static Soldado soldadoMayorVida(ArrayList<Soldado> soldados) {
115     int idx = 0;
116     for (int i = 1; i < soldados.size(); i++) {
117         if (soldados.get(i).getVida() > soldados.get(idx).getVida())
118             idx = i;
119     }
120     return soldados.get(idx);
121 }
122 
123 public static double promedioPuntosVida(ArrayList<Soldado> soldados) {
124     int suma = sumaPuntosVida(soldados);
125     return 1.0 * suma / soldados.size();
126 }
127 
128 public static void imprimirSoldados(ArrayList<Soldado> soldados) {
129     for (Soldado soldado : soldados)
130         System.out.println(soldado);
131 }
132 
133 public static void copiarArrayList(ArrayList<Soldado> original, ArrayList<Soldado> copia)
        {
134     for (Soldado soldado : original)
135         copia.add(soldado);
136 }
137 
138 public static void ordenarSoldadosBurbuja(ArrayList<Soldado> soldados) {
139     for (int i = 0; i < soldados.size() - 1; i++) {
140         for (int j = 0; j < soldados.size() - i - 1; j++) {
141             int vida1 = soldados.get(j).getVida();
142             int vida2 = soldados.get(j + 1).getVida();
143             if (vida1 < vida2)
144                 intercambiar(soldados, j, j + 1);
145         }
146     }
147 }
148 
149 public static void ordenarSoldadosSeleccion(ArrayList<Soldado> soldados) {
150     for (int i = 0; i < soldados.size() - 1; i++) {
151         int idx = i;
152         for (int j = i + 1; j < soldados.size(); j++) {
153             int vida1 = soldados.get(j).getVida();
154             int vida2 = soldados.get(idx).getVida();
155             if (vida1 > vida2)
156                 idx = j;
157         }
158         intercambiar(soldados, i, idx);
159     }
160 }
161 
```

```
162     public static void imprimirGanador(ArrayList<Soldado> soldados1, ArrayList<Soldado>
            soldados2) {
163         int suma1 = sumaPuntosVida(soldados1);
164         int suma2 = sumaPuntosVida(soldados2);
165         if (suma1 == suma2)
166             System.out.printf("Hay un empate con %d puntos de vida!");
167         else if (suma1 > suma2)
168             System.out.printf("Gana el ejercito 1 con %d a %d puntos de vida!", suma1, suma2);
169         else
170             System.out.printf("Gana el ejercito 2 con %d a %d puntos de vida!", suma2, suma1);
171     }
172
173     public static int sumaPuntosVida(ArrayList<Soldado> soldados) {
174         int suma = 0;
175         for (int i = 0; i < soldados.size(); i++)
176             suma += soldados.get(i).getVida();
177         return suma;
178     }
179
180     public static void intercambiar(ArrayList<Soldado> soldados, int i, int j) {
181         Soldado t = soldados.get(i);
182         soldados.set(i, soldados.get(j));
183         soldados.set(j, t);
184     }
185
186     public static char intToChar(int n) {
187         return (char) (n + 'A' - 1);
188     }
189 }
```

- Método inicializarTablero() crea un tablero de 10x10 completamente vacio.
- Método inicializarSoldados() crea a los soldados, los ubica en el tablero y los guarda en un arreglo de soldados por separado.
- Método imprimirTablero() imprime el tablero con ayuda de los métodos auxiliares generarEncabezado(), generarSeparacion() y generarFila(), ubicando a los soldados por su numero.
- Método soldadoMayorVida() retorna el soldado con mayor vida.
- Método promedioPuntosVida() retorna el promedio de los puntos de vida de todos los soldados.
- Método imprimirSoldados() imprime los soldados del ejército.
- Se crean dos copias del arreglo de soldados para demostrar los 2 ordenamientos.
- Método ordenarSoldadosBurbuja() ordena los soldados por vida de mayor a menor, usando ordenamiento burbuja.
- Método ordenarSoldadosSeleccion() ordena los soldados por vida de mayor a menor, usando ordenamiento selección.
- Se reusan los métodos para mostrar los datos de ambos ejércitos.
- Método imprimirGanador() imprime el ejército ganador de acuerdo a la cantidad total de puntos de vida de cada uno.

6. Ejecución del código

VideoJuego3.java

```
          A     B     C     D     E     F     G     H     I     J
        ------------------------------------------------------------
1       |     |     |     |     |     |     |     |     | 3X2 |     |
        ------------------------------------------------------------
2       | 8X1 |     |     |     |     |     |     |     |     |     |
        ------------------------------------------------------------
3       |     |     |     | 4X2 |     |     |     |     |     | 6X1 |
        ------------------------------------------------------------
4       |     |     |     |     |     |     |     |     |     |     |
        ------------------------------------------------------------
5       |     |     |     |     | 1X2 |     |     |     |     | 0X1 |
        ------------------------------------------------------------
6       |     | 2X2 |     |     |     | 6X2 |     |     |     |     |
        ------------------------------------------------------------
7       |     |     |     |     |     |     |     | 3X1 |     | 5X2 |
        ------------------------------------------------------------
8       |     |     | 4X1 |     |     |     | 8X2 | 1X1 |     |     |
        ------------------------------------------------------------
9       |     |     |     | 7X1 |     |     |     | 0X2 |     | 7X2 |
        ------------------------------------------------------------
10      |     | 2X1 |     |     |     |     |     |     | 5X1 |     |
        ------------------------------------------------------------
Soldado con mayor vida del ejercito 1: Soldado0X1 - 5
Promedio de puntos de vida del ejercito 1: 3.333333
Soldado0X1 - 5
Soldado1X1 - 5
Soldado2X1 - 2
Soldado3X1 - 2
Soldado4X1 - 2
Soldado5X1 - 5
Soldado6X1 - 4
Soldado7X1 - 4
Soldado8X1 - 1

Soldado0X1 - 5
Soldado1X1 - 5
Soldado5X1 - 5
Soldado6X1 - 4
Soldado7X1 - 4
Soldado2X1 - 2
Soldado3X1 - 2
Soldado4X1 - 2
Soldado8X1 - 1

Soldado0X1 - 5
Soldado1X1 - 5
Soldado5X1 - 5
Soldado6X1 - 4
Soldado7X1 - 4
Soldado2X1 - 2
Soldado3X1 - 2
Soldado4X1 - 2
```

```
Soldado8X1 - 1

Soldado con mayor vida del ejercito 2: Soldado4X2 - 5
Promedio de puntos de vida del ejercito 2: 2.777778
Soldado0X2 - 2
Soldado1X2 - 1
Soldado2X2 - 3
Soldado3X2 - 2
Soldado4X2 - 5
Soldado5X2 - 3
Soldado6X2 - 5
Soldado7X2 - 3
Soldado8X2 - 1

Soldado4X2 - 5
Soldado6X2 - 5
Soldado2X2 - 3
Soldado5X2 - 3
Soldado7X2 - 3
Soldado0X2 - 2
Soldado3X2 - 2
Soldado1X2 - 1
Soldado8X2 - 1

Soldado4X2 - 5
Soldado6X2 - 5
Soldado2X2 - 3
Soldado5X2 - 3
Soldado7X2 - 3
Soldado3X2 - 2
Soldado0X2 - 2
Soldado1X2 - 1
Soldado8X2 - 1

Gana el ejercito 1 con 30 a 25 puntos de vida!
```

7. Estructura de laboratorio 06

- El contenido que se entrega en este laboratorio es el siguiente:

```
lab06/
|--- Soldado.java
|--- VideoJuego3.java
|--- ejec01.bash
|--- Informe.tex
|--- Informe.pdf
|--- img
   |--- logo_abet.png
   |--- logo_episunsa.png
   |--- logo_unsa.jpg
   |--- commit01.jpg
   |--- commit02.jpg
   |--- commit03.jpg
   |--- commit04.jpg
   |--- commit05.jpg
   |--- commit06.jpg
   |--- commit07.jpg
   |--- commit08.jpg
   |--- commit09.jpg
   |--- commit10.jpg
   |--- commit11.jpg
   |--- commit12.jpg
```

8. Rúbricas

8.1. Entregable Informe

Tipo de Informe

Informe	
Latex	El informe está en formato PDF desde Latex, con un formato limpio (buena presentación) y facil de leer.

8.2. Rúbrica para el contenido del Informe y demostración

- El alumno debe marcar o dejar en blanco en celdas de la columna **Checklist** si cumplio con el ítem correspondiente.
- Si un alumno supera la fecha de entrega, su calificación será sobre la nota mínima aprobatoria, siempre y cuando cumpla con todos los items.
- El alumno debe autocalificarse en la columna **Estudiante** de acuerdo a la siguiente tabla:

Niveles de desempeño

	Nivel			
Puntos	Insatisfactorio 25 %	En Proceso 50 %	Satisfactorio 75 %	Sobresaliente 100 %
2.0	0.5	1.0	1.5	2.0
4.0	1.0	2.0	3.0	4.0

Rúbrica para contenido del Informe y demostración

Contenido y demostración		Puntos	Checklist	Estudiante	Profesor
1. GitHub	Hay enlace URL activo del directorio para el laboratorio hacia su repositorio GitHub con código fuente terminado y fácil de revisar.	2	X	2	
2. Commits	Hay capturas de pantalla de los commits más importantes con sus explicaciones detalladas. (El profesor puede preguntar para refrendar calificación).	4	X	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	1.5	
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 estan dentro de los plazos de fecha de entrega establecidos.	2	X	2	
7. Ortografía	El documento no muestra errores ortográficos.	2	X	1.5	
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	

9. Referencias

- Aedo, M. y Castro, E. (2021). FUNDAMENTOS DE PROGRAMACIÓN 2 - Tópicos de Programación Orientada a Objetos. Editorial UNSA.