

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
```

Showing 2 changed files with 81 additions and 0 deletions.

- fase01/lab06
 - Soldado.java
 - VideoJuego3.java

```

3 + public class Soldado {
4 +     private String nombre;
5 +     private int vida;
6 +     private int equipo;
7 +
8 +     public Soldado(String nombre, int vida, int equipo) {
9 +         setNombre(nombre);
10 +        setVida(vida);
11 +        setEquipo(equipo);
12 +    }
13 +
14 +    public void setNombre(String nombre) {
15 +        this.nombre = nombre;
16 +    }
17 +
18 +    public void setVida(int vida) {
19 +        this.vida = vida;
20 +    }
21 +
22 +    public void setEquipo(int equipo) {
23 +        this.equipo = equipo;
24 +    }
25 +
26 +    public String getNombre() {
27 +        return nombre;
28 +    }
29 +
30 +    public int getVida() {
31 +        return vida;
32 +    }
33 +
34 +    public int getEquipo() {
35 +        return equipo;
36 +    }
37 +
38 +    public String toString() {
39 +        return "Nombre = " + nombre + " Vida = " + vida;
40 +    }
41 + }

```

40 fase01/lab06/VideoJuego3.java

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
```

Showing 1 changed file with 10 additions and 0 deletions.

+	@@ -8,6 +8,7 @@ public static void main(String[] args) {	8	ArrayList<ArrayList<Soldado>> tablero = new ArrayList<ArrayList<Soldado>>();
8	ArrayList<ArrayList<Soldado>> tablero = new ArrayList<ArrayList<Soldado>>();	8	ArrayList<ArrayList<Soldado>> tablero = new ArrayList<ArrayList<Soldado>>();
9	ArrayList<Soldado> soldados1 = new ArrayList<Soldado>();	9	ArrayList<Soldado> soldados1 = new ArrayList<Soldado>();
10	ArrayList<Soldado> soldados2 = new ArrayList<Soldado>();	10	ArrayList<Soldado> soldados2 = new ArrayList<Soldado>();
11	inicializarSoldados(tablero, soldados1, 1);	11 +	inicializarTablero(tablero, 10);
12	inicializarSoldados(tablero, soldados2, 2);	12	inicializarSoldados(tablero, soldados1, 1);
13	imprimirTablero(tablero);	13	inicializarSoldados(tablero, soldados2, 2);
		14	imprimirTablero(tablero);
+	@@ -37,4 +38,13 @@ public static void main(String[] args) {	38	ordenarSoldadosSeleccion(soldados2b);
37	ordenarSoldadosSeleccion(soldados2b);	39	imprimirSoldados(soldados2b);
38	imprimirSoldados(soldados2b);	40	}
39	}	41 +	public static void inicializarTablero(ArrayList<ArrayList<Soldado>> tablero, int tam) {
		42 +	for (int i = 0; i < tam; i++) {
		43 +	tablero.add(new ArrayList<Soldado>());
		44 +	for (int j = 0; j < tam; j++) {
		45 +	tablero.get(i).add(null);
		46 +	}
		47 +	}
		48 +	}
40	}	49 +	}
		50	}

Segundo Commit.

Showing 1 changed file with 17 additions and 0 deletions.

51 +	public static void inicializarSoldados(ArrayList<ArrayList<Soldado>> tablero, ArrayList<Soldado> soldados, int equipo) {
52 +	Random r = new Random();
53 +	int cantidad = r.nextInt(10) + 1;
54 +	for (int i = 0; i < cantidad; i++) {
55 +	String nombre = "Soldado" + i + " " + "X" + equipo;
56 +	int vida = r.nextInt(5) + 1;
57 +	int fila, columna;
58 +	do {
59 +	fila = r.nextInt(10);
60 +	columna = r.nextInt(10);
61 +	} while (tablero.get(fila).get(columna) != null);
62 +	Soldado soldado = new Soldado(nombre, vida, equipo);
63 +	tablero.get(fila).set(columna, soldado);
64 +	soldados.add(soldado);
65 +	}
66 +	}
67	}

Tercer Commit.

Showing 1 changed file with 47 additions and 2 deletions.

```

68 + public static void imprimirTablero(ArrayList<ArrayList<Soldado>> tablero) {
69 +     System.out.println(generarEncabezado(tablero));
70 +     String separacion = generarSeparacion(tablero);
71 +     for (int i = 0; i < tablero.size(); i++) {
72 +         System.out.println(separacion);
73 +         System.out.print(generarFila(tablero, i));
74 +     }
75 +     System.out.println(separacion);
76 + }
77 +
78 + public static String generarEncabezado(ArrayList<ArrayList<Soldado>> tablero) {
79 +     String encabezado = "\n";
80 +     for (int i = 0; i < tablero.size(); i++)
81 +         encabezado += (" " + Integer.toString(i + 1) + " ");
82 +     encabezado += "\n";
83 +     return encabezado;
84 + }
85 +
86 + public static String generarSeparacion(ArrayList<ArrayList<Soldado>> tablero) {
87 +     String fila = "\n";
88 +     for (int i = 0; i < tablero.get(0).size(); i++)
89 +         fila += "-----";
90 +     fila += "\n";
91 +     return fila;
92 + }
93 +
94 + public static String generarFila(ArrayList<ArrayList<Soldado>> tablero, int f) {
95 +     String fila = "(" + f + ") = \n";
96 +     for (int i = 0; i < tablero.get(f).size(); i++) {
97 +         fila += " | ";
98 +         Soldado soldado = tablero.get(f).get(i);
99 +         if (soldado != null)
100 +             fila += soldado.getNombre().substring(soldado.getNombre().length() - 3);
101 +         else
102 +             fila += " ";
103 +         fila += " | ";
104 +     }
105 +     fila += "\n";
106 +     return fila;
107 + }
108 +
109 + public static char intoChar(int n) {
110 +     return (char) (n + 'A' - 1);
111 + }
112 + }

```

Cuarto Commit.

Showing 1 changed file with 11 additions and 2 deletions.

```

109 + public static Soldado soldadoMayorVida(ArrayList<Soldado> soldados) {
110 +     int idx = 0;
111 +     for (int i = 1; i < soldados.size(); i++) {
112 +         if (soldados.get(i).getVida() > soldados.get(idx).getVida())
113 +             idx = i;
114 +     }
115 +     return soldados.get(idx);
116 + }
117 +
118 + public static char intoChar(int n) {
119 +     return (char) (n + 'A' - 1);
120 + }

```

Quinto Commit.

Showing 1 changed file with 9 additions and 2 deletions.

```

117 +
118 + public static double promedioPuntosVida(ArrayList<Soldado> soldados) {
119 +     int suma = 0;
120 +     for (int i = 0; i < soldados.size(); i++)
121 +         suma += soldados.get(i).getVida();
122 +     return 1.0 * suma / soldados.size();
123 + }
124 +
125 + public static char intoChar(int n) {
126 +     return (char) (n + 'A' - 1);
127 + }

```

Sexto Commit.

Showing 1 changed file with 7 additions and 2 deletions.

```

122 +     return 1.0 * suma / soldados.size();
123 + }
124 +
125 + public static void imprimirSoldados(ArrayList<Soldado> soldados) {
126 +     for (Soldado soldado : soldados)
127 +         System.out.println(soldado);
128 + }
129 + }

```

Séptimo Commit.

Showing 1 changed file with 7 additions and 2 deletions.

```
127         System.out.println(soldado);
128     }
129 }
```

```
127         System.out.println(soldado);
128     }
129 }
130 + public static void copiaArray(ArrayList<Soldado> original, ArrayList<Soldado> copia) {
131 +     for (Soldado soldado : original)
132 +         copia.add(soldado);
133 + }
134 + }
```

Octavo Commit.

Showing 1 changed file with 39 additions and 17 deletions.

```
140 + public static void ordenarSoldadosBurbuja(ArrayList<Soldado> soldados) {
141 +     for (int i = 0; i < soldados.size() - 1; i++) {
142 +         for (int j = 0; j < soldados.size() - i - 1; j++) {
143 +             int v1 = soldados.get(j).getVida();
144 +             int v2 = soldados.get(j + 1).getVida();
145 +             if (v1 < v2)
146 +                 intercambiar(soldados, j, j + 1);
147 +         }
148 +     }
149 + }
150 +
151 + public static void intercambiar(ArrayList<Soldado> soldados, int i, int j) {
152 +     Soldado t = soldados.get(i);
153 +     soldados.set(i, soldados.get(j));
154 +     soldados.set(j, t);
155 + }
156 + }
```

Noveno Commit.

Showing 1 changed file with 39 additions and 19 deletions.

```
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 v1 = soldados.get(j).getVida();
154 +             int v2 = soldados.get(idx).getVida();
155 +             if (v1 > v2)
156 +                 idx = j;
157 +         }
158 +         intercambiar(soldados, i, idx);
159 +     }
160 + }
161 + }
```

Decimo Commit.

Showing 1 changed file with 24 additions and 3 deletions.

```
42         System.out.println();
43         int ganador = obtenerGanador(soldados1, soldados2);
44         if (ganador == 0)
45             System.out.println("Hubo un empate!");
46         else
47             System.out.printf("Gana el ejército %d!%n", ganador);
48     }
49 }
50 public static void inicializarTablero(ArrayList<ArrayList<Soldado>> tablero, int tam) {
51     @@ -119,9 +125,7 @@ public static Soldado soldadoMayorVida(ArrayList<Soldado> soldados) {
52 }
53 }
54 public static double promedioPuntosVida(ArrayList<Soldado> soldados) {
55     int suma = 0;
56     for (int i = 0; i < soldados.size(); i++)
57         suma += soldados.get(i).getVida();
58     return 1.0 * suma / soldados.size();
59 }
60 }
61 @@ -159,6 +163,23 @@ public static void ordenarSoldadosSeleccion(ArrayList<Soldado> soldados) {
62 }
63 }
64 }
65 }
66 + public static int obtenerGanador(ArrayList<Soldado> soldados1, ArrayList<Soldado> soldados2) {
67 +     int suma1 = sumaPuntosVida(soldados1);
68 +     int suma2 = sumaPuntosVida(soldados2);
69 +     if (suma1 == suma2)
70 +         return 0;
71 +     if (suma1 > suma2)
72 +         return 1;
73 +     return 2;
74 + }
75 +
76 + public static int sumaPuntosVida(ArrayList<Soldado> soldados) {
77 +     int suma = 0;
78 +     for (int i = 0; i < soldados.size(); i++)
79 +         suma += soldados.get(i).getVida();
80 +     return suma;
81 + }
82 + }
```

Decimo Primer Commit.

Showing 1 changed file with 8 additions and 11 deletions.

Split Unified

<pre>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; </pre>	<pre>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); </pre>
--	---

Decimo Segundo Commit.

5. Código desarrollado

Soldado.java

```
1 package fase01.lab06;
2
3 public class Soldado {
4     private String nombre;
5     private int vida;
6     private int equipo;
7
8     public Soldado(String nombre, int vida, int equipo) {
9         setNombre(nombre);
10        setVida(vida);
11        setEquipo(equipo);
12    }
13
14    public void setNombre(String nombre) {
15        this.nombre = nombre;
16    }
17
18    public void setVida(int vida) {
19        this.vida = vida;
20    }
21
22    public void setEquipo(int equipo) {
23        this.equipo = equipo;
24    }
25
26    public String getNombre() {
27        return nombre;
28    }
29
30    public int getVida() {
31        return vida;
32    }
33
34    public int getEquipo() {
35        return equipo;
36    }
37
38    public String toString() {
39        return (nombre + " - " + vida);
40    }
41 }
```

- 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

```
1 package fase01.lab06;
2
3 import java.util.Random;
4 import java.util.ArrayList;
5
6 public class VideoJuego3 {
7     public static void main(String[] args) {
8         ArrayList<ArrayList<Soldado>> tablero = new ArrayList<ArrayList<Soldado>>();
9         ArrayList<Soldado> soldados1 = new ArrayList<Soldado>();
10        ArrayList<Soldado> soldados2 = new ArrayList<Soldado>();
11        inicializarTablero(tablero, 10);
12        inicializarSoldados(tablero, soldados1, 1);
13        inicializarSoldados(tablero, soldados2, 2);
14        imprimirTablero(tablero);
15        System.out.printf("Soldado con mayor vida del ejercito 1: %s%n",
16                           soldadoMayorVida(soldados1));
17        System.out.printf("Promedio de puntos de vida del ejercito 1: %f%n",
18                           promedioPuntosVida(soldados1));
19        imprimirSoldados(soldados1);
20        ArrayList<Soldado> soldados1a = new ArrayList<Soldado>();
21        ArrayList<Soldado> soldados1b = new ArrayList<Soldado>();
22        copiarArrayList(soldados1, soldados1a);
23        copiarArrayList(soldados1, soldados1b);
24        System.out.println();
25        ordenarSoldadosBurbuja(soldados1a);
26        imprimirSoldados(soldados1a);
27        System.out.println();
28        ordenarSoldadosSeleccion(soldados1b);
29        imprimirSoldados(soldados1b);
30        System.out.println();
31        System.out.printf("Soldado con mayor vida del ejercito 2: %s%n",
32                           soldadoMayorVida(soldados2));
33        System.out.printf("Promedio de puntos de vida del ejercito 2: %f%n",
34                           promedioPuntosVida(soldados2));
35        imprimirSoldados(soldados2);
36        ArrayList<Soldado> soldados2a = new ArrayList<Soldado>();
37        ArrayList<Soldado> soldados2b = new ArrayList<Soldado>();
38        copiarArrayList(soldados2, soldados2a);
39        copiarArrayList(soldados2, soldados2b);
40        System.out.println();
41        ordenarSoldadosBurbuja(soldados2a);
42        imprimirSoldados(soldados2a);
43        System.out.println();
44        ordenarSoldadosSeleccion(soldados2b);
45        imprimirSoldados(soldados2b);
46        System.out.println();
47        imprimirGanador(soldados1, soldados2);
48    }
49
50    public static void inicializarTablero(ArrayList<ArrayList<Soldado>> tablero, int tam) {
51        for (int i = 0; i < tam; i++) {
52            tablero.add(new ArrayList<Soldado>());
53            for (int j = 0; j < tam; j++) {
54                tablero.get(i).add(null);
55            }
56        }
57    }
58 }
```

```
52     }
53 }
54
55 public static void inicializarSoldados(ArrayList<ArrayList<Soldado>> tablero,
56     ArrayList<Soldado> soldados,
57     int equipo) {
58     Random r = new Random();
59     int cantidad = r.nextInt(10) + 1;
60     for (int i = 0; i < cantidad; i++) {
61         String nombre = "Soldado" + i + "X" + equipo;
62         int vida = r.nextInt(5) + 1;
63         int fila, columna;
64         do {
65             fila = r.nextInt(10);
66             columna = r.nextInt(10);
67         } while (tablero.get(fila).get(columna) != null);
68         Soldado soldado = new Soldado(nombre, vida, equipo);
69         tablero.get(fila).set(columna, soldado);
70         soldados.add(soldado);
71     }
72 }
73
74 public static void imprimirTablero(ArrayList<ArrayList<Soldado>> tablero) {
75     System.out.print(generarEncabezado(tablero));
76     String separacion = generarSeparacion(tablero);
77     for (int i = 0; i < tablero.size(); i++) {
78         System.out.print(separacion);
79         System.out.print(generarFila(tablero, i));
80     }
81     System.out.print(separacion);
82 }
83
84 public static String generarEncabezado(ArrayList<ArrayList<Soldado>> tablero) {
85     String encabezado = "\t";
86     for (int i = 0; i < tablero.size(); i++)
87         encabezado += (" " + intToChar(i + 1) + " ");
88     encabezado += " \n";
89     return encabezado;
90 }
91
92 public static String generarSeparacion(ArrayList<ArrayList<Soldado>> tablero) {
93     String fila = "\t";
94     for (int i = 0; i < tablero.get(0).size(); i++)
95         fila += "-----";
96     fila += "-\n";
97     return fila;
98 }
99
100 public static String generarFila(ArrayList<ArrayList<Soldado>> tablero, int f) {
101     String fila = (f + 1) + "\t";
102     for (int i = 0; i < tablero.get(f).size(); i++) {
103         fila += "| ";
104         Soldado soldado = tablero.get(f).get(i);
105         if (soldado != null)
106             fila += soldado.getNombre().substring(soldado.getNombre().length() - 3);
107         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     {
135         for (Soldado soldado : original)
136             copia.add(soldado);
137     }
138
139 public static void ordenarSoldadosBurbuja(ArrayList<Soldado> soldados) {
140     for (int i = 0; i < soldados.size() - 1; i++) {
141         for (int j = 0; j < soldados.size() - i - 1; j++) {
142             int vida1 = soldados.get(j).getVida();
143             int vida2 = soldados.get(j + 1).getVida();
144             if (vida1 < vida2)
145                 intercambiar(soldados, j, j + 1);
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 vacío.
- 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 número.
- 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 ordenarSoldadosSelección() 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

```

1      A    B    C    D    E    F    G    H    I    J
2      -----
3  1      |    |    |    |    |    |    |    |    | 3X2 |    |
4      -----
5  2      | 8X1 |    |    |    |    |    |    |    |    |
6      -----
7  3      |    |    |    | 4X2 |    |    |    |    |    | 6X1 |
8      -----
9  4      |    |    |    |    |    |    |    |    |    |    |
10     -----
11  5      |    |    |    |    | 1X2 |    |    |    |    | 0X1 |
12     -----
13  6      |    | 2X2 |    |    |    | 6X2 |    |    |    |    |
14     -----
15  7      |    |    |    |    |    |    |    | 3X1 |    | 5X2 |
16     -----
17  8      |    |    | 4X1 |    |    |    | 8X2 | 1X1 |    |    |
18     -----
19  9      |    |    |    | 7X1 |    |    |    | 0X2 |    | 7X2 |
20     -----
21 10      |    | 2X1 |    |    |    |    |    |    | 5X1 |    |
22     -----
23 Soldado con mayor vida del ejercito 1: Soldado0X1 - 5
24 Promedio de puntos de vida del ejercito 1: 3.333333
25 Soldado0X1 - 5
26 Soldado1X1 - 5
27 Soldado2X1 - 2
28 Soldado3X1 - 2
29 Soldado4X1 - 2
30 Soldado5X1 - 5
31 Soldado6X1 - 4
32 Soldado7X1 - 4
33 Soldado8X1 - 1
34
35 Soldado0X1 - 5
36 Soldado1X1 - 5
37 Soldado5X1 - 5
38 Soldado6X1 - 4
39 Soldado7X1 - 4
40 Soldado2X1 - 2
41 Soldado3X1 - 2
42 Soldado4X1 - 2
43 Soldado8X1 - 1
44
45 Soldado0X1 - 5
46 Soldado1X1 - 5
47 Soldado5X1 - 5
48 Soldado6X1 - 4
49 Soldado7X1 - 4
50 Soldado2X1 - 2
51 Soldado3X1 - 2
52 Soldado4X1 - 2

```

```
53 Soldado8X1 - 1
54
55 Soldado con mayor vida del ejercito 2: Soldado4X2 - 5
56 Promedio de puntos de vida del ejercito 2: 2.777778
57 Soldado0X2 - 2
58 Soldado1X2 - 1
59 Soldado2X2 - 3
60 Soldado3X2 - 2
61 Soldado4X2 - 5
62 Soldado5X2 - 3
63 Soldado6X2 - 5
64 Soldado7X2 - 3
65 Soldado8X2 - 1
66
67 Soldado4X2 - 5
68 Soldado6X2 - 5
69 Soldado2X2 - 3
70 Soldado5X2 - 3
71 Soldado7X2 - 3
72 Soldado0X2 - 2
73 Soldado3X2 - 2
74 Soldado1X2 - 1
75 Soldado8X2 - 1
76
77 Soldado4X2 - 5
78 Soldado6X2 - 5
79 Soldado2X2 - 3
80 Soldado5X2 - 3
81 Soldado7X2 - 3
82 Soldado3X2 - 2
83 Soldado0X2 - 2
84 Soldado1X2 - 1
85 Soldado8X2 - 1
86
87 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

Puntos	Nivel			
	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.