Informe de Laboratorio 12

Tema: Definición de Clases de Usuario Clase Soldado - Menú

Nota

Estudiante	Escuela	Asignatura
Julio Rubén Chura Acabana jchuraaca@unsa.edu.pe	Escuela Profesional de Ingeniería de Sistemas	F. de Programción 2 Semestre: I Código: 20230472

Laboratorio	Tema	Duración
12	Definición de Clases de Usuario Clase Soldado - Menú	04 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2023 - B	Del 6 de Diciembre 2023	Al 11 de Diciembre 2023

1. Tarea

- Al comenzar el videojuego, se presentan dos opciones principales: "Juego rápido"para partidas inmediatas con la capacidad de repetir o regresar al menú principal, y "Juego personalizado"que implica la gestión de dos ejércitos con soldados individualizados. Los soldados tienen atributos específicos, y las acciones incluyen la creación, eliminación, clonación y modificación de soldados, así como la comparación y el intercambio entre ellos. Se puede visualizar información detallada y sumar niveles usando "Method-Call Chaining". Tras efectuar cambios, se ofrece la posibilidad de iniciar el juego con las mismas opciones disponibles al finalizar, con la opción de regresar al menú principal en cualquier momento. Este diseño permite a los jugadores una experiencia estratégica y personalizada con opciones flexibles en la gestión de ejércitos antes de entrar en la acción del juego.
- Usted debe realizar varios commits y al término de la actividad deberá realizar un informe.

2. Equipos, materiales y temas utilizados

- Sistema Operativo Windows
- vim 9.0
- OpenJDK 64-Bits 20.0.2.
- Git 2.42.0.
- Cuenta en GitHub con el correo institucional.
- POO- Clases de Usuario

3. URL de Repositorio Github

- URL del Repositorio GitHub para clonar o recuperar.
- `https://github.com/JulioChura/fp2-23b.git`
- URL para el laboratorio 01 en el Repositorio GitHub.
- `https://github.com/JulioChura/fp2-23b/tree/main/fase02/lab12`

4. Actividades con el repositorio GitHub

4.1. Preparación del espacio de trabajo

Listing 1: Se crea la carpeta de laboratorio 12 y se copian los archivos del lab07 al lab12

```
mkdir lab12
cd lab07
Copy-Item "Soldier.java" -Destination "..\lab12"
Copy-Item "VideoJuego5.java" -Destination "..\lab12\VideoJuego.java"
cd ..
cd lab12
code VideoJuego5.java
```

Listing 2: Commit: 1eaab04e81dc9e3fefb9e8c2eb3828b197354723

```
git add Soldier.java
git commit -m "Se sube Soldier al lab12"
git push -u origin main
```

Listing 3: Commit: 3c1a953b3b0e88c13f5ed99d538f64928469608b

```
git add VideoJuego.java
git commit -m "Se copia VideoJuego del Lab07 al lab12"
git push -u origin main
```

4.2. Métodos de laboratorios pasados que no han sido alterados

```
1    InsertionSort(arr[], n)
2        for i = 1 to n-1
3        key = arr[i]
4        j = i-1
5        while j >= 0 and arr[j] > key
6            arr[j+1] = arr[j]
7            j = j-1
8        arr[j+1] = key
```

```java
// Se hará uso del ordenamiento por inserción
public static Soldier longerLife(ArrayList<Soldier> s) {
    int n = s.size();
    for (int i = 1; i < n; i++) {
        Soldier key = s.get(i);
        int j = i - 1;
        while (j >= 0 && s.get(j).getLifePoints() > key.getLifePoints()) {
            s.set(j + 1, s.get(j));
            j--;
        }
        s.set(j + 1, key);
    }
    return s.get(s.size() - 1);
}
```

- longerLife(): Se adapta el pseudocódigo de insertion y se le da la forma para que se aplique en un ArrayList. Lo que hace el método es ordenar de menor a mayor, por lo que último elemento es el mayor. El método devuelve al Soldier que se encuentra en esa posición.
- Se muestra el pseudocódigo, pero aplicado a un arreglo Estándar. A pesar de ser otra estructura, la idea es la misma

```java
public static void showByCreation(ArrayList<Soldier> sol) {
    for (Soldier n : sol) {
        System.out.println(n);
    }
}
```

- showByCreation(): Básicamente imprime los datos de un ArrayList unidimensional de tipo Soldier.

```java
public static int totalLife(ArrayList<Soldier> sol) {
    int addition = 0;
    for (Soldier n : sol) {
        addition = addition + n.getLifePoints();
    }
    return addition;
}
```

- totalLife(): El método devuelve los puntos de vida total del ejército. Se decide hacer el método con retorno para poder usar esos valores para determinar al ganador y calcular el promedio.

```java
    // Por condiciones del problema se solicitan dos metodos de ordenamiento, por lo
    // que este seria el segundo. Se usara el burbuja.
    public static void orderByPower(ArrayList<Soldier> sol) {
        boolean swapped;
        Soldier temp;
        for (int i = 0; i < sol.size() - 1; i++) {
            swapped = false;
            for (int j = 0; j < sol.size() - 1 - i; j++) {
                temp = sol.get(j);
                sol.set(j, sol.get(j + 1));
                sol.set(j + 1, temp);
                swapped = true;
            }
            if (swapped == false) {
                break;
            }
        }
        for (Soldier n : sol) {
            System.out.println(n);
        }
    }
```

```
Procedimiento bubbleSort(entero arr[], entero n)
Inicio
    entero i, j
    booleano swapped
    Para i = 0 Hasta n - 1 Hacer
        swapped = falso
        Para j = 0 Hasta n - i - 1 Hacer
            Si arr[j] > arr[j + 1] Entonces
                intercambiar(arr[j], arr[j + 1])
                swapped = verdadero
            Fin Si
        Fin Para
        Si swapped == falso Entonces
            Romper
        Fin Si
    Fin Para
Fin
```

- orderByPower(): El método ordena el ArrayList tomando como criterio los puntos de Vida. No se retorna nada
- Se muestra el pseudocódigo, pero aplicado a un arreglo Estándar. A pesar de ser otra estructura, la idea es la misma.

```java
//Pregunta al jugador si quiere jugar una ronda
public static boolean validation() {
    do {
        System.out.println(x:"Desea jugar una ronda?(si/no)");
        String answer = sc.next();
        if (answer.equalsIgnoreCase(anotherString:"Si")) {
            return true;
        } else if (answer.equalsIgnoreCase(anotherString:"No")) {
            return false;
        } else {
            System.out.println(x:"Respuesta no admsible");
        }
    } while (true);

}
```

- validation(): El método pregunta al usuario si desea jugar una ronda.

```
function busqueda_binaria (datos, valor_busqueda):
    inicio = 0
    fin = longitud (datos) - 1
    mientras inicio <= fin:
        medio = (inicio + fin) // 2
        si datos [medio] == valor_busqueda:
            devolver medio
        sino si datos [medio] < valor_busqueda:
            inicio = medio + 1
        sino:
            fin = medio - 1
    devolver -1
```

- Este es el pseudocódigo de búsqueda binaria.

```java
public static void binarySearchByName(ArrayList<Soldier> armyA, String name) {
    Collections.sort(armyA, new Comparator<Soldier>() {
        public int compare(Soldier soldier1, Soldier soldier2) {
            return soldier1.getName().compareTo(soldier2.getName());
        }
    });
    int high = armyA.size() - 1;
    int low = 0;
    while (low <= high) {
        int mid = (high + low) / 2;
        Soldier soldier = armyA.get(mid);
        if (name.equalsIgnoreCase(soldier.getName())) {
            System.out.println("Se ha encontrado: " + soldier);
            return;
        } else if (name.compareTo(soldier.getName()) < 0) {
            high = mid - 1;
        } else {
            low = mid + 1;
        }
    }
    System.out.println(x:"No fue encontrado");
}
```

- La búsqueda binaria será aplicada para buscar por nombres dentro del ArrayList .

4.3. Modificación de la clase Soldier

Listing 4: Se han aumentado atributos y se crearon tres sobreconstructores sobrecargads

```
code Soldier.java
```

```java
public class Soldier {

    private String name;
    private int row;
    private int column;
    private int attackLevel;
    private int defenseLevel;
    private int actualLife;
    private int speed;
    private String attitude;
    private boolean current;

    public Soldier() {
        this(name:"NotFound", row:0, column:0, attackLevel:0, defenseLevel:0,
        actualLife:0, speed:0, attitude:"repose", current:true);
    }

    public Soldier(String name, int row, int column, int attackLevel, int defenseLevel, int
        actualLife, int speed) {
        this(name, row, column, attackLevel, defenseLevel, actualLife, speed:0,
        attitude:"Repose", current:true);
    }

    public Soldier(String name, int row, int column, int attackLevel, int defenseLevel, int actualLife,
            int speed, String attitude, boolean current) {
        this.name = name;
        this.row = row;
        this.column = column;
        this.attackLevel = attackLevel;
        this.defenseLevel = defenseLevel;
        this.actualLife = actualLife;
        this.speed = speed;
        this.attitude = attitude;
        this.current = current;
    }
```

- En una condición de la actividad se nos pedirá usar el método Call Chaining, por lo que al crear un soldier este se iniciará con atributos ya establecidos y ya nos permitirá hacer acciones como sumar los attackLevel por ejemplo. Nos servirá a manera de acumulador.
- El segundo es para crear un soldier estableciendo sus atributos desde el main. Este se usará más
- El tercero es como plantilla el cual contiene todos los atributos de la clase soldier

Listing 5: Commit: 59579c7cc3df93a5c893c22138b0b929f99d597e

```
git add Soldier.java
git commit -m "Se crean los 3 constructores"
git push -u origin main
```

Listing 6: Se crean los métodos advance, attack, defend, back, die, beAttacked, scape, getActualLife, setActualLife

```
code Soldier.java
```

```
36      public void advance() {
37          speed++;
38      }
39
40      public void attack() {
41          attitude = "offensive";
42          advance();
43      }
44
45      public void defend() {
46          attitude = "defensive";
47          speed = 0;
48      }
49
50      public void back() {
51          if (speed > 0) {
52              speed = 0;
53              attitude = "defensive";
54          } else if (speed == 0) {
55              speed--;
56          }
57      }
58
59      public void die() {
60          if (actualLife < 0) {
61              current = false;
62          } else {
63              current = true;
64          }
65      }
```

```
67      public void beAttacked(int damage) {
68          this.actualLife = this.actualLife - damage;
69          die();
70      }
71
72      public void scape() {
73          this.speed = this.speed + 2;
74      }
75
76      public int getActualLife() {
77          return actualLife;
78      }
79
80      public void setActualLife(int actualLife) {
81          this.actualLife = actualLife;
82      }
```

```
108     public int getAttackLevel() {
109         return attackLevel;
110     }
111
112     public void setAttackLevel(int attackLevel) {
113         this.attackLevel = attackLevel;
114     }
115
116     public int getDefenseLevel() {
117         return defenseLevel;
118     }
119
120     public void setDefenseLevel(int defenseLevel) {
121         this.defenseLevel = defenseLevel;
122     }
```

```
124     public int getSpeed() {
125         return speed;
126     }
127
128     public void setSpeed(int speed) {
129         this.speed = speed;
130     }
131
132     public boolean getCurrent() {
133         return current;
134     }
```

- Estos métodos, a pesar de no tener una gran utilidad dentro del videojuego (porque la métrica no se ajusta a los métodos creados), se nos exige para este laboratorio, además otros se crean con el fin de poder realizar el método de suma de atributos que se verá más adelante .

Listing 7: Commit: 28c2163ac9e7a02e2c83549533267a8c22e28547

```
git add Soldier.java
git commit -m "Se termina de crear todos los getters y setters"
git push -u origin main
```

Listing 8: Se crea el método que clonará la mayoría de atributos de un soldier

```
code Soldier.java
```

```
136     public Soldier clone() {
137         Soldier clon = new Soldier();
138         clon.name = this.name;
139         clon.attackLevel = this.attackLevel;
140         clon.defenseLevel = this.defenseLevel;
141         clon.actualLife = this.actualLife;
142         clon.speed = this.speed;
143         clon.attitude = this.attitude;
144         clon.current = this.current;
145         clon.row = this.row;
146         clon.column = this.column;
147         return clon;
148     }
```

- Este método clona la mayoría de atributos del soldier que queramos, sin embargo, no se copia los atributos de row y column porque son atributos únicos ya que son la posición del soldier.

Listing 9: Commit: 614e5b5aae38cf2232af5edb122c0db671195ef5

```
git add Soldier.java
git commit -m "Metodo que clona un soldier"
git push -u origin main
```

Listing 10: El siguiente método comparará atributos

```
code Soldier.java
```

```
149     public void compareTo(Soldier s) {
150         if (this.attackLevel == s.getAttackLevel()) {
151             System.out.println(x:"Tienen igual nivel de ataque");
152         } else {
153             System.out.println(x:"Tienen distintos niveles de ataque");
154         }
155         if (this.defenseLevel == s.getDefenseLevel()) {
156             System.out.println(x:"Tiene igual nivel de defensa");
157         } else {
158             System.out.println(x:"Tienen diferente nivel de defensa");
159         }
160         if (this.actualLife == s.getActualLife()) {
161             System.out.println(x:"Su vida actual de vida es actual");
162         } else {
163             System.out.println(x:"Su vida actual son diferentes");
164         }
165         if (this.current && s.getCurrent()) {
166             System.out.println(x:"Ambos viven");
167         } else {
168             System.out.println(x:"El estado de si vive o no es distinto");
169         }
170         if (this.speed == s.getSpeed()) {
171             System.out.println(x:"Tienen igual velocidad");
172         } else {
173             System.out.println(x:"Tienen diferentes velocidades");
174         }
175     }
```

Listing 11: Commit: c1eefb734f5354d04097855759507371f7c12d5c

```
git add Soldier.java
git commit -m "Se crea el metodo comparateTo"
git push -u origin main
```

Listing 12: Se genera el soldado más poderoso

```
code Soldier.java
```

```java
public Soldier mightySoldier(Soldier s) {
    Soldier sol = new Soldier();
    this.actualLife = this.actualLife + s.getActualLife();
    this.attackLevel = this.attackLevel + s.getAttackLevel();
    this.defenseLevel = this.defenseLevel + s.getDefenseLevel();
    this.speed = this.speed + s.getSpeed();

    setActualLife(this.actualLife);
    setAttackLevel(this.attackLevel);
    setDefenseLevel(this.defenseLevel);
    setSpeed(this.speed);
    return sol;
}
```

- Este método genera un Soldier con atributos en base a la suma de otros Soldier.

Listing 13: Commit: f813202b9c72f5cefe894fcf22abc6c6f57a3a14

```
git add Soldier.java
git commit -m "Se corrige el metodo mightySoldier"
git push -u origin main
```

Listing 14: Se implementa el método toString

```
code Soldier.java
```

```java
@Override
public String toString() {
    return "Soldier [name=" + name + ", row=" + row + ", column=" + column + ", attackLevel=" + attackLevel
            + ", defenseLevel=" + defenseLevel + ", actualLife=" + actualLife + ", speed=" + speed + ", attitude="
            + attitude + ", current=" + current + "]";
}
```

Listing 15: Commit: 256779ffb9e686cb971dc63fb7c6ac7fadf36ee1

```
git add Soldier.java
git commit -m "Se implementa el toString"
git push -u origin main
```

4.4. Editando VideoJuego.java

Listing 16: Se crea un metodo que brinda detalles del enfrentamiento

```
code VideoJuego.java
```

```
328     public static void stagesOfWar(ArrayList<ArrayList<Soldier>> armyA,
329             ArrayList<ArrayList<Soldier>> armyB, ArrayList<Soldier> armyAU, ArrayList<Soldier> armyBU) {
330
331         System.out.println(x:"ooooooooooooooo  FASE 1 DE LA CONTIENDA  ooooooooooooooo");
332         System.out.println("Mostrando estadisticas de cada ejercito" + "\n");
333         System.out.println(x:"DATOS DEL DEL EJERCITO A");
334         showByCreation(armyAU);
335         System.out.println(x:"DATOS DEL EJRCITO B");
336         showByCreation(armyBU);
337         System.out.println();
338
339         System.out.println(x:"ooooooooooooooo  FASE 2 DE LA CONTIENDA  ooooooooooooooo");
340         System.out.println(x:"Mostrando el tablero de juego");
341         myBoard(armyA, armyB);
342         System.out.println();
343
344         System.out.println(x:"++++++++++++++++   FASE 3 DE LA CONTIENDA  ++++++++++++++++");
345
346     }
```

- Este fragmento de código no es algo nuevo, ya estaba en el main. Ahora se decide incorporarlo a un método para poder rehusarlo ya que el programa nos pedirá iniciar una partida en dos opciones y ya no en una sola como se vino haciendo antes.

Listing 17: Commit: 5818ab3f0158ea7f297ac2415bb32b86f8057596

```
git add VideoJuego.java
git commit -m "Metodo stagesOfWar pulido"
git push -u origin main
```

Listing 18: Se crean arreglos por referencia

```
code VideoJuego.java
```

```
79                      do {
80                          System.out.println(x:"Elija el ejercito que editara:\n1: Ejercito A\n2: Ejercito B");
81                          option = sc.nextInt();
82                          if (option == 1) {
83                              copyB = armyA;
84                              copyU = armyAU;
85                              anotherCopyB = armyB;
86                              anotherCopyU = armyBU;
87                              break;
88                          } else if (option == 2) {
89                              copyB = armyB;
90                              copyU = armyBU;
91                              anotherCopyB = armyA;
92                              anotherCopyU = armyAU;
93                              break;
94
95                          } else {
96                              System.out.println(x:"Opcion no valida!");
97                              continue;
98                          }
99
100                     } while (true);
```

- En el main, hasta este punto se realizó un bosquejo del flujo del programa, sin embargo, lo más importante es este fragmento de código el cual nos permite inicializar ArrayList usando referencia, esto con el fin de poder realizar cambios en los ArrayList originales.

Listing 19: Commit: 5ae36c4b0202868075638aae95cc8a9a36f36118

```
git add VideoJuego.java
git commit -m "Se crean arreglos por referencia"
git push -u origin main
```

Listing 20: Metodo que trasnforma ArrayList Bidimensioanles a Unidimensionales

```
code VideoJuego.java
```

```java
// Transforma un ArrayList Bidimensional a Unidimensional
public static ArrayList<Soldier> arrayListUnidimensional(ArrayList<ArrayList<Soldier>> s) {
    ArrayList<Soldier> armyUni = new ArrayList<Soldier>();
    for (int i = 0; i < s.size(); i++) {
        for (int j = 0; j < s.get(i).size(); j++) {
            if (s.get(i).get(j) != null) {
                armyUni.add(s.get(i).get(j));
            }
        }
    }
    return armyUni;
}
```

- Este método sirve para poder trabajar de manera más sencilla con otros métodos como binarySearch entre otros.

Listing 21: Se crea un método que iniciliza elementos de un ArrayList con nulls

```
code VideoJuego.java
```

```java
// Genera un ArrayList bidimensional pero vacio
public static ArrayList<ArrayList<Soldier>> armyEmpty() {
    ArrayList<ArrayList<Soldier>> army = new ArrayList<>(initialCapacity:10);
    for (int i = 0; i < 10; i++) {
        army.add(new ArrayList<>(Collections.nCopies(n:10, o:null)));
    }
    return army;
}
```

- Este método genera llena ArrayList con nulls con el fin de servir como apoyo al momento de inicializar los ArrayList con objetos del tipo Soldier sin generar repeticiones.

Listing 22: Commit: 55bd8cf312296a9a1727dc08824c0de7936e5060

```
git add VideoJuego.java
git commit -m "Metodo que genera un ArrayList bidimensional vacio"
git push -u origin main
```

Listing 23: Método que transforma ArrayList bidimensioanles en unidimensionales

```
code VideoJuego.java
```

```java
// Transforma un ArrayList Bidimensional a Unidimensional
public static ArrayList<Soldier> arrayListUnidimensional(ArrayList<ArrayList<Soldier>> s) {
    ArrayList<Soldier> armyUni = new ArrayList<Soldier>();
    for (int i = 0; i < s.size(); i++) {
        for (int j = 0; j < s.get(i).size(); j++) {
            if (s.get(i).get(j) != null) {
                armyUni.add(s.get(i).get(j));
            }
        }
    }
    return armyUni;
}
```

- arrayListUnidimensional: A pesar de ya contar con dos ArrayList bidimensionales, se opta por transformarlos en ArrayList unidimensionales ya que nos facilitará trabajar con los demás métodos que la práctica de laboratorio solicita. En el main se hace la creación de dos ArrayList unidimensionales tanto para el ejército A y B. Este método ya se venía presentando en laboratorios anteriores, por lo que no sufrió alteraciones.

Listing 24: Se mejora el método que permite iniciliazar los ArrayList con objetos de tipo soldier.

```
code VideoJuego.java
```

```java
//Inicializa ArrayList con objetos de tipo Soldier
public static ArrayList<ArrayList<Soldier>> generateArmy(ArrayList<ArrayList<Soldier>> a,
ArrayList<ArrayList<Soldier>> b) {

    Random random = new Random();
    int amount = random.nextInt(bound:10) + 1;
    int n = 0;
    int  row, column;
    int lifePoints;
    String name;
    do {
        row = random.nextInt(ROW_BOARD);
        column = random.nextInt(COLUMN_BOARD);
        if (a.get(row).get(column) == null && b.get(row).get(column) == null) {
            name = "Soldier" + (row + 1) + "x" + (column + 1);
            lifePoints = random.nextInt(MAX_VALUE) + 1;
            int attack = random.nextInt(MAX_VALUE) + 1;
            int defense = random.nextInt(MAX_VALUE) + 1;
            Soldier sol = new Soldier(name, row + 1, column + 1, attack, defense, lifePoints, speed:0);
            a.get(row).set(column, sol);
            n++;
        }
    } while (n < amount);
    return a;
}
```

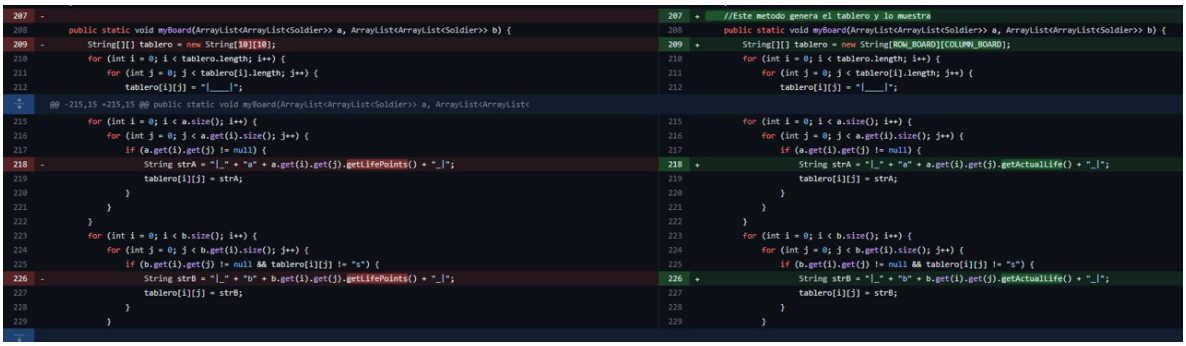
- Este método genera objetos de tipo Soldier bien definidos y los guarda en un ArrayList. Se usan constantes de clases para generar los atributos de cada soldier. Básicamente, para el llenado de objetos para ambos ejércitos, la primera vez que se lo invoca, se pasan como parámetros dos ArrayList vacíos, para generarel segundo ejército se le manda un ArrayList vacío y el que hemos llenado con el uso de este método (el primer ArrayLis que se llenó).

Listing 25: Commit: d4f69f7ac1e9eaddbdca6dae2b5b253ebfcce269

```
git add VideoJuego.java
git commit -m "Se mejora el metodo generateArmy"
git push -u origin main
```

Listing 26: Se hacen cambios en el método myBoard

```
code VideoJuego.java
```



- La lógica con la que el tablero se genera y se muestra no se alteró, sin embargo, se optó por usar las constantes de clases para evitar "números fantasmas", además se cambia el método getLifePoints por getActualLifePoints.

```java
// Este metodo genera el tablero y lo muestra
public static void myBoard(ArrayList<ArrayList<Soldier>> a, ArrayList<ArrayList<Soldier>> b) {
    String[][] tablero = new String[ROW_BOARD][COLUMN_BOARD];
    for (int i = 0; i < tablero.length; i++) {
        for (int j = 0; j < tablero[i].length; j++) {
            tablero[i][j] = "|____|";
        }
    }
    for (int i = 0; i < a.size(); i++) {
        for (int j = 0; j < a.get(i).size(); j++) {
            if (a.get(i).get(j) != null) {
                String strA = "|_" + "a" + a.get(i).get(j).getActualLife() + "_|";
                tablero[i][j] = strA;
            }
        }
    }
    for (int i = 0; i < b.size(); i++) {
        for (int j = 0; j < b.get(i).size(); j++) {
            if (b.get(i).get(j) != null && tablero[i][j] != "s") {
                String strB = "|_" + "b" + b.get(i).get(j).getActualLife() + "_|";
                tablero[i][j] = strB;
            }
        }
    }
    System.out.print(s:"   A     B     C     D     E    F     G     H     I     J \n");
    for (int i = 0; i < tablero.length; i++) {
        System.out.printf(format:"%2d", (i + 1));
        for (int j = 0; j < tablero[i].length; j++) {
            System.out.print(tablero[i][j]);
        }
        System.out.println();
    }
}
```

- Aquí se muestra el método myBoard de forma completa.

Listing 27: Commit: a0f41647ec8dad5414cdd80eeac576c145bc032f

```
git add VideoJuego.java
git commit -m "Se modifican algunas cosas dentro del tablero"
git push -u origin main
```

Listing 28: Se implementa el método que recibe una coordenada

```
code VideoJuego.java
```

```java
// Se encarga de devolver la coordenada del soldado que se quiere mover
public static int[] coordinate(ArrayList<ArrayList<Soldier>> army) {
    int[] coordinate = new int[2];

    do {

        System.out.println(x:"Ingrese las coordenadas");
        String position = sc.next().toLowerCase();
        int row;
        int column;
        if (position.length() == 2) {
            row = Integer.parseInt(position.substring(beginIndex:0, endIndex:1));
            column = position.charAt(index:1) - 'a' + 1;
        } else if (position.length() == 3) {
            row = Integer.parseInt(position.substring(beginIndex:0, endIndex:2));
            column = position.charAt(index:2) - 'a' + 1;
        } else if (position.equalsIgnoreCase(anotherString:"Salir")) {
            coordinate[0] = -1;
            coordinate[1] = -1;
            return coordinate;
        } else {
            System.out.println(x:"Coordenda no válida");
            continue;
        }
        if (row >= 1 && row <= 10 && column <= 10 && column >= 1 && army.get(row - 1).get(column - 1) != null) {
            coordinate[0] = row - 1;
            coordinate[1] = column - 1;
            return coordinate;
        }
    } while (true);
}
```

- Este método sirve para que el usuario ingrese una coordenda (por ejemplo 1A), desglosando así el String en números enteros que representan la fila y columna (Ejm: 0 y 0 es el resultado de ingresar 1A, dichos valores son procesados por los demás métodos).

Listing 29: Commit: 0350a4047a04c3909a8829c31ee143472164c509

```
git add VideoJuego.java
git commit -m "Se crea el coordinate"
git push -u origin main
```

Listing 30: Se implementa el método que da una dirección de movimiento

```
code VideoJuego.java
```

```java
/*
 * Se encarga de validar la nueva posicion del soldier retornando asi la nueva
 * coordenada (Solo se retorna dos enteros, siendo row y column)
 */
public static int[] newCoordinateToMove(int row, int column, ArrayList<ArrayList<Soldier>> attack) {
    int[] rowAndColumn = new int[2];
    do {
        System.out.println(x:"Hacia donde quiere mover? up(1), down(2), left(3), right(4), exit(5) ");
        int direction = sc.nextInt();
        int newRowPosition = row;
        int newColumnPosition = column;

        switch (direction) {
            case 1: // arriba
                newRowPosition--;
                break;
            case 2: // abajo
                newRowPosition++;
                break;
            case 3: // izquierda
                newColumnPosition--;
                break;
            case 4: // derecha
                newColumnPosition++;
                break;
            case 5:
                rowAndColumn[0] = -1;
                rowAndColumn[1] = -1;
                return rowAndColumn;
            default:
                System.out.println(x:"Dirección no válida");
                break;
        }

        if (newRowPosition < 0 || newRowPosition >= 10 || newColumnPosition < 0 || newColumnPosition >= 10) {
            System.out.println(x:"Movimiento fuera del tablero");
            continue;
        }
        if (attack.get(newRowPosition).get(newColumnPosition) != null) {
            System.out.println(x:"No se puede mover a una posición ocupada por un soldado del mismo bando");
            continue;
        }
        rowAndColumn[0] = newRowPosition;
        rowAndColumn[1] = newColumnPosition;
        return rowAndColumn;
    } while (true);

}
```

- Este método sirve para que el soldier pueda moverse a una determinada posición (arriba, abajo, izquierda, derecha). Retorna un arreglo que contiene la nueva coordenada donde el soldier deberá moverse.

Listing 31: Commit: ae376e9c6b501f2d37a8979eb6f488a5b4ce4890

```
git add VideoJuego.java
git commit -m "Se crea el metodo newCoordinateToMove"
git push -u origin main
```

Listing 32: Se implementa el método que mueve al Soldier a otra casilla

```
code VideoJuego.java
```

```java
    // Hace que el soldier se mueve a otra casilla
    public static boolean moveSoldier(ArrayList<ArrayList<Soldier>> attack, ArrayList<ArrayList<Soldier>> repose,
            String A) {
        int[] coordinates = coordinate(attack);
        int rowPosition = coordinates[0];
        int columnPosition = coordinates[1];

        // Se verifica que el usuario quiere o no seguir en el juego
        if (rowPosition == -1) {
            return false;
        }
        Soldier attackingSoldier = attack.get(rowPosition).get(columnPosition);
        int[] rowAndColumn = newCoordinateToMove(rowPosition, columnPosition, attack);
        int newRowPosition = rowAndColumn[0];
        int newColumnPosition = rowAndColumn[1];

        if (newRowPosition == -1) {
            return false;
        }

        Soldier targetSoldier = repose.get(newRowPosition).get(newColumnPosition);
        targetSoldier.advance();
        if (targetSoldier != null) {

            int actualLifeRepose = targetSoldier.getActualLife();
            int actualLifeAAttack = attackingSoldier.getActualLife();
            double totalLife = actualLifeAAttack + actualLifeRepose;
            double probabilityAttack = 100 * actualLifeAAttack / totalLife;

            double random = Math.random() * 101;
            System.out.println("Estadisticas de batalla:\tSoldado de atacante: " + probabilityAttack
                    + "%\tSoldado en reposo: " + Math.abs((100 - probabilityAttack)) + "\tSalio como aleatorio: "
                    + random);
            attackingSoldier.attack();
            targetSoldier.attack();
            System.out.println(targetSoldier + " \n " + attackingSoldier);
            if (random < probabilityAttack) {
                repose.get(newRowPosition).set(newColumnPosition, element:null);
                attack.get(newRowPosition).set(newColumnPosition, attackingSoldier);
                System.out.println(targetSoldier);
                System.out.println(x:"Ha ganado el soldado del ejército atacante");
            } else if (random > probabilityAttack) {
                System.out.println(x:"Ha ganado el soldado en reposo");
                System.out.println(attackingSoldier);
            } else {
                repose.get(newRowPosition).set(newColumnPosition, element:null);
                System.out.println(x:"EMPATE, ambos son destruidos");
            }

        } else {
            attack.get(newRowPosition).set(newColumnPosition, attackingSoldier);
            System.out.println(x:"El soldado se ha movido.");
        }
        attack.get(rowPosition).set(columnPosition, element:null);
        return true;

    }
}
```

- Este método usa los métodos coordinate y newCoordinateToMove para hacer el desplazamiento corespondiente, para ello tenemos dos casos principales. El primero es que el soldier a mover no tenga cruces con un soldier del equipo rival, el segundo caso es que haya cruce y por ende se ocasione una batalla y en esta parte se implementa la lógica de enfrentamiento, la cual consiste en generar un número aleatorio que debe estar entre el rango de la probabilidad que tiene cada soldier de ganar que se calcula en base de la vida actual, por ejemplo si el Soldier A tiene 2 y el Soldier B tiene 4, la probabilidad de A es 33.3 y la de B es de 66.6, por lo que si el aletorio es de 27.7, ganaría el ejército A.

Listing 33: Commit: aa83f83f4aac63d6a92bb824f0dab2411cbb886f

```
git add VideoJuego.java
git commit -m "Se crea el metodo moveSoldier"
git push -u origin main
```

Listing 34: Se implementa el método que verifica si queda algún objeto dentro de un ArrayList

```
code VideoJuego.java
```

```
492     // Se encarga de revisar si un ejercito se ha quedado sin
493     public static boolean isEmpty(ArrayList<ArrayList<Soldier>> a) {
494         for (ArrayList<Soldier> row : a) {
495             for (Soldier soldier : row) {
496                 if (soldier != null) {
497                     return false;
498                 }
499             }
500 
501         }
502         return true;
503     }
```

- Este método se encarga de verificar si aún quedan objetos dentro del ArrayList, más adelante se complementará con otros métodos que dictaminarán al ganador de la contienda.

Listing 35: Commit: ab3d57e656981530cd4e74610edf81c1cc600f95

```
git add VideoJuego.java
git commit -m "Se crea el metodo isEmpty"
git push -u origin main
```

Listing 36: Se implementa el método que determina al ganador

```
code VideoJuego.java
```

```
505     // Determina al ganador de la contienda
506     public static boolean winnerBattle(ArrayList<ArrayList<Soldier>> a, ArrayList<ArrayList<Soldier>> b) {
507         boolean emptyInA = isEmpty(a);
508         boolean emptyInB = isEmpty(b);
509         if (emptyInA && emptyInB) {
510             System.out.println(x:"EMPATE");
511             return false;
512         } else if (emptyInA) {
513             System.out.println(x:"Ha ganado B");
514             return false;
515         } else if (emptyInB) {
516             System.out.println(x:"Ha ganado A");
517             return false;
518         }
519         return true;
520     }
```

- Este método se encarga de verificar si aún quedan objetos dentro del ArrayList, más adelante se complementará con otros método que dictaminarán al ganador de la contienda.

Listing 37: Commit: 04134f5051779fbd5b19cbea7b9a076d56d46d90

```
git add VideoJuego.java
git commit -m "Se crea el metodo el metodo winnerBattle"
git push -u origin main
```

Listing 38: Se implementa el método para crear un Soldier

```
code VideoJuego.java
```

```java
// Case 1: Crear soldier.
public static void createSoldier(ArrayList<ArrayList<Soldier>> army, ArrayList<Soldier> armyU,
        ArrayList<ArrayList<Soldier>> enemy) {
    int amount = armyU.size();
    int row, column, attack, defense, life, speed;
    String name;
    if (amount <= 9) {
        do {
            System.out.println(x:" Ingrese la fila (ejm 1,2,3..10)");
            row = sc.nextInt() - 1;
            System.out.println(x:"Ingrese la columna (ejm 1,2,3...10)");
            column = sc.nextInt() - 1;
            if (army.get(row).get(column) == null && enemy.get(row).get(column) == null) {
                System.out.println(x:"Ingrese el nivel de ataque");
                attack = sc.nextInt();

                System.out.println(x:"Ingrese el nivel de defensa");
                defense = sc.nextInt();

                System.out.println(x:"Ingrese el nivel de vida");
                life = sc.nextInt();

                System.out.println(x:"Ingrese la velocidad");
                speed = sc.nextInt();

                name = "Soldier" + (row + 1) + "X" + (column + 1);

                Soldier newSoldier = new Soldier(name, row + 1, column + 1, attack, defense, life, speed);
                army.get(row).set(column, newSoldier);

                armyU.add(newSoldier);
                break;
            }
        } while (true);
    } else {
        System.out.println(x:"Este ejército sobrepasó la cantidad permitida");
    }
}
```

- Este método se encarga de crear un nuevo Soldier sin que sobrepase la capacidad máxima establecida.

Listing 39: Compilando y probando

```
javac VideoJuego.java
java VideoJuego
1: Juego rapido
```

```
2: Juego personalizado
3: Salir
2
DATOS DEL EJRCITO A
Soldier [name=Soldier2x6, lifePoints=, row=2, column=6]
Soldier [name=Soldier8x7, lifePoints=, row=8, column=7]
DATOS DEL EJRCITO B
Soldier [name=Soldier2x4, lifePoints=, row=2, column=4]
Soldier [name=Soldier3x9, lifePoints=, row=3, column=9]
Soldier [name=Soldier5x7, lifePoints=, row=5, column=7]
Soldier [name=Soldier6x7, lifePoints=, row=6, column=7]
Soldier [name=Soldier8x8, lifePoints=, row=8, column=8]
Soldier [name=Soldier9x3, lifePoints=, row=9, column=3]
1: Crear soldado
2: Eliminar soldado
3: Clonar soldado
4: Modificar soldado
5: Comparar Soldado
6: Intercambiar soldado
7: Ver soldado
8: Ver ejercito
9: Sumar niveles
10: Jugar
11: Volver al menu principal
1
Elija el ejercito que editara:
1: Ejercito A
2: Ejercito B
1
Ingrese la fila (ejm 1,2,3..10)
1
Ingrese la columna (ejm 1,2,3...10)
1
Ingrese el nivel de ataque
5
Ingrese el nivel de defensa
5
Ingrese el nivel de vida
5
Ingrese la velocidad
5
DATOS DEL EJRCITO A
Soldier [name=Soldier2x6, lifePoints=, row=2, column=6]
Soldier [name=Soldier8x7, lifePoints=, row=8, column=7]
Soldier [name=Soldier1X1, lifePoints=, row=1, column=1]
DATOS DEL EJRCITO B
Soldier [name=Soldier2x4, lifePoints=, row=2, column=4]
Soldier [name=Soldier3x9, lifePoints=, row=3, column=9]
Soldier [name=Soldier5x7, lifePoints=, row=5, column=7]
Soldier [name=Soldier6x7, lifePoints=, row=6, column=7]
Soldier [name=Soldier8x8, lifePoints=, row=8, column=8]
Soldier [name=Soldier9x3, lifePoints=, row=9, column=3]
1: Crear soldado
2: Eliminar soldado
3: Clonar soldado
4: Modificar soldado
```

```
5: Comparar Soldado
6: Intercambiar soldado
7: Ver soldado
8: Ver ejercito
9: Sumar niveles
10: Jugar
11: Volver al menu principal
11
1: Juego rapido
2: Juego personalizado
3: Salir
3
```

- Se está probando el menú y la opción de crear Soldier.

Listing 40: Commit: 0802334d8d151230f14cdf4a1e05ecbccf6f5738

```
git add VideoJuego.java
git commit -m "Se crea el metodo createSoldier"
git push -u origin main
```

Listing 41: Se implementa el método para eliminar un Soldier

```
code VideoJuego.java
```

```java
// Este metodo elimina un soldado
public static void removeSoldier(ArrayList<ArrayList<Soldier>> copyB, ArrayList<Soldier> copyU) {
    showByCreation(copyU);
    int row, column;
    int[] position;
    int size = copyU.size();
    if (size == 1) {
        return;
    }
    do {
        position = coordinate(copyB);
        row = position[0];
        column = position[1];
        Soldier temporary = copyB.get(row).get(column);
        if (temporary != null) {
            copyB.get(row).set(column, element:null);
            copyU.remove(temporary);
            break;
        } else {
            System.out.println(x:"No existe ningun soldado en esa posicion!");
            continue;
        }
    } while (true);

}
```

- Este método se encarga de borrar un Soldier, la acción no se ejecutará si el ejército tiene como máximo un elemento.

Listing 42: Compilando y probando el método removeSoldier

```
javac VideoJuego.java
java VideoJuego
1: Juego rapido
2: Juego personalizado
3: Salir
2
DATOS DEL EJRCITO A
Soldier [name=Soldier10x7, lifePoints=, row=10, column=7]
DATOS DEL EJRCITO B
Soldier [name=Soldier5x7, lifePoints=, row=5, column=7]
Soldier [name=Soldier8x3, lifePoints=, row=8, column=3]
1: Crear soldado
2: Eliminar soldado
3: Clonar soldado
4: Modificar soldado
5: Comparar Soldado
6: Intercambiar soldado
7: Ver soldado
8: Ver ejercito
9: Sumar niveles
10: Jugar
11: Volver al menu principal
2
Elija el ejercito que editara:
1: Ejercito A
2: Ejercito B
2
Debe Eliminar un soldado
   A     B      C    D    E     F    G     H    I    J
1 |____||____||____||____||____||____||____||____||____|
2 |____||____||____||____||____||____||____||____||____|
3 |____||____||____||____||____||____||____||____||____|
4 |____||____||____||____||____||____||____||____||____|
5 |____||____||____||____||____||____||_b2_||____||____|
6 |____||____||____||____||____||____||____||____||____|
7 |____||____||____||____||____||____||____||____||____|
8 |____||____||_b2_||____||____||____||____||____||____|
9 |____||____||____||____||____||____||____||____||____|
10|____||____||____||____||____||____||_a2_||____||____|
Soldier [name=Soldier5x7, lifePoints=, row=5, column=7]
Soldier [name=Soldier8x3, lifePoints=, row=8, column=3]
Ingrese las coordenadas
8c
DATOS DEL EJRCITO A
Soldier [name=Soldier10x7, lifePoints=, row=10, column=7]
DATOS DEL EJRCITO B
Soldier [name=Soldier5x7, lifePoints=, row=5, column=7]
1: Crear soldado
2: Eliminar soldado
3: Clonar soldado
4: Modificar soldado
5: Comparar Soldado
6: Intercambiar soldado
7: Ver soldado
8: Ver ejercito
9: Sumar niveles
```

```
10: Jugar
11: Volver al menu principal
11
1: Juego rapido
2: Juego personalizado
3: Salir
3
```

Listing 43: Commit: 28886c9cee9a351810f14da8769e1c922a221904

```
git add VideoJuego.java
git commit -m "Se crea el metodo removeSoldier"
git push -u origin main
```

Listing 44: Se implementa el método para clonar un soldado

```
code VideoJuego.java
```

```java
// Este metodo clona un soldado tal cual
public static void cloneSoldier(ArrayList<ArrayList<Soldier>> army, ArrayList<Soldier> armyUni) {
    System.out.println(x:"Ingrese el soldado a clonar");
    int[] coordinates;
    int row, column, i, j;
    int size = armyUni.size();
    Soldier clonedSoldier;
    Soldier original;
    if (size >= 10) {
        System.out.println(x:"Usted no puede agregar mas soldados");
        return;
    }
    do {
        coordinates = coordinate(army);
        row = coordinates[0];
        column = coordinates[1];
        clonedSoldier = new Soldier();
        original = army.get(row).get(column);

        if (original != null) {
            for (i = 0; i < army.size(); i++) {
                ArrayList<Soldier> rowArrayList = army.get(i);
                for (j = 0; j < rowArrayList.size(); j++) {
                    if (army.get(i).get(j) == null) {
                        clonedSoldier = original.clone();
                        clonedSoldier.setRow(row);
                        clonedSoldier.setColumn(column);
                        army.get(i).set(j, clonedSoldier);
                        armyUni.add(clonedSoldier);
                        return;
                    }
                }
            }
        }
    } while (true);

}
```

- Este método se encarga de clonar un Soldier, la acción no se ejecutará si el ejército tiene 10

objetos. Para lograr este propósito, se hace uso del método de la clase Soldier clone el cual ya fue explicado. A pesar de que el método debe clonar un Soldier tal cual, no se llega a cumplir a cabalidad ya que los atributos de row y column son únicos puesto que son la ubicación del soldier.

Listing 45: Compilando y probando el metodo cloneSoldier

```
javac VideoJuego.java
java VideoJuego
1: Juego rapido
2: Juego personalizado
3: Salir
2
DATOS DEL EJRCITO A
Soldier [name=Soldier2x4, lifePoints=, row=2, column=4]
Soldier [name=Soldier2x8, lifePoints=, row=2, column=8]
Soldier [name=Soldier3x5, lifePoints=, row=3, column=5]
Soldier [name=Soldier6x1, lifePoints=, row=6, column=1]
Soldier [name=Soldier6x3, lifePoints=, row=6, column=3]
Soldier [name=Soldier7x9, lifePoints=, row=7, column=9]
DATOS DEL EJRCITO B
Soldier [name=Soldier1x1, lifePoints=, row=1, column=1]
Soldier [name=Soldier2x3, lifePoints=, row=2, column=3]
Soldier [name=Soldier4x6, lifePoints=, row=4, column=6]
Soldier [name=Soldier5x4, lifePoints=, row=5, column=4]
Soldier [name=Soldier6x7, lifePoints=, row=6, column=7]
Soldier [name=Soldier7x6, lifePoints=, row=7, column=6]
Soldier [name=Soldier8x7, lifePoints=, row=8, column=7]
Soldier [name=Soldier9x2, lifePoints=, row=9, column=2]
1: Crear soldado
2: Eliminar soldado
3: Clonar soldado
4: Modificar soldado
5: Comparar Soldado
6: Intercambiar soldado
7: Ver soldado
8: Ver ejercito
9: Sumar niveles
10: Jugar
11: Volver al menu principal
3
Elija el ejercito que editara:
1: Ejercito A
2: Ejercito B
1
Ingrese el soldado a clonar
Ingrese las coordenadas
6a
DATOS DEL EJRCITO A
Soldier [name=Soldier2x4, lifePoints=, row=2, column=4]
Soldier [name=Soldier2x8, lifePoints=, row=2, column=8]
Soldier [name=Soldier3x5, lifePoints=, row=3, column=5]
Soldier [name=Soldier6x1, lifePoints=, row=6, column=1]
Soldier [name=Soldier6x3, lifePoints=, row=6, column=3]
Soldier [name=Soldier7x9, lifePoints=, row=7, column=9]
Soldier [name=Soldier6x1, lifePoints=, row=5, column=0]
```

```
DATOS DEL EJRCITO B
Soldier [name=Soldier1x1, lifePoints=, row=1, column=1]
Soldier [name=Soldier2x3, lifePoints=, row=2, column=3]
Soldier [name=Soldier4x6, lifePoints=, row=4, column=6]
Soldier [name=Soldier5x4, lifePoints=, row=5, column=4]
Soldier [name=Soldier6x7, lifePoints=, row=6, column=7]
Soldier [name=Soldier7x6, lifePoints=, row=7, column=6]
Soldier [name=Soldier8x7, lifePoints=, row=8, column=7]
Soldier [name=Soldier9x2, lifePoints=, row=9, column=2]
1: Crear soldado
2: Eliminar soldado
3: Clonar soldado
4: Modificar soldado
5: Comparar Soldado
6: Intercambiar soldado
7: Ver soldado
8: Ver ejercito
9: Sumar niveles
10: Jugar
11: Volver al menu principal
11
1: Juego rapido
2: Juego personalizado
3: Salir
3
```

Listing 46: Commit: c98829adbf11f4704c7b2950873e39e70603d38d

```
git add VideoJuego.java
git commit -m "Se crea el metodo cloneSoldier"
git push -u origin main
```

Listing 47: Se implementa el método para cambiar los atributos de un soldado

```
code VideoJuego.java
```

```java
    // case 4: Este metodo modifica cualqueira de los atributos de un soldado
    public static void changeAttributes(ArrayList<ArrayList<Soldier>> armyB, ArrayList<Soldier> armyU) {
        System.out.println(x:"Modificaremos un soldado!");
        do {
            int[] rowAndColumn = coordinate(armyB);
            int row = rowAndColumn[0];
            int column = rowAndColumn[1];
            Soldier chosenSoldier = armyB.get(row).get(column);
            if (chosenSoldier != null) {
                System.out.println(x:"1: Nivel de ataque\n2: Nivel de defensa\n3: Vida actual");
                int option = sc.nextInt();
                do {
                    System.out.println(x:"Ingrese el valor (1-5)");
                    int value = sc.nextInt();
                    if (value <= 5 && value >= 1) {
                        switch (option) {
                            case 1:
                                chosenSoldier.setAttackLevel(value);
                                break;
                            case 2:
                                chosenSoldier.setDefenseLevel(value);
                                break;
                            case 3:
                                chosenSoldier.setActualLife(value);
                                break;
                        }
                        armyB.get(row).set(column, chosenSoldier);

                        return;

                    } else {
                        System.out.println(x:"Valor no válido");
                    }
                } while (true);
            } else {
                System.out.println(x:"Soldado no encontrado!");
            }
        } while (true);
    }
```

- Este método se encarga de modificar los atributos de un Soldier, entre estos tenemos el nivel de vida actual, los puntos de ataque y defensa.

Listing 48: Compilando y probando el metodo changeAttribute

```
javac VideoJuego.java
java VideoJuego
1: Juego rapido
2: Juego personalizado
3: Salir
2
DATOS DEL EJRCITO A
Soldier [name=Soldier1x10, row=1, column=10, attackLevel=3, defenseLevel=1,
    actualLife=2, speed=0, attitude=Repose, current=true]
Soldier [name=Soldier2x7, row=2, column=7, attackLevel=1, defenseLevel=5, actualLife=1,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier3x2, row=3, column=2, attackLevel=1, defenseLevel=2, actualLife=2,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier4x8, row=4, column=8, attackLevel=1, defenseLevel=2, actualLife=3,
    speed=0, attitude=Repose, current=true]
```

```
Soldier [name=Soldier5x7, row=5, column=7, attackLevel=3, defenseLevel=2, actualLife=1,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier5x9, row=5, column=9, attackLevel=1, defenseLevel=4, actualLife=1,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier8x6, row=8, column=6, attackLevel=3, defenseLevel=5, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier8x8, row=8, column=8, attackLevel=2, defenseLevel=5, actualLife=3,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier9x2, row=9, column=2, attackLevel=3, defenseLevel=1, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier9x9, row=9, column=9, attackLevel=2, defenseLevel=5, actualLife=2,
    speed=0, attitude=Repose, current=true]
DATOS DEL EJRCITO B
Soldier [name=Soldier1x2, row=1, column=2, attackLevel=3, defenseLevel=4, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier4x4, row=4, column=4, attackLevel=2, defenseLevel=1, actualLife=5,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier7x9, row=7, column=9, attackLevel=4, defenseLevel=2, actualLife=1,
    speed=0, attitude=Repose, current=true]
1: Crear soldado
2: Eliminar soldado
3: Clonar soldado
4: Modificar soldado
5: Comparar Soldado
6: Intercambiar soldado
7: Ver soldado
8: Ver ejercito
9: Sumar niveles
10: Jugar
11: Volver al menu principal
4
Elija el ejercito que editara:
1: Ejercito A
2: Ejercito B
1
Modificaremos un soldado!
Ingrese las coordenadas
3B
1: Nivel de ataque
2: Nivel de defensa
3: Vida actual
1
Ingrese el valor (1-5)
5
DATOS DEL EJRCITO A
Soldier [name=Soldier1x10, row=1, column=10, attackLevel=3, defenseLevel=1,
    actualLife=2, speed=0, attitude=Repose, current=true]
Soldier [name=Soldier2x7, row=2, column=7, attackLevel=1, defenseLevel=5, actualLife=1,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier3x2, row=3, column=2, attackLevel=5, defenseLevel=2, actualLife=2,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier4x8, row=4, column=8, attackLevel=1, defenseLevel=2, actualLife=3,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier5x7, row=5, column=7, attackLevel=3, defenseLevel=2, actualLife=1,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier5x9, row=5, column=9, attackLevel=1, defenseLevel=4, actualLife=1,
```

```
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier8x6, row=8, column=6, attackLevel=3, defenseLevel=5, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier8x8, row=8, column=8, attackLevel=2, defenseLevel=5, actualLife=3,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier9x2, row=9, column=2, attackLevel=3, defenseLevel=1, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier9x9, row=9, column=9, attackLevel=2, defenseLevel=5, actualLife=2,
    speed=0, attitude=Repose, current=true]
DATOS DEL EJRCITO B
Soldier [name=Soldier1x2, row=1, column=2, attackLevel=3, defenseLevel=4, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier4x4, row=4, column=4, attackLevel=2, defenseLevel=1, actualLife=5,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier7x9, row=7, column=9, attackLevel=4, defenseLevel=2, actualLife=1,
    speed=0, attitude=Repose, current=true]
1: Crear soldado
2: Eliminar soldado
3: Clonar soldado
4: Modificar soldado
5: Comparar Soldado
6: Intercambiar soldado
7: Ver soldado
8: Ver ejercito
9: Sumar niveles
10: Jugar
11: Volver al menu principal
11
1: Juego rapido
2: Juego personalizado
3: Salir
3
```

Listing 49: Commit: 9277404a20d663b11bc7578ad51641619cca367a

```
git add VideoJuego.java
git commit -m "Se crea el metodo changeAttributes"
git push -u origin main
```

Listing 50: Se implementa el método para comparar dos soldados

```
code VideoJuego.java
```

```java
// CASE 5. Comparar soldados
public static void compareSoldier(ArrayList<ArrayList<Soldier>> army) {
    int row1, column1, row2, column2;
    Soldier s1, s2;
    int[] rowAndColumn1, rowAndColumn2;
    do {
        rowAndColumn1 = coordinate(army);
        row1 = rowAndColumn1[0];
        column1 = rowAndColumn1[1];
        s1 = army.get(row1).get(column1);

        rowAndColumn2 = coordinate(army);
        row2 = rowAndColumn2[0];
        column2 = rowAndColumn2[1];
        s2 = army.get(row2).get(column2);

        if (s1 == null || s2 == null) {
            System.out.println(x:"No se puede comparar nada!");
        } else {
            s2.compareTo(s1);
            return;
        }

    } while (true);
}
```

- Este método se encarga de comparar los atributos de un Soldier, entre estos tenemos el nivel de vida actual, los puntos de ataque, defensa, estado (Si viven o no) y la velocidad.

Listing 51: Compilando y probando el metodo compareSoldier

```
javac VideoJuego.java
java VideoJuego
1: Juego rapido
2: Juego personalizado
3: Salir
2
DATOS DEL EJRCITO A
Soldier [name=Soldier1x2, row=1, column=2, attackLevel=2, defenseLevel=3, actualLife=5,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier2x6, row=2, column=6, attackLevel=5, defenseLevel=1, actualLife=2,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier4x9, row=4, column=9, attackLevel=5, defenseLevel=4, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier7x9, row=7, column=9, attackLevel=4, defenseLevel=2, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier10x10, row=10, column=10, attackLevel=1, defenseLevel=1,
    actualLife=3, speed=0, attitude=Repose, current=true]
DATOS DEL EJRCITO B
Soldier [name=Soldier5x4, row=5, column=4, attackLevel=2, defenseLevel=1, actualLife=5,
    speed=0, attitude=Repose, current=true]
```

```
Soldier [name=Soldier5x7, row=5, column=7, attackLevel=2, defenseLevel=1, actualLife=1,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier7x8, row=7, column=8, attackLevel=2, defenseLevel=1, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier8x4, row=8, column=4, attackLevel=2, defenseLevel=4, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier8x6, row=8, column=6, attackLevel=1, defenseLevel=4, actualLife=2,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier9x8, row=9, column=8, attackLevel=2, defenseLevel=4, actualLife=3,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier10x3, row=10, column=3, attackLevel=2, defenseLevel=1,
    actualLife=4, speed=0, attitude=Repose, current=true]
Soldier [name=Soldier10x6, row=10, column=6, attackLevel=1, defenseLevel=3,
    actualLife=1, speed=0, attitude=Repose, current=true]
Soldier [name=Soldier10x7, row=10, column=7, attackLevel=4, defenseLevel=5,
    actualLife=5, speed=0, attitude=Repose, current=true]
Soldier [name=Soldier10x8, row=10, column=8, attackLevel=5, defenseLevel=2,
    actualLife=3, speed=0, attitude=Repose, current=true]
1: Crear soldado
2: Eliminar soldado
3: Clonar soldado
4: Modificar soldado
5: Comparar Soldado
6: Intercambiar soldado
7: Ver soldado
8: Ver ejercito
9: Sumar niveles
10: Jugar
11: Volver al menu principal
5
Elija el ejercito que editara:
1: Ejercito A
2: Ejercito B
2
Ingrese las coordenadas
5d
Ingrese las coordenadas
10c
Tienen igual nivel de ataque
Tiene igual nivel de defensa
Su vida actual son diferentes
Ambos viven
Tienen igual velocidad
DATOS DEL EJRCITO A
Soldier [name=Soldier1x2, row=1, column=2, attackLevel=2, defenseLevel=3, actualLife=5,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier2x6, row=2, column=6, attackLevel=5, defenseLevel=1, actualLife=2,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier4x9, row=4, column=9, attackLevel=5, defenseLevel=4, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier7x9, row=7, column=9, attackLevel=4, defenseLevel=2, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier10x10, row=10, column=10, attackLevel=1, defenseLevel=1,
    actualLife=3, speed=0, attitude=Repose, current=true]
DATOS DEL EJRCITO B
Soldier [name=Soldier5x4, row=5, column=4, attackLevel=2, defenseLevel=1, actualLife=5,
```

```
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier5x7, row=5, column=7, attackLevel=2, defenseLevel=1, actualLife=1,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier7x8, row=7, column=8, attackLevel=2, defenseLevel=1, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier8x4, row=8, column=4, attackLevel=2, defenseLevel=4, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier8x6, row=8, column=6, attackLevel=1, defenseLevel=4, actualLife=2,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier9x8, row=9, column=8, attackLevel=2, defenseLevel=4, actualLife=3,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier10x3, row=10, column=3, attackLevel=2, defenseLevel=1,
    actualLife=4, speed=0, attitude=Repose, current=true]
Soldier [name=Soldier10x6, row=10, column=6, attackLevel=1, defenseLevel=3,
    actualLife=1, speed=0, attitude=Repose, current=true]
Soldier [name=Soldier10x7, row=10, column=7, attackLevel=4, defenseLevel=5,
    actualLife=5, speed=0, attitude=Repose, current=true]
Soldier [name=Soldier10x8, row=10, column=8, attackLevel=5, defenseLevel=2,
    actualLife=3, speed=0, attitude=Repose, current=true]
1: Crear soldado
2: Eliminar soldado
3: Clonar soldado
4: Modificar soldado
5: Comparar Soldado
6: Intercambiar soldado
7: Ver soldado
8: Ver ejercito
9: Sumar niveles
10: Jugar
11: Volver al menu principal
11
1: Juego rapido
2: Juego personalizado
3: Salir
3
```

Listing 52: Commit: 9d78c3620d8f037fa789e96d6367c1c0f8d2a075

```
git add VideoJuego.java
git commit -m "Se crea el metodo compareSoldier"
git push -u origin main
```

Listing 53: Se implementa el método para intercambiar posiciones entre soldados

```
code VideoJuego.java
```

```java
// CASE 6: Este metodo intercambia las posiciones
public static void exchangePosition(ArrayList<ArrayList<Soldier>> army) {
    int[] rowAndColumn1, rowAndColumn2;
    int row1, row2, column1, column2;
    Soldier s1, s2;

    do {
        System.out.println(x:"Coordenadas del primer soldado");
        rowAndColumn1 = coordinate(army);
        row1 = rowAndColumn1[0];
        column1 = rowAndColumn1[1];
        System.out.println(x:"Coordenadas del segundo soldado ");
        rowAndColumn2 = coordinate(army);
        row2 = rowAndColumn2[0];
        column2 = rowAndColumn2[1];
        s1 = army.get(row1).get(column1);
        s2 = army.get(row2).get(column2);
        if (s1 == null && s2 == null) {
            System.out.println(x:"No se puede intercambiar");
        } else {
            s1.setRow(row2 + 1);
            s1.setColumn(column2 + 1);
            s2.setRow(row1 + 1);
            s2.setColumn(column1 + 1);
            army.get(row1).set(column1, s2);
            army.get(row2).set(column2, s1);
            return;
        }
    } while (true);
}
```

- Este método se encarga de intercambiar la posición de dos Soldier del mismo ejército.

Listing 54: Compilando y probando el metodo exchangeSoldier

```
javac VideoJuego.java
java VideoJuego
1: Juego rapido
2: Juego personalizado
3: Salir
2
DATOS DEL EJRCITO A
Soldier [name=Soldier2x10, row=2, column=10, attackLevel=2, defenseLevel=2,
    actualLife=2, speed=0, attitude=Repose, current=true]
Soldier [name=Soldier3x10, row=3, column=10, attackLevel=2, defenseLevel=3,
    actualLife=2, speed=0, attitude=Repose, current=true]
Soldier [name=Soldier4x4, row=4, column=4, attackLevel=4, defenseLevel=5, actualLife=3,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier5x9, row=5, column=9, attackLevel=3, defenseLevel=3, actualLife=2,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier6x7, row=6, column=7, attackLevel=5, defenseLevel=1, actualLife=4,
    speed=0, attitude=Repose, current=true]
```

```
Soldier [name=Soldier8x3, row=8, column=3, attackLevel=2, defenseLevel=4, actualLife=5,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier8x4, row=8, column=4, attackLevel=1, defenseLevel=1, actualLife=2,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier10x5, row=10, column=5, attackLevel=3, defenseLevel=2,
    actualLife=5, speed=0, attitude=Repose, current=true]
Soldier [name=Soldier10x6, row=10, column=6, attackLevel=2, defenseLevel=5,
    actualLife=4, speed=0, attitude=Repose, current=true]
DATOS DEL EJRCITO B
Soldier [name=Soldier2x2, row=2, column=2, attackLevel=4, defenseLevel=3, actualLife=3,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier2x3, row=2, column=3, attackLevel=3, defenseLevel=3, actualLife=3,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier3x5, row=3, column=5, attackLevel=4, defenseLevel=5, actualLife=1,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier6x2, row=6, column=2, attackLevel=2, defenseLevel=5, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier6x6, row=6, column=6, attackLevel=5, defenseLevel=3, actualLife=3,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier6x10, row=6, column=10, attackLevel=5, defenseLevel=4,
    actualLife=1, speed=0, attitude=Repose, current=true]
Soldier [name=Soldier8x1, row=8, column=1, attackLevel=2, defenseLevel=4, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier9x7, row=9, column=7, attackLevel=4, defenseLevel=3, actualLife=2,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier9x10, row=9, column=10, attackLevel=5, defenseLevel=2,
    actualLife=5, speed=0, attitude=Repose, current=true]
Soldier [name=Soldier10x4, row=10, column=4, attackLevel=2, defenseLevel=4,
    actualLife=3, speed=0, attitude=Repose, current=true]
1: Crear soldado
2: Eliminar soldado
3: Clonar soldado
4: Modificar soldado
5: Comparar Soldado
6: Intercambiar soldado
7: Ver soldado
8: Ver ejercito
9: Sumar niveles
10: Jugar
11: Volver al menu principal
6
Elija el ejercito que editara:
1: Ejercito A
2: Ejercito B
1
Coordenadas del primer soldado
Ingrese las coordenadas
2j
Coordenadas del segundo soldado
Ingrese las coordenadas
3j
DATOS DEL EJRCITO A
Soldier [name=Soldier2x10, row=3, column=10, attackLevel=2, defenseLevel=2,
    actualLife=2, speed=0, attitude=Repose, current=true]
Soldier [name=Soldier3x10, row=2, column=10, attackLevel=2, defenseLevel=3,
    actualLife=2, speed=0, attitude=Repose, current=true]
```

```
Soldier [name=Soldier4x4, row=4, column=4, attackLevel=4, defenseLevel=5, actualLife=3,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier5x9, row=5, column=9, attackLevel=3, defenseLevel=3, actualLife=2,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier6x7, row=6, column=7, attackLevel=5, defenseLevel=1, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier8x3, row=8, column=3, attackLevel=2, defenseLevel=4, actualLife=5,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier8x4, row=8, column=4, attackLevel=1, defenseLevel=1, actualLife=2,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier10x5, row=10, column=5, attackLevel=3, defenseLevel=2,
    actualLife=5, speed=0, attitude=Repose, current=true]
Soldier [name=Soldier10x6, row=10, column=6, attackLevel=2, defenseLevel=5,
    actualLife=4, speed=0, attitude=Repose, current=true]
DATOS DEL EJRCITO B
Soldier [name=Soldier2x2, row=2, column=2, attackLevel=4, defenseLevel=3, actualLife=3,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier2x3, row=2, column=3, attackLevel=3, defenseLevel=3, actualLife=3,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier3x5, row=3, column=5, attackLevel=4, defenseLevel=5, actualLife=1,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier6x2, row=6, column=2, attackLevel=2, defenseLevel=5, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier6x6, row=6, column=6, attackLevel=5, defenseLevel=3, actualLife=3,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier6x10, row=6, column=10, attackLevel=5, defenseLevel=4,
    actualLife=1, speed=0, attitude=Repose, current=true]
Soldier [name=Soldier8x1, row=8, column=1, attackLevel=2, defenseLevel=4, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier9x7, row=9, column=7, attackLevel=4, defenseLevel=3, actualLife=2,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier9x10, row=9, column=10, attackLevel=5, defenseLevel=2,
    actualLife=5, speed=0, attitude=Repose, current=true]
Soldier [name=Soldier10x4, row=10, column=4, attackLevel=2, defenseLevel=4,
    actualLife=3, speed=0, attitude=Repose, current=true]
1: Crear soldado
2: Eliminar soldado
3: Clonar soldado
4: Modificar soldado
5: Comparar Soldado
6: Intercambiar soldado
7: Ver soldado
8: Ver ejercito
9: Sumar niveles
10: Jugar
11: Volver al menu principal
11
1: Juego rapido
2: Juego personalizado
3: Salir
3
```

Listing 55: Commit: b825c23cde0e98518d0047f2867cb2a3f0d6fcf2

```
git add VideoJuego.java
git commit -m "Se implementa exchangePosition"
```

```
git push -u origin main
```

Listing 56: Se implementa el método para intercambiar posiciones entre soldados

```
code VideoJuego.java
```

```
122                         case 7:
123                             System.out.println(x:"Ingrese el nombre del soldado a buscar");
124                             String name = sc.next();
125                             binarySearchByName(copyU, name);
126                             break;
```

- En el case 7 se implementa lo necesario para efectuar la búsqueda del Soldier que el usuario deseé. Se emplea la búsqueda binaria para tal propósito debido a su eficiencia.

Listing 57: Compilando y probando el binarySearch

```
javac VideoJuego.java
java VideoJuego
1: Juego rapido
2: Juego personalizado
3: Salir
2
DATOS DEL EJRCITO A
Soldier [name=Soldier4x5, row=4, column=5, attackLevel=3, defenseLevel=1, actualLife=3,
    speed=0, attitude=Repose, current=true]
DATOS DEL EJRCITO B
Soldier [name=Soldier4x9, row=4, column=9, attackLevel=1, defenseLevel=5, actualLife=1,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier6x1, row=6, column=1, attackLevel=1, defenseLevel=4, actualLife=1,
    speed=0, attitude=Repose, current=true]
1: Crear soldado
2: Eliminar soldado
3: Clonar soldado
4: Modificar soldado
5: Comparar Soldado
6: Intercambiar soldado
7: Ver soldado
8: Ver ejercito
9: Sumar niveles
10: Jugar
11: Volver al menu principal
7
Elija el ejercito que editara:
1: Ejercito A
2: Ejercito B
1
Ingrese el nombre del soldado a buscar
soldier4x5
Se ha encontrado: Soldier [name=Soldier4x5, row=4, column=5, attackLevel=3,
    defenseLevel=1, actualLife=3, speed=0, attitude=Repose, current=true]
DATOS DEL EJRCITO A
Soldier [name=Soldier4x5, row=4, column=5, attackLevel=3, defenseLevel=1, actualLife=3,
    speed=0, attitude=Repose, current=true]
DATOS DEL EJRCITO B
```

```
Soldier [name=Soldier4x9, row=4, column=9, attackLevel=1, defenseLevel=5, actualLife=1,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier6x1, row=6, column=1, attackLevel=1, defenseLevel=4, actualLife=1,
    speed=0, attitude=Repose, current=true]
1: Crear soldado
2: Eliminar soldado
3: Clonar soldado
4: Modificar soldado
5: Comparar Soldado
6: Intercambiar soldado
7: Ver soldado
8: Ver ejercito
9: Sumar niveles
10: Jugar
```

Listing 58: Se implementa el método para sumar los atributos de un ArrayList de Soldier

```
code VideoJuego.java
```

```java
// Case 9: Se suman los atributos de tipo int, (menos speed), sin embargo no se
// agregara al ejercito dicho Soldie
public static void sumOfAttributes(ArrayList<ArrayList<Soldier>> armyB, ArrayList<Soldier> armyU) {
    Soldier soldierA = new Soldier();
    for (ArrayList<Soldier> row : armyB) {
        for (Soldier soldier : row) {
            if (soldier == null) {
                continue;
            } else {
                soldierA.mightySoldier(soldier);
            }

        }
    }
    System.out.println("Se ha creado a: " + soldierA);
}
```

- Este método se encarga de sumar los atributos de tipo int de un ArrayList de Soldier. Para lograr este cometido, debemos crear un Soldier cuyos atributos de tipo int sean iniciados en 0, por eso en la clase Soldier se creó un constructor que no admitía parámetros, pero cuyos valores ya eran inicializados en 0. Se usará dicho Soldier como un acumulador, ya que recordemos que cuando usamos métodos en objetos, lo que se pasa y se modifica es la referencia en si, por lo que en cada iteración los valores sumados de los atributos se irán acumulando en el Soldier creado.

Listing 59: Compilando y probando el metodo sumOfAttributes

```
javac VideoJuego.java
java VideoJuego
1: Juego rapido
2: Juego personalizado
3: Salir
2
DATOS DEL EJRCITO A
Soldier [name=Soldier2x5, row=2, column=5, attackLevel=5, defenseLevel=5, actualLife=5,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier3x7, row=3, column=7, attackLevel=1, defenseLevel=5, actualLife=5,
    speed=0, attitude=Repose, current=true]
```

```
Soldier [name=Soldier4x10, row=4, column=10, attackLevel=3, defenseLevel=3,
    actualLife=5, speed=0, attitude=Repose, current=true]
DATOS DEL EJRCITO B
Soldier [name=Soldier5x4, row=5, column=4, attackLevel=4, defenseLevel=3, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier8x1, row=8, column=1, attackLevel=1, defenseLevel=4, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier8x8, row=8, column=8, attackLevel=4, defenseLevel=4, actualLife=5,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier10x3, row=10, column=3, attackLevel=4, defenseLevel=2,
    actualLife=3, speed=0, attitude=Repose, current=true]
1: Crear soldado
2: Eliminar soldado
3: Clonar soldado
4: Modificar soldado
5: Comparar Soldado
6: Intercambiar soldado
7: Ver soldado
8: Ver ejercito
9: Sumar niveles
10: Jugar
11: Volver al menu principal
9
Elija el ejercito que editara:
1: Ejercito A
2: Ejercito B
1
Se ha creado a: Soldier [name=NotFound, row=0, column=0, attackLevel=9,
    defenseLevel=13, actualLife=15, speed=0, attitude=repose, current=true]
DATOS DEL EJRCITO A
Soldier [name=Soldier2x5, row=2, column=5, attackLevel=5, defenseLevel=5, actualLife=5,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier3x7, row=3, column=7, attackLevel=1, defenseLevel=5, actualLife=5,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier4x10, row=4, column=10, attackLevel=3, defenseLevel=3,
    actualLife=5, speed=0, attitude=Repose, current=true]
DATOS DEL EJRCITO B
Soldier [name=Soldier5x4, row=5, column=4, attackLevel=4, defenseLevel=3, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier8x1, row=8, column=1, attackLevel=1, defenseLevel=4, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier8x8, row=8, column=8, attackLevel=4, defenseLevel=4, actualLife=5,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier10x3, row=10, column=3, attackLevel=4, defenseLevel=2,
    actualLife=3, speed=0, attitude=Repose, current=true]
1: Crear soldado
2: Eliminar soldado
3: Clonar soldado
4: Modificar soldado
5: Comparar Soldado
6: Intercambiar soldado
7: Ver soldado
8: Ver ejercito
9: Sumar niveles
10: Jugar
11: Volver al menu principal
```

```
11
1: Juego rapido
2: Juego personalizado
3: Salir
3
```

Listing 60: Commit: ccb448aabe0c75743e18946a8e2965bc9790ce37

```
git add VideoJuego.java
git commit -m "Se crea el metodo sumOfAttribute"
git push -u origin main
```

Listing 61: Se implementa el método de batalla rápida

```
code VideoJuego.java
```

```java
741     //Este metodo permite jugar
742     public static boolean quickBattle(ArrayList<ArrayList<Soldier>> armyA,
743             ArrayList<ArrayList<Soldier>> armyB) {
744         boolean outPutOptions = true;
745         int optionsDuringGameFast;
746         myBoard(armyA, armyB);
747         String str;
748         int i = 0;
749         do {
750             if (i % 2 == 0) {
751                 str = "A";
752             } else {
753                 str = "B";
754             }
755             System.out.println("Turno de " + str);
756 
757             if (i % 2 == 0) {
758                 outPutOptions = moveSoldier(armyA, armyB, str);
759             } else {
760                 outPutOptions = moveSoldier(armyB, armyA, str);
761             }
762             myBoard(armyA, armyB);
763             if (!outPutOptions) {
764                 System.out.println(x:"1: Reinciar\n2: Volver al menú principal");
765                 optionsDuringGameFast = sc.nextInt();
766                 switch (optionsDuringGameFast) {
767                     case 1:
768                         return true;
769                     case 2:
770                         return false;
771                 }
772                 break;
773 
774             }
775 
776             i++;
777         } while (winnerBattle(armyA, armyB));
778         return false;
779     }
```

- Este método se encarga de poder jugar. En este método se usan los método moveSoldier el cual permite desplazarnos casilla por casilla y el método winnerBattle el cual detiene el flujo del programa arrojando un mensaje indicando el ganador. Este método retorna booleanos porque esto nos indica si el usuario decidió cancelar la partida o no.

Listing 62: Compilando y probando el programa completo

```
javac VideoJuego.java
java VideoJuego
1: Juego rapido
2: Juego personalizado
3: Salir
2
DATOS DEL EJRCITO A
Soldier [name=Soldier6x2, row=6, column=2, attackLevel=1, defenseLevel=5, actualLife=1,
    speed=0, attitude=Repose, current=true]
DATOS DEL EJRCITO B
Soldier [name=Soldier1x4, row=1, column=4, attackLevel=3, defenseLevel=3, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier6x4, row=6, column=4, attackLevel=1, defenseLevel=4, actualLife=2,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier7x6, row=7, column=6, attackLevel=4, defenseLevel=2, actualLife=1,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier7x7, row=7, column=7, attackLevel=1, defenseLevel=4, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier9x5, row=9, column=5, attackLevel=2, defenseLevel=5, actualLife=1,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier10x2, row=10, column=2, attackLevel=1, defenseLevel=3,
    actualLife=2, speed=0, attitude=Repose, current=true]
1: Crear soldado
2: Eliminar soldado
3: Clonar soldado
4: Modificar soldado
5: Comparar Soldado
6: Intercambiar soldado
7: Ver soldado
8: Ver ejercito
9: Sumar niveles
10: Jugar
11: Volver al menu principal
2
Elija el ejercito que editara:
1: Ejercito A
2: Ejercito B
2
Debe Eliminar un soldado
     A    B    C     D     E    F     G     H     I    J
1 |____||____||____||_b4_||____||____||____||____||____||____|
2 |____||____||____||____||____||____||____||____||____||____|
3 |____||____||____||____||____||____||____||____||____||____|
4 |____||____||____||____||____||____||____||____||____||____|
5 |____||____||____||____||____||____||____||____||____||____|
6 |____||_a1_||____||_b2_||____||____||____||____||____||____|
7 |____||____||____||____||____||_b1_||_b4_||____||____||____|
8 |____||____||____||____||____||____||____||____||____||____|
9 |____||____||____||____||_b1_||____||____||____||____||____|
```

```
10|____||_b2_||____||____||____||____||____||____||____||____|
Soldier [name=Soldier1x4, row=1, column=4, attackLevel=3, defenseLevel=3, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier6x4, row=6, column=4, attackLevel=1, defenseLevel=4, actualLife=2,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier7x6, row=7, column=6, attackLevel=4, defenseLevel=2, actualLife=1,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier7x7, row=7, column=7, attackLevel=1, defenseLevel=4, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier9x5, row=9, column=5, attackLevel=2, defenseLevel=5, actualLife=1,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier10x2, row=10, column=2, attackLevel=1, defenseLevel=3,
    actualLife=2, speed=0, attitude=Repose, current=true]
Ingrese las coordenadas
10b
DATOS DEL EJRCITO A
Soldier [name=Soldier6x2, row=6, column=2, attackLevel=1, defenseLevel=5, actualLife=1,
    speed=0, attitude=Repose, current=true]
DATOS DEL EJRCITO B
Soldier [name=Soldier1x4, row=1, column=4, attackLevel=3, defenseLevel=3, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier6x4, row=6, column=4, attackLevel=1, defenseLevel=4, actualLife=2,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier7x6, row=7, column=6, attackLevel=4, defenseLevel=2, actualLife=1,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier7x7, row=7, column=7, attackLevel=1, defenseLevel=4, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier9x5, row=9, column=5, attackLevel=2, defenseLevel=5, actualLife=1,
    speed=0, attitude=Repose, current=true]
1: Crear soldado
2: Eliminar soldado
3: Clonar soldado
4: Modificar soldado
5: Comparar Soldado
6: Intercambiar soldado
7: Ver soldado
8: Ver ejercito
9: Sumar niveles
10: Jugar
11: Volver al menu principal
1
Elija el ejercito que editara:
1: Ejercito A
2: Ejercito B
2
Ingrese la fila (ejm 1,2,3..10)
6
Ingrese la columna (ejm 1,2,3...10)
1
Ingrese el nivel de ataque
5
Ingrese el nivel de defensa
5
Ingrese el nivel de vida
5
Ingrese la velocidad
```

```
5
DATOS DEL EJRCITO A
Soldier [name=Soldier6x2, row=6, column=2, attackLevel=1, defenseLevel=5, actualLife=1,
    speed=0, attitude=Repose, current=true]
DATOS DEL EJRCITO B
Soldier [name=Soldier1x4, row=1, column=4, attackLevel=3, defenseLevel=3, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier6x4, row=6, column=4, attackLevel=1, defenseLevel=4, actualLife=2,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier7x6, row=7, column=6, attackLevel=4, defenseLevel=2, actualLife=1,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier7x7, row=7, column=7, attackLevel=1, defenseLevel=4, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier9x5, row=9, column=5, attackLevel=2, defenseLevel=5, actualLife=1,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier6X1, row=6, column=1, attackLevel=5, defenseLevel=5, actualLife=5,
    speed=0, attitude=Repose, current=true]
1: Crear soldado
2: Eliminar soldado
3: Clonar soldado
4: Modificar soldado
5: Comparar Soldado
6: Intercambiar soldado
7: Ver soldado
8: Ver ejercito
9: Sumar niveles
10: Jugar
11: Volver al menu principal
10
Elija el ejercito que editara:
1: Ejercito A
2: Ejercito B
1
ooooooooooooooo FASE 1 DE LA CONTIENDA ooooooooooooooo
Mostrando estadisticas de cada ejercito

DATOS DEL DEL EJERCITO A
Soldier [name=Soldier6x2, row=6, column=2, attackLevel=1, defenseLevel=5, actualLife=1,
    speed=0, attitude=Repose, current=true]
DATOS DEL EJRCITO B
Soldier [name=Soldier1x4, row=1, column=4, attackLevel=3, defenseLevel=3, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier6x4, row=6, column=4, attackLevel=1, defenseLevel=4, actualLife=2,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier7x6, row=7, column=6, attackLevel=4, defenseLevel=2, actualLife=1,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier7x7, row=7, column=7, attackLevel=1, defenseLevel=4, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier9x5, row=9, column=5, attackLevel=2, defenseLevel=5, actualLife=1,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier6X1, row=6, column=1, attackLevel=5, defenseLevel=5, actualLife=5,
    speed=0, attitude=Repose, current=true]

ooooooooooooooo FASE 2 DE LA CONTIENDA ooooooooooooooo
Mostrando el tablero de juego
   A     B     C     D     E     F     G     H     I     J
```

```
1 |____||____||____||_b4_||____||____||____||____||____||____|
2 |____||____||____||____||____||____||____||____||____||____|
3 |____||____||____||____||____||____||____||____||____||____|
4 |____||____||____||____||____||____||____||____||____||____|
5 |____||____||____||____||____||____||____||____||____||____|
6 |_b5_||_a1_||____||_b2_||____||____||____||____||____||____|
7 |____||____||____||____||____||_b1_||_b4_||____||____||____|
8 |____||____||____||____||____||____||____||____||____||____|
9 |____||____||____||____||_b1_||____||____||____||____||____|
10|____||____||____||____||____||____||____||____||____||____|

+++++++++++++++++ FASE 3 DE LA CONTIENDA +++++++++++++++++
    A     B     C     D     E     F     G     H     I     J
1 |____||____||____||_b4_||____||____||____||____||____||____|
2 |____||____||____||____||____||____||____||____||____||____|
3 |____||____||____||____||____||____||____||____||____||____|
4 |____||____||____||____||____||____||____||____||____||____|
5 |____||____||____||____||____||____||____||____||____||____|
6 |_b5_||_a1_||____||_b2_||____||____||____||____||____||____|
7 |____||____||____||____||____||_b1_||_b4_||____||____||____|
8 |____||____||____||____||____||____||____||____||____||____|
9 |____||____||____||____||_b1_||____||____||____||____||____|
10|____||____||____||____||____||____||____||____||____||____|
Turno de A
Ingrese las coordenadas
6b
Hacia donde quiere mover? up(1), down(2), left(3), right(4), exit(5)
3
Estadisticas de batalla:     Soldado de atacante: 16.666666666666668%   Soldado en
    reposo: 83.33333333333333 Salio como aleatorio: 82.79657290860615
Soldier [name=Soldier6X1, row=6, column=1, attackLevel=5, defenseLevel=5, actualLife=5,
    speed=2, attitude=offensive, current=true]
Soldier [name=Soldier6x2, row=6, column=2, attackLevel=1, defenseLevel=5, actualLife=1,
    speed=1, attitude=offensive, current=true]
Ha ganado el soldado en reposo
Soldier [name=Soldier6x2, row=6, column=2, attackLevel=1, defenseLevel=5, actualLife=1,
    speed=1, attitude=offensive, current=true]
    A     B     C     D     E     F     G     H     I     J
1 |____||____||____||_b4_||____||____||____||____||____||____|
2 |____||____||____||____||____||____||____||____||____||____|
3 |____||____||____||____||____||____||____||____||____||____|
4 |____||____||____||____||____||____||____||____||____||____|
5 |____||____||____||____||____||____||____||____||____||____|
6 |_b5_||____||____||_b2_||____||____||____||____||____||____|
7 |____||____||____||____||____||_b1_||_b4_||____||____||____|
8 |____||____||____||____||____||____||____||____||____||____|
9 |____||____||____||____||_b1_||____||____||____||____||____|
10|____||____||____||____||____||____||____||____||____||____|
Ha ganado B
DATOS DEL EJRCITO A
Soldier [name=Soldier6x2, row=6, column=2, attackLevel=1, defenseLevel=5, actualLife=1,
    speed=1, attitude=offensive, current=true]
DATOS DEL EJRCITO B
Soldier [name=Soldier1x4, row=1, column=4, attackLevel=3, defenseLevel=3, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier6x4, row=6, column=4, attackLevel=1, defenseLevel=4, actualLife=2,
```

```
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier7x6, row=7, column=6, attackLevel=4, defenseLevel=2, actualLife=1,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier7x7, row=7, column=7, attackLevel=1, defenseLevel=4, actualLife=4,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier9x5, row=9, column=5, attackLevel=2, defenseLevel=5, actualLife=1,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier6X1, row=6, column=1, attackLevel=5, defenseLevel=5, actualLife=5,
    speed=2, attitude=offensive, current=true]
1: Crear soldado
2: Eliminar soldado
3: Clonar soldado
4: Modificar soldado
5: Comparar Soldado
6: Intercambiar soldado
7: Ver soldado
8: Ver ejercito
9: Sumar niveles
10: Jugar
11: Volver al menu principal
11
1: Juego rapido
2: Juego personalizado
3: Salir
1
Desea jugar una ronda?(si/no)
si
oooooooooooooooo FASE 1 DE LA CONTIENDA oooooooooooooooo
Mostrando estadisticas de cada ejercito

DATOS DEL DEL EJERCITO A
Soldier [name=Soldier5x1, row=5, column=1, attackLevel=5, defenseLevel=1, actualLife=2,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier8x5, row=8, column=5, attackLevel=5, defenseLevel=4, actualLife=5,
    speed=0, attitude=Repose, current=true]
DATOS DEL EJRCITO B
Soldier [name=Soldier4x1, row=4, column=1, attackLevel=1, defenseLevel=3, actualLife=1,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier4x5, row=4, column=5, attackLevel=2, defenseLevel=2, actualLife=2,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier5x5, row=5, column=5, attackLevel=1, defenseLevel=5, actualLife=3,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier6x5, row=6, column=5, attackLevel=1, defenseLevel=2, actualLife=2,
    speed=0, attitude=Repose, current=true]
Soldier [name=Soldier7x9, row=7, column=9, attackLevel=2, defenseLevel=4, actualLife=1,
    speed=0, attitude=Repose, current=true]

oooooooooooooooo FASE 2 DE LA CONTIENDA oooooooooooooooo
Mostrando el tablero de juego
    A     B     C     D     E     F     G     H     I     J
1 |____||____||____||____||____||____||____||____||____||____|
2 |____||____||____||____||____||____||____||____||____||____|
3 |____||____||____||____||____||____||____||____||____||____|
4 |_b1_||____||____||____||_b2_||____||____||____||____||____|
5 |_a2_||____||____||____||_b3_||____||____||____||____||____|
6 |____||____||____||____||_b2_||____||____||____||____||____|
```

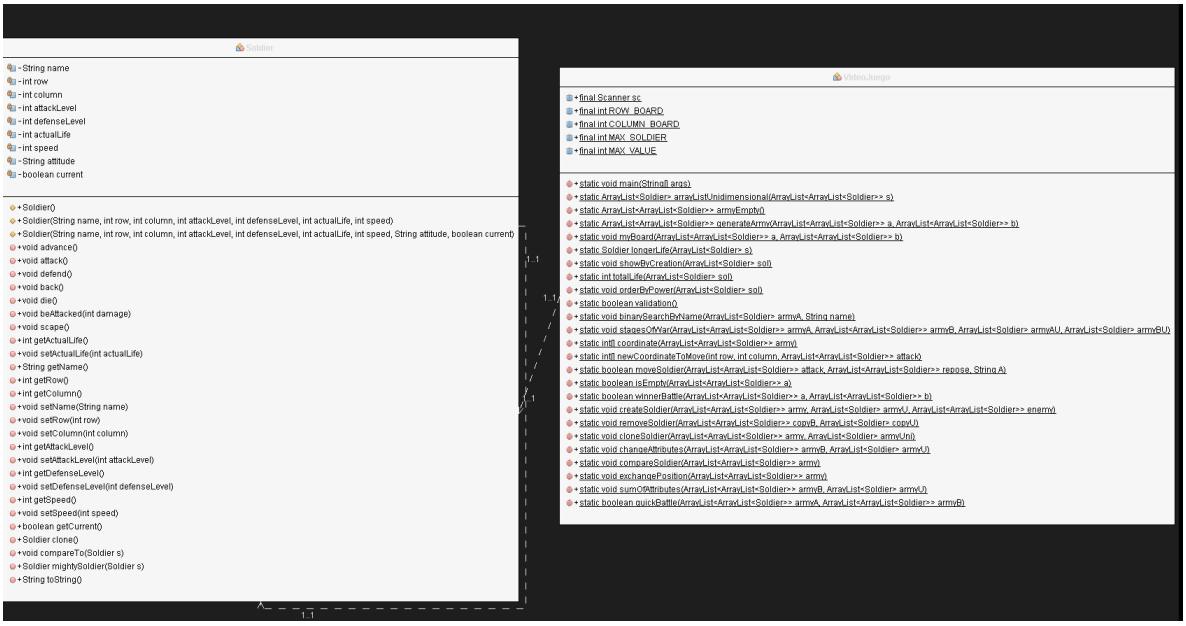
```
7 |____||____||____||____||____||____||____||____||_b1_||____|
8 |____||____||____||____||_a5_||____||____||____||____||____|
9 |____||____||____||____||____||____||____||____||____||____|
10|____||____||____||____||____||____||____||____||____||____|

++++++++++++++++++ FASE 3 DE LA CONTIENDA ++++++++++++++++++
   A    B    C    D    E    F    G    H    I    J
1 |____||____||____||____||____||____||____||____||____||____|
2 |____||____||____||____||____||____||____||____||____||____|
3 |____||____||____||____||____||____||____||____||____||____|
4 |_b1_||____||____||____||_b2_||____||____||____||____||____|
5 |_a2_||____||____||____||_b3_||____||____||____||____||____|
6 |____||____||____||____||_b2_||____||____||____||____||____|
7 |____||____||____||____||____||____||____||____||_b1_||____|
8 |____||____||____||____||_a5_||____||____||____||____||____|
9 |____||____||____||____||____||____||____||____||____||____|
10|____||____||____||____||____||____||____||____||____||____|
Turno de A
Ingrese las coordenadas
salir
   A    B    C    D    E    F    G    H    I    J
1 |____||____||____||____||____||____||____||____||____||____|
2 |____||____||____||____||____||____||____||____||____||____|
3 |____||____||____||____||____||____||____||____||____||____|
4 |_b1_||____||____||____||_b2_||____||____||____||____||____|
5 |_a2_||____||____||____||_b3_||____||____||____||____||____|
6 |____||____||____||____||_b2_||____||____||____||____||____|
7 |____||____||____||____||____||____||____||____||_b1_||____|
8 |____||____||____||____||_a5_||____||____||____||____||____|
9 |____||____||____||____||____||____||____||____||____||____|
10|____||____||____||____||____||____||____||____||____||____|
1: Reinciar
2: Volver al menu principal
2
1: Juego rapido
2: Juego personalizado
3: Salir
3
```

Listing 63: Commit: a2ccc9dec857e9f4649ee5e6729ac4f3194ffb7d

```
git add VideoJuego.java
git commit -m "Metodo quickBattle terminado"
git push -u origin main
```

4.5. Diagrama UML



4.6. Estructura de laboratorio 12

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

```
lab12:
|   Soldier.java
|   VideoJuego.java
|
 ---latex
    |   programacion_lab12_rescobedoq_v1.0.pdf
    |   programacion_lab12_rescobedoq_v1.0.tex
    |
     ---img
           armyEmpty.jpg
           arrayListU.jpg
           binary.jpg
           binarySearchByName.jpg
           burbuja.jpg
           changeAttribute.jpg
           changesMyBoard.jpg
           classSoldier.jpg
           clone.jpg
           cloneSoldier.jpg
           compare.jpg
           compareSoldier.jpg
           constructors.jpg
           coordinate.jpg
           copyReference.jpg
           createSoldier.jpg
           exchangeSoldier.jpg
           generateArmy.jpg
```

```
generateUni.jpg
insertion.jpg
isEmpty.jpg
logo_abet.png
logo_episunsa.png
logo_unsa.jpg
longerLife.jpg
method.jpg
method2.jpg
method3.jpg
method4.jpg
mightySoldier.jpg
moveSoldier.jpg
myBoard.jpg
newCoordinate.jpg
orderByPower.png
quickBattle.jpg
removeSoldier.jpg
searchSoldier.jpg
sequence.jpg
showByCreation.jpg
stages.jpg
sumAttributes.jpg
toString.jpg
totalLife.jpg
uml.png
validation.jpg
winnerBattle.png
```

5. Rúbricas

5.1. Entregable Informe

Tabla 1: Tipo de Informe

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

5.2. 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 aprobada, siempre y cuando cumpla con todos lo items.
- El alumno debe autocalificarse en la columna **Estudiante** de acuerdo a la siguiente tabla:

Tabla 2: 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

Tabla 3: 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	4	
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 estan 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	2	
Total		20		18	

6. Referencias

- https://www.geeksforgeeks.org/binary-search/
- https://www.geeksforgeeks.org/insertion-sort/