

Informe de Laboratorio 11

Tema: Definición de Clases de Usuario Clase Soldado

Nota

Estudiante	Escuela	Asignatura
Sarayasi Huanaco,Jeferson Jesus	Escuela Profesional de Ingeniería de Sistemas	Fundamentos de la Programación 2

Laboratorio	Tema	Duración
11	Definición de Clases de Usuario Clase Soldado	48 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2024 - B		

Docente
Aedo Lopez, Marco Wilfredo

1. Tarea

- Que el alumno demuestre poder crear “clases definidas por el programador”
- Implementar métodos para las clases definidas por el programador

2. Equipos, Materiales y temas utilizados

- Sistema Operativo Windows 11.
- VIM 9.0.
- OpenJHK 64-Bits 17.0.7.
- Git 2.39.2.
- Cuenta en GitHub con el correo institucional
- Arreglos Estandar

3. URL de Repositorio Github

- URL del Repositorio GitHub para clonar o recuperar
- <https://github.com/JefersonSH/FP2-2024B.git>
- URL para el laboratorio 07 en el Repositorio GitHub.
- https://github.com/JefersonSH/FP2-2024B/tree/eb261ca92998cb186b738e515a242259f00efc83/Laboratorio_11

4. Ejercicios Resuletos

4.1. Commits

Ejercicio 01

- **Commit 1:** VideoJuego numero 8 con las modificaciones pedidas por el docente

Listing 1: Clase Direccion

```
1 import java.util.*;
2
3 public class VideoJuego8 {
4
5     public static void main(String[] args){
6         Scanner sc = new Scanner(System.in);
7         boolean condicion = true;
8         int turno = 1;
9
10        int tamao1 = (int)(Math.random()* 10 + 1); // Soldados del ejercito 1
11        int tamao2 = (int)(Math.random()* 10 + 1); // Soldados del ejercito 2
12        Soldado[][] tablero = new Soldado[10][10]; // Creando un arreglo bidimensional de
13           Soldados
14
15        AsignarAtributos(tablero, tamao1, 1); // Creando Soldados en el ejercito 0
16        AsignarAtributos(tablero, tamao2, 2); // Creando Soldados en el ejercito 1
17
18        ImprimirDatos(tablero);
19        ImprimirTablero(tablero);
20
21        System.out.println("Soldado con mas vida del ejercito 1 es: ");
22        EncontrarMayorVida(tablero, 10, 1);
23        System.out.println("Soldado con mas vida del ejercito 2 es: ");
24        EncontrarMayorVida(tablero, 10, 2);
25
26        System.out.println("Promedio de Vida del Ejercito 1: " + PromedioVida(tablero, 10, 1) +
27           "\n");
28        System.out.println("Promedio de Vida del Ejercito 2: " + PromedioVida(tablero, 10, 2) +
29           "\n");
30        System.out.println("\n-----\n");
31
32        System.out.println("Vida de todos los soldados:");
33        ImprimirVida(tablero);
34
35        System.out.println("Ranking de Vida segun Ordenamiento Burbuja");
```

```
33     OrdenamientoBurbuja(tablero, SoldadosVivos(tablero));
34
35     System.out.println("Ranking de Vida segun Ordenamiento por Insercion");
36     OrdenamientoInsercion(tablero, SoldadosVivos(tablero));
37
38     while(condicion){
39         Jugada(tablero, turno); //Ejecutara toda la jugada de cada turno
40         ImprimirTablero(tablero); //Despues de la jugada, se imprime el tablero y los datos
           de los soldados vivos
41         ImprimirDatos(tablero);
42
43         condicion = ActualizarCondicion(tablero); // Si no hay soldados vivos en ambos
           ejércitos, sera false
44
45         turno++;
46     }
47
48     Ganador(tablero);
49 }
50
51 public static void AsignarAtributos(Soldado[] [] tablero, int n, int ejercito){
52     for(int i = 0; i < n; i++){
53         int x = (int)(Math.random()*10); // x representa Filas
54         int y = (int)(Math.random()*10); // y representa a las columnas, siendo del 1 al 9
55         if(tablero[x][y] == null){ // Si la posicion [y][x] esta vacia, crea un Soldado
56             String nombre = "Soldado" + (ejercito) + "x" + (i + 1);
57             int vida = (int)(Math.random()*5+1);
58             int ataque = (int)(Math.random()*5+1);
59             int defensa = (int)(Math.random()*5+1);
60             tablero[x][y] = new Soldado(nombre, vida, ejercito, x + 1, y + 1, ataque, defensa);
61         } else {
62             i--; // Caso contrario, se regresa al ciclo
63         }
64     }
65 }
66
67 public static void ImprimirDatos(Soldado[] [] tablero){
68     for(int i = 0; i < 10; i++) // Filas
69         for(int j = 0; j < 10; j++) // Controla las columnas
70             if(tablero[i][j] != null)
71                 if(tablero[i][j].getVive()){
72                     System.out.println("Nombre: " + tablero[i][j].getNombre());
73                     System.out.println("Salud: " + tablero[i][j].getVidaActual());
74                     System.out.println("Ejercito: " + tablero[i][j].getEjercito());
75                     System.out.println("Fila: " + tablero[i][j].getFila());
76                     System.out.println("Columna: " + tablero[i][j].getColumna());
77                 }
78     System.out.println("\n-----\n");
79 }
80
81 public static void ImprimirTablero(Soldado[] [] tablero){
82     for(int i = 0; i < 10; i++){ // Filas
83         for(int j = 0; j < 10; j++){ // Columnas
84             if(tablero[i][j] != null) // Si el indice contiene un Soldado continua
85                 if(tablero[i][j].getVive()){ // Si el soldado esta vivo continua
86                     if(tablero[i][j].getEjercito() == 1) System.out.print("1 ");
```

```
87         if(tablero[i][j].getEjercito() == 2) System.out.print("2 ");
88     } else System.out.print("= ");
89     else System.out.print("= ");
90
91     }
92     System.out.print("\n"); // Una vez acabada una fila, hace un salto de linea
93 }
94 System.out.println("\n-----\n");
95 }
96
97 public static void EncontrarMayorVida(Soldado[][] tablero, int n, int ejercito){
98     int mayor = Integer.MIN_VALUE;
99
100     Soldado masVida = new Soldado();
101     for(int i = 0; i < n; i++) // Filas
102         for(int j = 0; j < n; j++) // Columnas
103             if(tablero[j][i] != null)
104                 if(tablero[j][i].getVive()){
105                     if(tablero[j][i].getEjercito() == ejercito) // Pasa si son del mismo ejercito
106                         if(tablero[j][i].getVidaActual() > mayor){
107                             mayor = tablero[j][i].getVidaActual(); //Si es mayor, actualiza la
108                                 variable
109                             masVida = tablero[j][i];
110                         }
111                 }
112     Imprimir(masVida);
113 }
114
115 public static void Imprimir(Soldado soldado){
116     System.out.println("Nombre: " + soldado.getNombre());
117     System.out.println("Vida: " + soldado.getVidaActual());
118     System.out.println("Fila: " + soldado.getFila());
119     System.out.println("Columna: " + soldado.getColumna());
120     System.out.println("Ejercito" + soldado.getEjercito());
121     System.out.println("\n-----\n");
122 }
123
124 public static int PromedioVida(Soldado[][] tablero, int n, int ejercito){
125     int promedio = 0;
126     int size = 0;
127     for(int i = 0; i < n; i++) // Filas
128         for(int j = 0; j < n; j++) // Columnas
129             if(tablero[j][i] != null)
130                 if(tablero[j][i].getVive() && tablero[j][i].getEjercito() == ejercito){
131                     promedio += tablero[j][i].getVidaActual();
132                     size++;
133                 }
134     return promedio/size;
135 }
136
137 public static void ImprimirVida(Soldado[][] tablero){
138     for(int i = 0; i < 10; i++) // Filas
139         for(int j = 0; j < 10; j++) // Columnas
140             if(tablero[j][i] != null)
141                 if(tablero[j][i].getVive())
142                     System.out.println("Vida de " + tablero[j][i].getNombre() + ": " +
```

```
142         tablero[j][i].getVidaActual());
143     System.out.println("\n-----\n");
144 }
145
146 public static void OrdenamientoBurbuja(Soldado[][] tablero, int n){
147     Soldado temp = new Soldado();
148     int contador = 0;
149     Soldado[] lista = new Soldado[n];
150     for(int i = 0; i < 10; i++) // Filas
151         for(int j = 0; j < 10; j++) // Columnas
152             if(tablero[j][i] != null){
153                 if(tablero[j][i].getVive()){
154                     lista[contador] = tablero[j][i];
155                     contador++;
156                 }
157             }
158
159     for(int i = 0; i < n - 1; i++)
160         for(int j = 0; j < n - 1; j++)
161             if(lista[j].getVidaActual() > lista[j+1].getVidaActual()){
162                 temp = lista[j+1];
163                 lista[j+1] = lista[j];
164                 lista[j] = temp;
165             }
166     for(int i = n - 1; i >= 0; i--){
167         Imprimir(lista[i]);
168     }
169     System.out.println("\n-----\n");
170 }
171
172 public static void OrdenamientoInsercion(Soldado[][] tablero, int n){
173     Soldado temp = new Soldado();
174     int contador = 0;
175     Soldado[] lista = new Soldado[n];
176     for(int i = 0; i < 10; i++) // Filas
177         for(int j = 0; j < 10; j++) // Columnas
178             if(tablero[j][i] != null){
179                 if(tablero[j][i].getVive()){
180                     lista[contador] = tablero[j][i];
181                     contador++;
182                 }
183             }
184
185     for(int i = 1; i < n; i++){
186         Soldado key = lista[i];
187         int j = i - 1;
188         while(j >= 0 && lista[j].getVidaActual() > key.getVidaActual()){
189             lista[j + 1] = lista[j];
190             j = j - 1;
191         }
192         lista[j + 1] = key;
193     }
194
195     for(int i = n - 1; i >= 0; i--){
196         Imprimir(lista[i]);
197     }
198 }
```

```
197     System.out.println("\n-----\n");
198 }
199
200 public static int SoldadosVivos(Soldado[] [] tablero){
201     int contador = 0;
202     for(int i = 0; i < 10; i++) // Filas
203         for(int j = 0; j < 10; j++) // Columnas
204             if(tablero[i][j] != null)
205                 if(tablero[i][j].getVive())
206                     contador++;
207     return contador;
208 }
209
210 public static int SoldadosVivos(Soldado[] [] tablero, int ejercito){
211     int contador = 0;
212     for(int i = 0; i < 10; i++) //Controla las filas
213         for(int j = 0; j < 10; j++) //Controla las columnas
214             if(tablero[i][j] != null)
215                 if(tablero[i][j].getVive() && tablero[i][j].getEjercito() == ejercito)
216                     contador++;
217     return contador;
218 }
219
220 public static void Jugada(Soldado[] [] tablero, int turno){
221     Scanner sc = new Scanner(System.in);
222
223     turno %= 2; //Los turnos pares seran los turnos del jugador 2
224     if(turno == 0) turno = 2; //Si es turno impar, jugara el jugador 1
225     //Se pide al usuario la jugada y se verifica si es valida
226     System.out.println("Jugador " + turno + ". Ingrese el soldado a mover del Ejercito " +
227         turno + ". (Ejm. Soldado1x1)");
228     String soldado = sc.nextLine();
229     boolean condicion1 = VerificarSoldado(soldado, turno, tablero);
230
231     while(!condicion1){
232         System.out.println("Soldado no encontrado, por favor ingrese otro");
233         soldado = sc.nextLine();
234         condicion1 = VerificarSoldado(soldado, turno, tablero);
235     }
236
237     System.out.println("Ahora ingrese la direccion (w:up, s:down, d:right, a:left)");
238     String direccion = sc.nextLine();
239     boolean condicion2 = VerificarDireccion(direccion, turno, tablero, soldado);
240     while(!condicion2){
241         System.out.println("Direccion no admitida, por favor ingrese otro");
242         direccion = sc.nextLine();
243         condicion2 = VerificarDireccion(direccion, turno, tablero, soldado);
244     }
245     //Si hay enfrentamiento, se realiza
246     SoldadoMoviendo(tablero, turno, soldado, direccion);
247 }
248
249 public static boolean VerificarSoldado(String soldado, int turno, Soldado[] [] tablero){
250     for(int i = 0; i < 10; i++) // Filas
251         for(int j = 0; j < 10; j++) // Columnas
252             if(tablero[i][j] != null)
```

```
252         if(tablero[i][j].getVive() && tablero[i][j].getEjercito() == turno)
253             if(tablero[i][j].getNombre().equals(soldado))
254                 return true;
255     return false;
256 }
257
258     public static boolean VerificarDireccion(String direccion, int turno, Soldado[] []
259         tablero, String soldado){
260         if(!(direccion.equals("w") || direccion.equals("s") || direccion.equals("a") ||
261             direccion.equals("d")))
262             return false;
263
264         for(int i = 0; i < 10; i++) { // Filas
265             for(int j = 0; j < 10; j++) { // Columnas
266                 if(tablero[i][j] != null && tablero[i][j].getVive() && tablero[i][j].getEjercito()
267                     == turno && tablero[i][j].getNombre().equals(soldado)){
268                     if(direccion.equals("w") && i - 1 >= 0) // Verifica movimiento hacia arriba
269                         return true;
270                     else if(direccion.equals("s") && i + 1 < 10) // Verifica movimiento hacia abajo
271                         return true;
272                     else if(direccion.equals("a") && j - 1 >= 0) // Verifica movimiento hacia la
273                         izquierda
274                         return true;
275                     else if(direccion.equals("d") && j + 1 < 10) // Verifica movimiento hacia la
276                         derecha
277                         return true;
278                 }
279             }
280         }
281         return false; //Si no devuelve true, el movimiento no es valido
282     }
283
284     public static void SoldadoMoviendose(Soldado[] [] tablero, int turno, String soldado,
285         String direccion){
286         int fila, columna;
287         int contador = 0; // Este contador es MUY IMPORTANTE, evitara que se llame el metodo
288             Combate() mas de 1 vez, evitando posibles errores con s o d
289         for(int i = 0; i < 10; i++) // Filas
290             for(int j = 0; j < 10; j++) // Columnas
291                 if(tablero[i][j] != null && tablero[i][j].getVive() && tablero[i][j].getEjercito()
292                     == turno && tablero[i][j].getNombre().equals(soldado) && contador == 0){
293                     if(direccion.equals("w") && i - 1 >= 0){ // Verifica movimiento hacia arriba
294                         Combate(tablero, i, j, direccion);
295                         contador++;
296                     }else if(direccion.equals("s") && i + 1 < 10){ // Verifica movimiento hacia
297                         abajo
298                         Combate(tablero, i, j, direccion);
299                         contador++;
300                     }else if(direccion.equals("a") && j - 1 >= 0){ // Verifica movimiento hacia la
301                         izquierda
302                         Combate(tablero, i, j, direccion);
303                         contador++;
304                     }else if(direccion.equals("d") && j + 1 < 10){ // Verifica movimiento hacia la
305                         derecha
306                         Combate(tablero, i, j, direccion);
307                         contador++;
308                     }
```

```
297     }
298     }
299 }
300
301 public static void Combate(Soldado[][] tablero, int x, int y, String direccion){
302     if(direccion.equals("w")){ //La estructura se repite para cada una de las 4 letras,
303         pero con sus variaciones
304         if(tablero[x - 1][y] == null || !tablero[x - 1][y].getVive()){ //Verifica que no
305             haya un Soldado vivo en el casillero a avanzar
306             tablero[x - 1][y] = new Soldado(tablero[x][y].getNombre(),
307                 tablero[x][y].getVidaActual(), tablero[x][y].getEjercito(), x + 1 - 1, y + 1);
308             tablero[x][y].morir();
309         } else {
310             int num = (int)(Math.random() * 100 + 1);
311             double total = tablero[x][y].getVidaActual() + tablero[x - 1][y].getVidaActual();
312             double probabilidadA = 100 * (tablero[x][y].getVidaActual() / total);
313             System.out.println("Probabilidades de ganar:\nSoldado actual: " + probabilidadA +
314                 "% --- Soldado enemigo: " + (100 - probabilidadA) + "%");
315             if(num < probabilidadA){ // El ganador seria el soldado 1
316                 tablero[x - 1][y] = new Soldado(tablero[x][y].getNombre(),
317                     tablero[x][y].getVidaActual() + 1, tablero[x][y].getEjercito(), x + 1 - 1,
318                     y + 1);
319                 tablero[x][y].morir();
320                 System.out.println("El Soldado actual gana");
321             } else { // Si no el ganador seria el soldado 2
322                 tablero[x - 1][y].setVidaActual(tablero[x - 1][y].getVidaActual() + 1);
323                 tablero[x][y].morir();
324                 System.out.println("El Soldado enemigo gana");
325             }
326         }
327     }
328     } else if(direccion.equals("s")){
329         if(tablero[x + 1][y] == null || !tablero[x + 1][y].getVive()){
330             tablero[x + 1][y] = new Soldado(tablero[x][y].getNombre(),
331                 tablero[x][y].getVidaActual(), tablero[x][y].getEjercito(), x + 1 + 1, y + 1);
332             tablero[x][y].morir();
333         } else {
334             int num = (int)(Math.random() * 100 + 1);
335             double total = tablero[x][y].getVidaActual() + tablero[x + 1][y].getVidaActual();
336             double probabilidadA = 100 * (tablero[x][y].getVidaActual() / total);
337             System.out.println("Probabilidades de ganar:\nSoldado actual: " + probabilidadA +
338                 "% --- Soldado enemigo: " + (100 - probabilidadA) + "%");
339             if(num < probabilidadA){ // El ganador seria el soldado 1
340                 tablero[x + 1][y] = new Soldado(tablero[x][y].getNombre(),
341                     tablero[x][y].getVidaActual() + 1, tablero[x][y].getEjercito(), x + 1 - 1,
342                     y + 1);
343                 tablero[x][y].morir();
344                 System.out.println("El Soldado actual gana");
345             } else { // Si no el ganador seria el soldado 2
346                 tablero[x + 1][y].setVidaActual(tablero[x + 1][y].getVidaActual() + 1);
347                 tablero[x][y].morir();
348                 System.out.println("El Soldado enemigo gana");
349             }
350         }
351     }
352     } else if(direccion.equals("a")){
353         if(tablero[x][y - 1] == null || !tablero[x][y - 1].getVive()){
354             tablero[x][y - 1] = new Soldado(tablero[x][y].getNombre(),
```



```
        tablero[x][y].getVidaActual(), tablero[x][y].getEjercito(), x + 1, y + 1 - 1
    );
    tablero[x][y].morir();
} else {
    int num = (int)(Math.random() * 100 + 1);
    double total = tablero[x][y].getVidaActual() + tablero[x][y - 1].getVidaActual();
    double probabilidadA = 100 * (tablero[x][y].getVidaActual() / total);
    System.out.println("Probabilidades de ganar:\nSoldado actual: " + probabilidadA +
        "% --- Soldado enemigo: " + (100 - probabilidadA) + "%");
    if(num < probabilidadA){ // El ganador seria el soldado 1
        tablero[x][y - 1] = new Soldado(tablero[x][y].getNombre(),
            tablero[x][y].getVidaActual() + 1, tablero[x][y].getEjercito(), x + 1 - 1,
            y + 1);
        tablero[x][y].morir();
        System.out.println("El Soldado actual gana");
    } else { // Si no el ganador seria el soldado 2
        tablero[x][y - 1].setVidaActual(tablero[x][y - 1].getVidaActual() + 1);
        tablero[x][y].morir();
        System.out.println("El Soldado enemigo gana");
    }
}
} else if(direccion.equals("d")){
    if(tablero[x][y + 1] == null || !tablero[x][y + 1].getVive()){
        tablero[x][y + 1] = new Soldado(tablero[x][y].getNombre(),
            tablero[x][y].getVidaActual(), tablero[x][y].getEjercito(), x + 1, y + 1 + 1);
        tablero[x][y].morir();
    } else {
        int num = (int)(Math.random() * 100 + 1);
        double total = tablero[x][y].getVidaActual() + tablero[x][y + 1].getVidaActual();
        double probabilidadA = 100 * (tablero[x][y].getVidaActual() / total);
        System.out.println("Probabilidades de ganar:\nSoldado actual: " + probabilidadA +
            "% --- Soldado enemigo: " + (100 - probabilidadA) + "%");
        if(num < probabilidadA){ // El ganador seria el soldado 1
            tablero[x][y + 1] = new Soldado(tablero[x][y].getNombre(),
                tablero[x][y].getVidaActual() + 1, tablero[x][y].getEjercito(), x + 1 - 1,
                y + 1);
            tablero[x][y].morir();
            System.out.println("El Soldado actual gana");
        } else { // Si no el ganador seria el soldado 2
            tablero[x][y + 1].setVidaActual(tablero[x][y + 1].getVidaActual() + 1);
            tablero[x][y].morir();
            System.out.println("El Soldado enemigo gana");
        }
    }
}
}
}

public static boolean ActualizarCondicion(Soldado[][] tablero){
    int ejercito1 = SoldadosVivos(tablero, 1);
    int ejercito2 = SoldadosVivos(tablero, 2);
    if(ejercito1 == 0 || ejercito2 == 0)
        return false;
    return true;
}

public static void Ganador(Soldado[][] tablero){
```

```
390     if(SoldadosVivos(tablero, 1) == 0)
391         System.out.println("El Ganador es el Ejercito 2");
392     else if(SoldadosVivos(tablero, 2) == 0)
393         System.out.println("El Ganador es el Ejercito 1");
394     else System.out.println("Error");
395 }
396 }
```

- **Commit 2:** Se modificaron y crearon algunos metodos del constructor


































Listing 2: Clase Persona

```
1 public class Soldado{
2     private String nombre;
3     private int nivelVida;
4     private int vidaActual;
5     private boolean vive;
6     private int columna;
7     private int fila;
8     private int ejercito;
9     private int nivelAtaque;
10    private int nivelDefensa;
11    private int velocidad;
12    private String actitud;
13
14    public Soldado(){
15        nivelVida = 0;
16        vidaActual = 0;
17        nivelAtaque = 0;
18        nivelDefensa = 0;
19        vive = true;
20    }
21
22    public Soldado(String nombre, int nivelVida, int ejercito, int fila, int columna){
23        this.nombre = nombre;
24        this.nivelVida = nivelVida;
25        this.vidaActual = nivelVida;
26        this.ejercito = ejercito;
27        this.fila = fila;
28        this.columna = columna;
29        vive = true;
30    }
31
32    public Soldado(String nombre, int nivelVida, int ejercito, int fila, int columna, int
        nivelAtaque, int nivelDefensa){
33        this.nombre = nombre;
34        this.nivelVida = nivelVida;
35        this.vidaActual = nivelVida;
36        this.ejercito = ejercito;
37        this.fila = fila;
38        this.columna = columna;
39        this.nivelAtaque = nivelAtaque;
40        this.nivelDefensa = nivelDefensa;
41        vive = true;
42    }
```

```
43
44 public void atacar(){
45     avanzar();
46     actitud = "ofensiva";
47 }
48
49 public void defender(){
50     velocidad = 0;
51     actitud = "defensiva";
52 }
53
54 public void avanzar(){
55     velocidad++;
56 }
57
58 public void retroceder(){
59     if(velocidad > 0){
60         velocidad = 0;
61         actitud = "defensiva";
62     } else velocidad --;
63 }
64
65 public void serAtacado(int num){
66     vidaActual -= num;
67 }
68
69 public void huir(){
70     velocidad += 2;
71     actitud = "fuga";
72 }
73
74 public void morir(){
75     vidaActual = 0;
76     vive = false;
77 }
78
79 public void setVidaActual(int vidaActual){
80     this.vidaActual = vidaActual;
81 }
82
83 public int getVidaActual(){
84     return vidaActual;
85 }
86
87 public void vivir(){
88     vive = true;
89 }
90
91 public void setColumna(int columna){
92     this.columna = columna;
93 }
94
95 public void setFila(int fila){
96     this.fila = fila;
97 }
98
```

```
99     public void setEjercito(int ejercito){
100         this.ejercito = ejercito;
101     }
102
103     public int getColumna(){
104         return columna;
105     }
106
107     public int getFila(){
108         return fila;
109     }
110
111     public String getNombre(){
112         return nombre;
113     }
114
115     public int getEjercito(){
116         return ejercito;
117     }
118
119     public boolean getVive(){
120         return vive;
121     }
122 }
```

■ Diagrama UML

 Soldado
<ul style="list-style-type: none">  -String nombre  -int nivelAtaque  -int nivelDefensa  -int nivelVida  -int vidaActual  -int velocidad  -String actitud  -boolean vive  -int columna  -int fila  -int ejercito
<ul style="list-style-type: none">  Soldado()  Soldado(String nombre, int nivelVida, int ejercito, int fila, int columna)  Soldado(String nombre, int nivelVida, int ejercito, int fila, int columna, int nivelAtaque, int nivelDefensa)  atacar()  avanzar()  defender()  getColumna() : int  getEjercito() : int  getFila() : int  getNombre() : String  getVidaActual() : int  getVive() : boolean  huir()  morir()  retroceder()  serAtacado(int num)  setColumna(int columna)  setEjercito(int ejercito)  setFila(int fila)  setVidaActual(int vidaActual)  vivir()

5. Programa Compilado

```
PS C:\Users\jefer> & 'C:\Program Files\Java\jdk-20\bin\java.exe' '-XX:+ShowCod
Nombre: Soldado2x1
Salud: 3
Ejercito: 2
Fila:1
Columna: 7
Nombre: Soldado2x2
Salud: 1
Ejercito: 2
Fila:2
Columna: 2
Nombre: Soldado2x3
Salud: 4
Ejercito: 2
Fila:3
Columna: 9
Nombre: Soldado1x2
Salud: 5
Ejercito: 1
Fila:4
Columna: 3
Nombre: Soldado1x1
Salud: 3
Ejercito: 1
Fila:5
Columna: 8
Nombre: Soldado2x5
Salud: 4
Ejercito: 2
Fila:6
Columna: 8
Nombre: Soldado2x4
Salud: 5
Ejercito: 2
Fila:9
Columna: 8

-----

= = = = = 2 = = =
= 2 = = = = =
= = = = = 2 =
= = 1 = = = = =
= = = = = 1 =
= = = = = 2 =
= = = = =
= = = = =
= = = = = 2 =
= = = = =

-----
```

```
-----  
Soldado con mas vida del ejercito 1 es:  
Nombre: Soldado1x2  
Vida: 5  
Fila: 4  
Columna: 3  
Ejercito1  
  
-----  
  
Soldado con mas vida del ejercito 2 es:  
Nombre: Soldado2x4  
Vida: 5  
Fila: 9  
Columna: 8  
Ejercito2  
  
-----  
  
Promedio de Vida del Ejercito 1: 4  
  
Promedio de Vida del Ejercito 2: 3  
  
-----  
  
Vida de todos los soldados:  
Vida de Soldado2x2: 1  
Vida de Soldado1x2: 5  
Vida de Soldado2x1: 3  
Vida de Soldado1x1: 3  
Vida de Soldado2x5: 4  
Vida de Soldado2x4: 5  
Vida de Soldado2x3: 4  
  
-----  
  
Ranking de Vida segun Ordenamiento Burbuja  
Nombre: Soldado2x4  
Vida: 5  
Fila: 9  
Columna: 8  
Ejercito2  
  
-----  
  
Nombre: Soldado1x2  
Vida: 5  
Fila: 4  
Columna: 3  
Ejercito1  
  
-----
```

Ranking de Vida segun Ordenamiento Burbuja

Nombre: Soldado2x4

Vida: 5

Fila: 9

Columna: 8

Ejercito2

Nombre: Soldado1x2

Vida: 5

Fila: 4

Columna: 3

Ejercito1

Nombre: Soldado2x3

Vida: 4

Fila: 3

Columna: 9

Ejercito2

Nombre: Soldado2x5

Vida: 4

Fila: 6

Columna: 8

Ejercito2

Nombre: Soldado1x1

Vida: 3

Fila: 5

Columna: 8

Ejercito1

Nombre: Soldado2x1

Vida: 3

Fila: 1

Columna: 7

Ejercito2

Nombre: Soldado2x2

Vida: 1

Fila: 2

Columna: 2

Ejercito2

Ranking de Vida segun Ordenamiento por Insercion

Nombre: Soldado2x4

Vida: 5

Fila: 9

Columna: 8

Ejercito2

Nombre: Soldado1x2

Vida: 5

Fila: 4

Columna: 3

Ejercito1

Nombre: Soldado2x3

Vida: 4

Fila: 3

Columna: 9

Ejercito2

Nombre: Soldado2x5

Vida: 4

Fila: 6

Columna: 8

Ejercito2

Nombre: Soldado1x1

Vida: 3

Fila: 5

Columna: 8

Ejercito1

Nombre: Soldado2x1

Vida: 3

Fila: 1

Columna: 7

Ejercito2

Nombre: Soldado2x2

Vida: 1

Fila: 2

Columna: 2

Ejercito2

```
Jugador 1. Ingrese el soldado a mover del Ejercito 1. (Ejm. Soldado1x1)
Soldado2x2
Soldado no encontrado, por favor ingrese otro
Soldado1x2
Ahora ingrese la direccion (w:up, s:down, d:righ, a:left)
s
= = = = = 2 = = =
= 2 = = = = =
= = = = = 2 =
= = = = =
= = 1 = = = 1 =
= = = = = 2 =
= = = = =
= = = = =
= = = = = 2 =
= = = = =
-----
Nombre: Soldado2x1
Salud: 3
Ejercito: 2
Fila:1
Columna: 7
Nombre: Soldado2x2
Salud: 1
Ejercito: 2
Fila:2
Columna: 2
Nombre: Soldado2x3
Salud: 4
Ejercito: 2
Fila:3
Columna: 9
Nombre: Soldado1x2
Salud: 5
Ejercito: 1
Fila:5
Columna: 3
Nombre: Soldado1x1
Salud: 3
Ejercito: 1
Fila:5
Columna: 8
Nombre: Soldado2x5
Salud: 4
Ejercito: 2
Fila:6
Columna: 8
Nombre: Soldado2x4
Salud: 5
Ejercito: 2
Fila:9
Columna: 8
```

```
Jugador 2. Ingrese el soldado a mover del Ejercito 2. (Ejm. Soldado1x1)
Soldado2x1
Ahora ingrese la direccion (w:up, s:down, d:righth, a:left)
d
= = = = = 2 = =
= 2 = = = = =
= = = = = 2 =
= = = = =
= = 1 = = = 1 =
= = = = = 2 =
= = = = =
= = = = =
= = = = = 2 =
= = = = =
-----
Nombre: Soldado2x1
Salud: 3
Ejercito: 2
Fila:1
Columna: 8
Nombre: Soldado2x2
Salud: 1
Ejercito: 2
Fila:2
Columna: 2
Nombre: Soldado2x3
Salud: 4
Ejercito: 2
Fila:3
Columna: 9
Nombre: Soldado1x2
Salud: 5
Ejercito: 1
Fila:5
Columna: 3
Nombre: Soldado1x1
Salud: 3
Ejercito: 1
Fila:5
Columna: 8
Nombre: Soldado2x5
Salud: 4
Ejercito: 2
Fila:6
Columna: 8
Nombre: Soldado2x4
Salud: 5
Ejercito: 2
Fila:9
Columna: 8
```

Commit 3: Se han probado los métodos `SoldadoMoviéndose()` y `Combate()`, esperando que después de indicarle al soldado la dirección, se cambien los datos de los soldados según lo ocurrido tras el avance del soldado, y en caso haya un enfrentamiento, según la probabilidad de tomar o el criterio de vida del soldado, se determina aleatoriamente cuál de ambos gana el combate, donde no habría empate, y que luego se muestre el tablero actualizado, siendo al final de esta manera mostrando los datos actualizados imprimiendo la razón por la cual un enfrentamiento se ha ganado.

6. Referencias

- https://www.w3schools.com/js/js_ajax_intro.asp
- <https://www.youtube.com/watch?v=cAqmF7mtZv0>
- <https://www.youtube.com/watch?v=cAqmF7mtZv0>