

	UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA	
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		
Aprobación: 2022/03/01	Código: GUIA-PRLE-001	Página: 1

INFORME DE LABORATORIO

(formato estudiante)

INFORMACIÓN BÁSICA					
ASIGNATURA:	<i>Fundamentos de la programación 2</i>				
TÍTULO DE LA PRÁCTICA:	<i>Definición de Clases de Usuario</i> <i>Clase Soldado-Miembros de clase</i>				
NÚMERO DE PRÁCTICA:	<i>13</i>	AÑO LECTIVO:	<i>2023</i>	NRO. SEMESTRE:	<i>2do Semestre</i>
FECHA DE PRESENTACIÓN	<i>27/12/2023</i>	HORA DE PRESENTACIÓN	<i>15/40/00</i>		
INTEGRANTE (s) <i>Juan Diego Gutiérrez Ccama</i>				NOTA (0-20)	<i>Nota colocada por el docente</i>
DOCENTE(s): <i>Linno Jose Pinto Oppe</i>					

RESULTADOS Y PRUEBAS
I. EJERCICIOS RESUELTOS:

```
1 public class Soldado {
2     private static int totalSoldadosCreados = 0;
3     private static int sEjer1 = 0;
4     private static int sEjer2 = 0;
5     public static final int Soldados = 10;
6     private String nombre;
7     private int nivelAtaque;
8     private int nivelDefensa;
9     private int vidaActual;
10    private int velocidad;
11    private String actitud;
12    private boolean vive;
13    private int fila;
14    private int columna;
15    private int nivelVida;
16
17    public static void sumarEjer1() {
18        sEjer1++;
19        totalSoldadosCreados++;
20    }
21    public static void sumarEjer2() {
22        sEjer2++;
23        totalSoldadosCreados++;
24    }
25    public static void restarEjer1() {
26        sEjer1--;
27        totalSoldadosCreados--;
28    }
29    public static void restarEjer2() {
30        sEjer2--;
31        totalSoldadosCreados--;
32    }
33
34    public void atacar() {
35        avanzar();
36        actitud = "ofensiva";
37    }
38    public void defender() {
39        velocidad = 0;
40        actitud = "defensiva";
41    }
42    public void avanzar() {
43        velocidad++;
44    }
45    public void retroceder() {
46        if (velocidad > 0) {
47            velocidad = 0;
48            actitud = "defensiva";
49        } else {
50            velocidad--;
51        }
52    }
53    public void serAtacado(int puntosDaño) {
54        vidaActual -= puntosDaño;
55        if (vidaActual <= 0) {
56            morir();
57        }
58    }
59    public void huir() {
60        velocidad += 2;
61        actitud = "fuga";
62    }
63    public void morir() {
64        vive = false;
65    }
66    public static int getSoldCread() {
67        return totalSoldadosCreados;
68    }
69
70    public static int getSEjer1() {
71        return sEjer1;
72    }
73
74    public static int getSEjer2() {
75        return sEjer2;
76    }
77    //zona de sets
78    public void setNombre(String n){
79        nombre = n;
80    }
81    public void setNivelAtaque(int ataque){
82        nivelAtaque = ataque;
83    }
84    public void setNivelDefensa(int defensa){
85        nivelDefensa = defensa;
86    }
87 }
```

```
--
87 public void setVidaActual(int vidaAc) {
88     vidaActual = vidaAc;
89 }
90 public void setFila(int f){
91     fila = f;
92 }
93 public void setColumna(int c){
94     columna = c;
95 }
96 public void setNivelVida(int vida){
97     nivelVida = vida;
98 }
99 public void setVelocidad(int veloci){
100     velocidad = veloci;
101 }
102 //Zona de gets
103 public String getNombre(){
104     return nombre;
105 }
106 public int getFila(){
107     return fila;
108 }
109 public int getColumna(){
110     return columna;
111 }
112 public int getNivelVida(){
113     return nivelVida;
114 }
115 public int getNivelAtaque(){
116     return nivelAtaque;
117 }
118 public int getNivelDefensa(){
119     return nivelDefensa;
120 }
121 public int getVelocidad(){
122     return velocidad;
123 }
124 public int getVidaActual() {
125     return vidaActual;
126 }
127 }
```

Soldado.java VideoJuego.java

```
1 import java.util.*;
2 import java.util.ArrayList;
3 public class VideoJuego {
4     private static ArrayList<Soldado> ejercito1 = new ArrayList<>();
5     private static ArrayList<Soldado> ejercito2 = new ArrayList<>();
6     public static void main(String[] args) {
7         Scanner sc = new Scanner(System.in);
8         int opcion = 0;
9         while (opcion != 3) {
10             System.out.println("Menú:");
11             System.out.println("1. Juego Rápido");
12             System.out.println("2. Personalizado");
13             System.out.println("3. Salir");
14             System.out.print("Seleccione una opción: ");
15             opcion = sc.nextInt();
16             switch (opcion) {
17                 case 1:
18                     juegoRapido();
19                     break;
20                 case 2:
21                     gestionarEjercitoPersonalizado(sc, ejercito1, ejercito2);
22                     break;
23                 case 3:
24                     System.out.println("¡Hasta luego!");
25                     break;
26                 default:
27                     System.out.println("Opción no válida. Por favor, elija una opción válida.");
28                     break;
29             }
30         }
31     }
32     public static void juegoRapido() {
33         Scanner sc = new Scanner(System.in);
34         Soldado[][] tablero = new Soldado[10][10];
35         int n = (int) (Math.random() * 10) + 1;
36         int x = (int) (Math.random() * 10) + 1;
37         // Ejercito1
38         llenarSoldados1(ejercito1, n);
39         System.out.println("\nEjercito-1");
40         imprimirSoldados(ejercito1);
41         System.out.println("\nEl soldado con mayor vida es: ");
42         Soldado mayorVida = soldadoConMayorVida(ejercito1);
43         System.out.println("Nombre: " + mayorVida.getNombre());
44     }
```

```
44 System.out.println("Vida: " + mayorVida.getNivelVida());
45 double promedioVida = calcularPromedioDeVida(ejercito1);
46 System.out.println("\nPromedio de vida de todos los soldados: " + promedioVida);
47 int sumaVidas = calcularSumaDeVidas(ejercito1);
48 System.out.println("\nEl nivel de vida de todo el ejército es: " + sumaVidas);
49 ordenamientoBurbuja(ejercito1);
50 System.out.println("\n-----");
51 for (Soldado soldado : ejercito1) {
52     int fila = soldado.getFila() - 1;
53     int columna = soldado.getColumna() - 1;
54     tablero[fila][columna] = soldado;
55 }
56 //Ejercito2
57 llenarSoldados2(ejercito2, x);
58 System.out.println("\nEjercito-2");
59 imprimirSoldados(ejercito2);
60 System.out.println("\nEl soldado con mayor vida es: ");
61 Soldado mayorV = soldadoConMayorVida(ejercito2);
62 System.out.println("Nombre: " + mayorV.getNombre());
63 System.out.println("Vida: " + mayorV.getNivelVida());
64 double promedioV = calcularPromedioDeVida(ejercito2);
65 System.out.println("\nPromedio de vida de todos los soldados: " + promedioV);
66 int sumaV = calcularSumaDeVidas(ejercito2);
67 System.out.println("\nEl nivel de vida de todo el ejército es: " + sumaV);
68 ordenamientoSeleccin(ejercito2);
69 System.out.println("\nCantidad total de objetos Soldado creados: " + Soldado.getSoldCread());
70 for (Soldado soldado : ejercito2) {
71     int fila = soldado.getFila() - 1;
72     int columna = soldado.getColumna() - 1;
73     tablero[fila][columna] = soldado;
74 }
75 boolean juegoActivo = true;
76 while (juegoActivo) {
77     System.out.println("\nTablero de juego:");
78     System.out.println("\nCantidad de soldados en el Ejército 1: " + Soldado.getSEjer1());
79     System.out.println("\nCantidad de soldados en el Ejército 2: " + Soldado.getSEjer2());
80     System.out.println("\nCantidad de soldados en total: " + Soldado.getSoldCread());
81     imprimirTablero(tablero, ejercito1, ejercito2);
82     System.out.println("\nTurno del Jugador 1 - S ");
83     moverSoldado(sc, tablero, ejercito1, ejercito2);
84     if (ejercito2.isEmpty()) {
85         System.out.println("¡El Jugador 1 ha ganado!");
86         break;
87     }
88     System.out.println("\nTablero de juego:");
89     System.out.println("\nCantidad de soldados en el Ejército 1: " + Soldado.getSEjer1());
90     System.out.println("\nCantidad de soldados en el Ejército 2: " + Soldado.getSEjer2());
91     System.out.println("\nCantidad de soldados en total: " + Soldado.getSoldCread());
92     imprimirTablero(tablero, ejercito1, ejercito2);
93     System.out.println("\nTurno del Jugador 2 - C ");
94     moverSoldado(sc, tablero, ejercito2, ejercito1);
95     if (ejercito1.isEmpty()) {
96         System.out.println("¡El Jugador 2 ha ganado!");
97         break;
98     }
99 }
100 }
101 public static void imprimirTablero(Soldado[][] tablero, ArrayList<Soldado> ejercito1, ArrayList<Soldado> ejercito2) {
102     System.out.print(" ");
103     for (char columna = 'A'; columna <= 'J'; columna++) {
104         System.out.print(" " + columna);
105     }
106     System.out.println();
107     System.out.print("  _ _ _ _ _ _ _ _ _ _ ");
108     for (int i = 0; i < tablero.length; i++) {
109         System.out.print((i + 1 < 10 ? " " : "") + (i + 1) );
110         for (int j = 0; j < tablero[i].length; j++) {
111             if (tablero[i][j] != null) {
112                 Soldado soldado = tablero[i][j];
113                 char letra = (ejercito1.contains(soldado)) ? 's' : 'c';
114                 System.out.print("|" + letra + soldado.getNivelVida());
115             } else {
116                 System.out.print("|_");
117             }
118         }
119         System.out.println("|");
120     }
121 }
122 public static void llenarSoldados1(ArrayList<Soldado> soldados, int n) {
123     int cantidadSoldadosRestantes = Soldado.Soldados - soldados.size();
124     if (n > cantidadSoldadosRestantes) {
125         n = cantidadSoldadosRestantes;
126         System.out.println("Se han creado " + n + " soldados para el Ejército 1.");
127     }
128     for (int i = 0; i < n; i++) {
129         int x, d, a, f, c, v;
```

```
130 String nombre;
131 boolean casillaOcupada;
132 while (true) {
133     casillaOcupada = false;
134     x = (int)(Math.random() * 5 + 1);
135     d = (int)(Math.random() * 5 + 1);
136     a = (int)(Math.random() * 5 + 1);
137     f = (int)(Math.random() * 5 + 1);
138     c = (int)(Math.random() * 5 + 1);
139     v = (int)(Math.random() * 5 + 1);
140     nombre = "soldado" + i + "X1";
141     for (Soldado s : soldados) {
142         if (s.getFila() == f && s.getColumna() == c) {
143             casillaOcupada = true;
144             break;
145         }
146     }
147     if (!casillaOcupada) {
148         Soldado soldado = new Soldado();
149         soldado.setNombre(nombre);
150         soldado.setFila(f);
151         soldado.setColumna(c);
152         soldado.setNivelAtaque(a);
153         soldado.setNivelDefensa(d);
154         soldado.setNivelVida(x);
155         soldado.setVelocidad(v);
156         soldados.add(soldado);
157         Soldado.sumarEjer1();
158         break;
159     }
160 }
161 }
162 }
163 public static void llenarSoldados2(ArrayList<Soldado> soldados, int n) {
164     int cantidadSoldadosRestantes = Soldado.Soldados - soldados.size();
165     if (n > cantidadSoldadosRestantes) {
166         n = cantidadSoldadosRestantes;
167         System.out.println("Se han creado " + n + " soldados para el Ejército 2.");
168     }
169     for (int i = 0; i < n; i++) {
170         int x, d, a, f, c, v;
171         String nombre;
172         boolean casillaOcupada;
173         while (true) {
174             casillaOcupada = false;
175             x = (int)(Math.random() * 5 + 1);
176             d = (int)(Math.random() * 5 + 1);
177             a = (int)(Math.random() * 5 + 1);
178             f = (int)(Math.random() * 5 + 1);
179             c = (int)(Math.random() * 5 + 1);
180             v = (int)(Math.random() * 5 + 1);
181             nombre = "soldado" + i + "X2";
182             for (Soldado s : soldados) {
183                 if (s.getFila() == f && s.getColumna() == c) {
184                     casillaOcupada = true;
185                     break;
186                 }
187             }
188             if (!casillaOcupada) {
189                 Soldado soldado = new Soldado();
190                 soldado.setNombre(nombre);
191                 soldado.setFila(f);
192                 soldado.setColumna(c);
193                 soldado.setNivelAtaque(a);
194                 soldado.setNivelDefensa(d);
195                 soldado.setNivelVida(x);
196                 soldado.setVelocidad(v);
197                 soldados.add(soldado);
198                 Soldado.sumarEjer2();
199                 break;
200             }
201         }
202     }
203 }
204 public static void imprimirSoldados(ArrayList<Soldado> ejercito) {
205     if (ejercito == ejercito1) {
206         System.out.println("Cantidad de soldados en el Ejército 1: " + Soldado.getSEjer1());
207     } else if (ejercito == ejercito2) {
208         System.out.println("Cantidad de soldados en el Ejército 2: " + Soldado.getSEjer2());
209     }
210     for (Soldado soldado : ejercito) {
211         System.out.println("nombre: " + soldado.getNombre());
212         System.out.println("fila: " + soldado.getFila());
213         System.out.println("columna: " + soldado.getColumna());
214         System.out.println("vida: " + soldado.getNivelVida());
215         System.out.println("nivel de Ataque: " + soldado.getNivelAtaque());
216         System.out.println("nivel de Defensa: " + soldado.getNivelDefensa());
217         System.out.println("velocidad: " + soldado.getVelocidad());
218     }
219     System.out.println();
220 }
221 public static Soldado soldadoConMayorVida(ArrayList<Soldado> ejercito) {
222     Soldado mayorVida = null;
223     int maxVida = 0;
224     for (Soldado soldado : ejercito) {
225         if (soldado.getNivelVida() > maxVida) {
226             maxVida = soldado.getNivelVida();
227             mayorVida = soldado;
228         }
229     }
230     return mayorVida;
231 }
```

```
228     }
229     }
230     return mayorVida;
231 }
232 public static double calcularPromedioDeVida(ArrayList<Soldado> ejercito) {
233     int totalVida = 0;
234     for (Soldado soldado : ejercito) {
235         totalVida += soldado.getNivelVida();
236     }
237     return (double) totalVida / ejercito.size();
238 }
239 public static int calcularSumaDeVidas(ArrayList<Soldado> ejercito) {
240     int sumaVidas = 0;
241     for (Soldado soldado : ejercito) {
242         sumaVidas += soldado.getNivelVida();
243     }
244     return sumaVidas;
245 }
246 public static void ordenamientoBurbuja(ArrayList<Soldado> ejercito) {
247     System.out.println("\nRanking de poder por ordenamiento burbuja:");
248     int n = ejercito.size();
249     boolean ordenado;
250     for (int i = 0; i < n - 1; i++) {
251         ordenado = false;
252         for (int j = 0; j < n - 1 - i; j++) {
253             if (ejercito.get(j).getNivelVida() < ejercito.get(j + 1).getNivelVida()) {
254                 Soldado temp = ejercito.get(j);
255                 ejercito.set(j, ejercito.get(j + 1));
256                 ejercito.set(j + 1, temp);
257                 ordenado = true;
258             }
259         }
260         if (!ordenado) {
261             break;
262         }
263     }
264     for (int i = 0; i < ejercito.size(); i++) {
265         System.out.println("Posición " + (i + 1) + ": " + ejercito.get(i).getNombre() + " - Vida: " + ejercito.get(i).getNivelVida());
266     }
267 }
268 public static void ordenamientoSeleccion(ArrayList<Soldado> ejercito) {
269     System.out.println("\nRanking de poder por ordenamiento de selección:");
270     int n = ejercito.size();
271     for (int i = 0; i < n - 1; i++) {
272         int max = i;
273         for (int j = i + 1; j < n; j++) {
274             if (ejercito.get(j).getNivelVida() > ejercito.get(max).getNivelVida()) {
275                 max = j;
276             }
277         }
278         Soldado temp = ejercito.get(i);
279         ejercito.set(i, ejercito.get(max));
280         ejercito.set(max, temp);
281     }
282     for (int i = 0; i < ejercito.size(); i++) {
283         System.out.println("Posición " + (i + 1) + ": " + ejercito.get(i).getNombre() + " - Vida: " + ejercito.get(i).getNivelVida());
284     }
285 }
286 public static void moverSoldado(Scanner sc, Soldado[][] tablero, ArrayList<Soldado> ejercitoMov, ArrayList<Soldado> ejercitoOtro) {
287     System.out.print("Ingrese la coordenada del soldado a mover (por ejemplo, d3): ");
288     String coordenada = sc.next();
289     int filaMov = Integer.parseInt(coordenada.substring(1)) - 1;
290     char columnaChar = coordenada.charAt(0);
291     int columnaMov = columnaChar - 'a';
292     if (filaMov < 0 || filaMov >= tablero.length || columnaMov < 0 || columnaMov >= tablero[0].length) {
293         System.out.println("Movimiento inválido: coordenada fuera del tablero.");
294         return;
295     }
296     Soldado soldado = tablero[filaMov][columnaMov];
297     if (soldado == null || !ejercitoMov.contains(soldado)) {
298         System.out.println("Movimiento inválido: no hay un soldado del ejército correspondiente en esa posición.");
299         return;
300     }
301     System.out.print("Ingrese la dirección del movimiento (w: arriba, a: izquierda, s: abajo, d: derecha): ");
302     char direccion = sc.next().charAt(0);
303     int nuevaFila = filaMov;
304     int nuevaColumna = columnaMov;
305     if (direccion == 'w') {
306         nuevaFila--;
307     } else if (direccion == 'a') {
308         nuevaColumna--;
309     } else if (direccion == 's') {
310         nuevaFila++;
311     } else if (direccion == 'd') {
312         nuevaColumna++;
313     } else {
314         System.out.println("Movimiento inválido: dirección no válida.");
315         return;
316     }
317     if (nuevaFila < 0 || nuevaFila >= tablero.length || nuevaColumna < 0 || nuevaColumna >= tablero[0].length) {
318         System.out.println("Movimiento inválido: movimiento fuera del tablero.");
319         return;
320     }
321     if (tablero[nuevaFila][nuevaColumna] != null) {
322         Soldado soldadoEnNuevaPos = tablero[nuevaFila][nuevaColumna];
323         if (ejercitoOtro.contains(soldadoEnNuevaPos)) {
```

```
324 double probabilidadSoldado = soldado.getNivelVida() * 100.0 / (soldado.getNivelVida() + soldadoEnNuevaPos.getNivelVida());
325 double probabilidadEnemigo = soldadoEnNuevaPos.getNivelVida() * 100.0 / (soldado.getNivelVida() + soldadoEnNuevaPos.getNivelVida());
326 Soldado ganadorBatalla = batalla(soldado, soldadoEnNuevaPos);
327 System.out.println("¡Batalla!");
328 System.out.println(soldado.getNombre() + " - Vida: " + soldado.getNivelVida() + " - Probabilidad de victoria: " + probabilidadSoldado + "%");
329 System.out.println(soldadoEnNuevaPos.getNombre() + " - Vida: " + soldadoEnNuevaPos.getNivelVida() + " - Probabilidad de victoria: " + probabilidadEnemigo + "%");
330
331 if (ganadorBatalla == soldado) {
332     tablero[nuevaFila][nuevaColumna] = soldado;
333     tablero[filaMov][columnaMov] = null;
334     ejercitoOtro.remove(soldadoEnNuevaPos);
335     soldado.setNivelVida(soldado.getNivelVida() + 1);
336     if (ejercitoOtro == ejercito1) {
337         Soldado.restarEjer1();
338     } else {
339         Soldado.restarEjer2();
340     }
341 } else {
342     tablero[filaMov][columnaMov] = null;
343     ejercitoMov.remove(soldado);
344     soldadoEnNuevaPos.setNivelVida(soldadoEnNuevaPos.getNivelVida() + 1);
345 }
346 System.out.println("Ganador: " + ganadorBatalla.getNombre() + " - Vida: " + ganadorBatalla.getNivelVida());
347 if (ejercitoOtro.isEmpty()) {
348     if (ejercitoOtro == ejercito1) {
349         if (Soldado.getSEjer1() == 0) {
350             System.out.println("¡El Ejército 2 es el ganador del juego!");
351         } else {
352             System.out.println("¡El Ejército 1 es el ganador del juego!");
353         }
354     }
355 }
356 return;
357 } else {
358     System.out.println("Movimiento inválido: ya hay un soldado en la nueva posición.");
359     return;
360 }
361 }
362 tablero[filaMov][columnaMov] = null;
363 tablero[nuevaFila][nuevaColumna] = soldado;
364 }
365 public static Soldado batalla(Soldado soldado1, Soldado soldado2) {
366     int totalVida = soldado1.getNivelVida() + soldado2.getNivelVida();
367     double porcentaje1 = (soldado1.getNivelVida() * 100.0) / totalVida;
368     double porcentaje2 = (soldado2.getNivelVida() * 100.0) / totalVida;
369     int numeroAleatorio = (int) (Math.random() * 100) + 1;
370     if (numeroAleatorio <= porcentaje1) {
371         return soldado1;
372     } else {
373         return soldado2;
374     }
375 }
376 public static void iCaracteristicas(Soldado soldado) {
377     System.out.println("Características del soldado:");
378     System.out.println("Nombre: " + soldado.getNombre());
379     System.out.println("Fila: " + soldado.getFila());
380     System.out.println("Columna: " + soldado.getColumna());
381     System.out.println("Vida: " + soldado.getNivelVida());
382     System.out.println("Nivel de Ataque: " + soldado.getNivelAtaque());
383     System.out.println("Nivel de Defensa: " + soldado.getNivelDefensa());
384     System.out.println("Velocidad: " + soldado.getVelocidad());
385 }
386 public static void gestionarEjercitoPersonalizado(Scanner sc, ArrayList<Soldado> ejercito1, ArrayList<Soldado> ejercito2) {
387     int ejercitoSeleccionado;
388     while (true) {
389         System.out.println("Seleccione el ejército a gestionar (1 o 2): ");
390         ejercitoSeleccionado = sc.nextInt();
391         if (ejercitoSeleccionado == 1 || ejercitoSeleccionado == 2) {
392             break;
393         }
394     }
395     ArrayList<Soldado> ejercitoActual = (ejercitoSeleccionado == 1) ? ejercito1 : ejercito2;
396     int opcion = 0;
397     while (opcion != 11) {
398         System.out.println("Menú de gestión de ejército " + ejercitoSeleccionado + ":");
399         System.out.println("1. Crear Soldado");
400         System.out.println("2. Eliminar Soldado");
401         System.out.println("3. Clonar Soldado");
402         System.out.println("4. Modificar Soldado");
403         System.out.println("5. Comparar Soldados");
404         System.out.println("6. Intercambiar Soldados");
405         System.out.println("7. Ver Soldado (Búsqueda por nombre)");
406         System.out.println("8. Ver Ejército");
407         System.out.println("9. Sumar Niveles");
408         System.out.println("10. Jugar");
409         System.out.println("11. Volver");
410         System.out.print("Seleccione una opción: ");
411         opcion = sc.nextInt();
412         switch (opcion) {
413             case 1:
414                 crearSoldado(sc, ejercitoActual);
415                 break;
416             case 2:
417                 eliminarSoldado(sc, ejercitoActual);
418                 break;
419             case 3:
420                 clonarSoldado(sc, ejercitoActual);
421                 break;
```

	<p style="text-align: center;">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p style="text-align: center;">Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLE-001</p>	<p>Página: 8</p>

```

422         case 4:
423             modificarSoldado(sc, ejercitoActual);
424             break;
425         case 5:
426             compararSoldados(sc, ejercitoActual);
427             break;
428         case 6:
429             intercambiarSoldados(sc, ejercitoActual);
430             break;
431         case 7:
432             buscarSoldadoPorNombre(sc, ejercitoActual);
433             break;
434         case 8:
435             verEjercito(ejercitoActual);
436             break;
437         case 9:
438             sumarNiveles(ejercitoActual);
439             break;
440         case 10:
441             jugar();
442             break;
443         case 11:
444             System.out.println("Volviendo al menú principal...");
445             break;
446         default:
447             System.out.println("Opción no válida. Por favor, elija una opción válida.");
448     }
449 }
450
451 public static void crearSoldado(Scanner sc, ArrayList<Soldado> ejercito) {
452     if (ejercito.size() >= 10) {
453         System.out.println("El ejército está completo. No se pueden agregar más soldados.");
454         return;
455     }
456     int numeroSoldado = ejercito.size() + 1;
457     String nombre = "soldado" + numeroSoldado + (ejercito == ejercito1 ? "X1" : "X2");
458     int x = (int)(Math.random() * 5 + 1);
459     int d = (int)(Math.random() * 5 + 1);
460     int a = (int)(Math.random() * 5 + 1);
461     int f = (int)(Math.random() * 5 + 1);
462     int c = (int)(Math.random() * 5 + 1);
463     int v = (int)(Math.random() * 5 + 1);
464     Soldado soldado = new Soldado();
465     soldado.setNombre(nombre);
466     soldado.setFila(f);
467     soldado.setColumna(c);
468     soldado.setNivelAtaque(a);
469     soldado.setNivelDefensa(d);
470     soldado.setNivelVida(x);
471     soldado.setVelocidad(v);
472     ejercito.add(soldado);
473     System.out.println("Soldado " + soldado.getNombre() + " creado y agregado al ejército.");
474 }
475 public static void eliminarSoldado(Scanner sc, ArrayList<Soldado> ejercito) {
476     if (ejercito.isEmpty()) {
477         System.out.println("El ejército está vacío. No se pueden eliminar soldados.");
478         return;
479     }
480     System.out.print("Ingrese el nombre del soldado que desea eliminar: ");
481     String nombreSoldado = sc.next();
482     boolean encontrado = false;
483     for (Soldado soldado : ejercito) {
484         if (soldado.getNombre().equals(nombreSoldado)) {
485             ejercito.remove(soldado);
486             System.out.println("Soldado " + nombreSoldado + " eliminado del ejército.");
487             encontrado = true;
488             break;
489         }
490     }
491     if (!encontrado) {
492         System.out.println("Soldado no encontrado en el ejército.");
493     }
494 }
495 public static void clonarSoldado(Scanner sc, ArrayList<Soldado> ejercito) {
496     if (ejercito.isEmpty()) {
497         System.out.println("El ejército está vacío. No se pueden clonar soldados.");
498         return;
499     }
500     System.out.print("Ingrese el nombre del soldado que desea clonar: ");
501     String nombreSoldado = sc.next();
502     Soldado soldadoOriginal = null;
503     for (Soldado soldado : ejercito) {
504         if (soldado.getNombre().equals(nombreSoldado)) {
505             soldadoOriginal = soldado;
506             break;
507         }
508     }
509 }

```


	<p style="text-align: center;">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p style="text-align: center;">Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLE-001</p>	<p>Página: 9</p>

```

508     }
509     if (soldadoOriginal == null) {
510         System.out.println("Soldado no encontrado en el ejército.");
511         return;
512     }
513     if (ejercito.size() >= 10) {
514         System.out.println("El ejército ya tiene el máximo de soldados (10). No se pueden clonar más.");
515         return;
516     }
517     Soldado copiaSoldado = new Soldado();
518     copiaSoldado.setNombre(soldadoOriginal.getNombre());
519     copiaSoldado.setNivelAtaque(soldadoOriginal.getNivelAtaque());
520     copiaSoldado.setNivelDefensa(soldadoOriginal.getNivelDefensa());
521     copiaSoldado.setNivelVida(soldadoOriginal.getNivelVida());
522     copiaSoldado.setVelocidad(soldadoOriginal.getVelocidad());
523     copiaSoldado.setFila(soldadoOriginal.getFila());
524     copiaSoldado.setColumna(soldadoOriginal.getColumna());
525     ejercito.add(copiaSoldado);
526     System.out.println("Soldado " + copiaSoldado.getNombre() + " clonado y añadido al ejército.");
527 }
528 public static void modificarSoldado(Scanner sc, ArrayList<Soldado> ejercito) {
529     if (ejercito.isEmpty()) {
530         System.out.println("El ejército está vacío. No se pueden modificar soldados.");
531         return;
532     }
533     System.out.print("Ingrese el nombre del soldado que desea modificar: ");
534     String nombreSoldado = sc.next();
535     Soldado soldadoAModificar = null;
536     for (Soldado soldado : ejercito) {
537         if (soldado.getNombre().equals(nombreSoldado)) {
538             soldadoAModificar = soldado;
539             break;
540         }
541     }
542     if (soldadoAModificar == null) {
543         System.out.println("Soldado no encontrado en el ejército.");
544         return;
545     }
546     System.out.println("Seleccione qué atributo desea modificar:");
547     System.out.println("1. Nivel de Ataque");
548     System.out.println("2. Nivel de Defensa");
549     System.out.println("3. Vida Actual");
550     int opcionAtributo = sc.nextInt();
551     switch (opcionAtributo) {
552         case 1:
553             System.out.print("Ingrese el nuevo nivel de ataque: ");
554             int nuevoNivelAtaque = sc.nextInt();
555             soldadoAModificar.setNivelAtaque(nuevoNivelAtaque);
556             System.out.println("Nivel de Ataque modificado.");
557             break;
558         case 2:
559             System.out.print("Ingrese el nuevo nivel de defensa: ");
560             int nuevoNivelDefensa = sc.nextInt();
561             soldadoAModificar.setNivelDefensa(nuevoNivelDefensa);
562             System.out.println("Nivel de Defensa modificado.");
563             break;
564         case 3:
565             System.out.print("Ingrese la nueva vida actual: ");
566             int nuevaVidaActual = sc.nextInt();
567             soldadoAModificar.setVidaActual(nuevaVidaActual);
568             System.out.println("Vida Actual modificada.");
569             break;
570         default:
571             System.out.println("Opción no válida. No se ha modificado ningún atributo.");
572     }
573 }
574 public static void compararSoldados(Scanner sc, ArrayList<Soldado> ejercito) {
575     if (ejercito.isEmpty()) {
576         System.out.println("El ejército está vacío. No se pueden comparar soldados.");
577         return;
578     }
579     System.out.print("Ingrese el nombre del primer soldado que desea comparar: ");
580     String nombreSoldado1 = sc.next();
581     System.out.print("Ingrese el nombre del segundo soldado que desea comparar: ");
582     String nombreSoldado2 = sc.next();
583     Soldado soldado1 = null;
584     Soldado soldado2 = null;
585     for (Soldado soldado : ejercito) {
586         if (soldado.getNombre().equals(nombreSoldado1)) {
587             soldado1 = soldado;
588         }
589         if (soldado.getNombre().equals(nombreSoldado2)) {
590             soldado2 = soldado;
591         }
592     }
593     if (soldado1 == null || soldado2 == null) {

```

```

594         System.out.println("Al menos uno de los soldados no se encuentra en el ejército.");
595         return;
596     }
597     boolean sonIguales = (
598         soldado1.getNivelVida() == soldado2.getNivelVida() &&
599         soldado1.getNivelAtaque() == soldado2.getNivelAtaque() &&
600         soldado1.getNivelDefensa() == soldado2.getNivelDefensa() &&
601         soldado1.getVelocidad() == soldado2.getVelocidad()
602     );
603     if (sonIguales) {
604         System.out.println("Los soldados son idénticos en los atributos seleccionados.");
605     } else {
606         System.out.println("Los soldados son diferentes en al menos uno de los atributos seleccionados.");
607     }
608 }
609 public static void intercambiarSoldados(Scanner sc, ArrayList<Soldado> ejercito) {
610     if (ejercito.isEmpty()) {
611         System.out.println("El ejército está vacío. No se pueden intercambiar soldados.");
612         return;
613     }
614     System.out.print("Ingrese el nombre del primer soldado que desea intercambiar: ");
615     String nombreSoldado1 = sc.next();
616     System.out.print("Ingrese el nombre del segundo soldado que desea intercambiar: ");
617     String nombreSoldado2 = sc.next();
618     Soldado soldado1 = null;
619     Soldado soldado2 = null;
620     for (Soldado soldado : ejercito) {
621         if (soldado.getNombre().equals(nombreSoldado1)) {
622             soldado1 = soldado;
623         }
624         if (soldado.getNombre().equals(nombreSoldado2)) {
625             soldado2 = soldado;
626         }
627     }
628     if (soldado1 == null || soldado2 == null) {
629         System.out.println("Al menos uno de los soldados no se encuentra en el ejército.");
630         return;
631     }
632     int indiceSoldado1 = ejercito.indexOf(soldado1);
633     int indiceSoldado2 = ejercito.indexOf(soldado2);
634     ejercito.set(indiceSoldado1, soldado2);
635     ejercito.set(indiceSoldado2, soldado1);
636     System.out.println("Los soldados han sido intercambiados con éxito.");
637 }
638 public static void buscarSoldadoPorNombre(Scanner sc, ArrayList<Soldado> ejercito) {
639     if (ejercito.isEmpty()) {
640         System.out.println("El ejército está vacío. No se pueden buscar soldados.");
641         return;
642     }
643     System.out.print("Ingrese el nombre del soldado que desea buscar: ");
644     String nombreSoldado = sc.next();
645     Soldado soldadoEncontrado = null;
646     for (Soldado soldado : ejercito) {
647         if (soldado.getNombre().equals(nombreSoldado)) {
648             soldadoEncontrado = soldado;
649             break;
650         }
651     }
652     if (soldadoEncontrado != null) {
653         iCaracteristicas(soldadoEncontrado);
654     } else {
655         System.out.println("Soldado no encontrado en el ejército.");
656     }
657 }
658 public static void verEjercito(ArrayList<Soldado> ejercito) {
659     if (ejercito.isEmpty()) {
660         System.out.println("El ejército está vacío.");
661         return;
662     }
663     System.out.println("Soldados en el ejército:");
664     for (Soldado soldado : ejercito) {
665         iCaracteristicas(soldado);
666         System.out.println("-----");
667     }
668 }
669 public static void sumarNiveles(ArrayList<Soldado> ejercito) {
670     if (ejercito.isEmpty()) {
671         System.out.println("El ejército está vacío. No se pueden sumar niveles.");
672         return;
673     }
674     int sumaNivelVida = 0;
675     int sumaNivelAtaque = 0;
676     int sumaNivelDefensa = 0;
677     int sumaVelocidad = 0;
678     for (Soldado soldado : ejercito) {
679         sumaNivelVida += soldado.getNivelVida();

```

```

680 sumaNivelAtaque += soldado.getNivelAtaque();
681 sumaNivelDefensa += soldado.getNivelDefensa();
682 sumaVelocidad += soldado.getVelocidad();
683 }
684 System.out.println("Suma de niveles en el ejército:");
685 System.out.println("Suma de Nivel de Vida: " + sumaNivelVida);
686 System.out.println("Suma de Nivel de Ataque: " + sumaNivelAtaque);
687 System.out.println("Suma de Nivel de Defensa: " + sumaNivelDefensa);
688 System.out.println("Suma de Velocidad: " + sumaVelocidad);
689 }
690 public static void jugar() {
691     Scanner sc = new Scanner(System.in);
692     Soldado[][] tablero = new Soldado[10][10];
693     int n = (int) (Math.random() * 10) + 1;
694     int x = (int) (Math.random() * 10) + 1;
695     // Ejercito1
696     llenarSoldados1(ejercito1, n);
697     System.out.println("\nEjercito-1");
698     imprimirSoldados(ejercito1);
699     System.out.println("\nEl soldado con mayor vida es: ");
700     Soldado mayorVida = soldadoConMayorVida(ejercito1);
701     System.out.println("Nombre: " + mayorVida.getNombre());
702     System.out.println("Vida: " + mayorVida.getNivelVida());
703     double promedioVida = calcularPromedioDeVida(ejercito1);
704     System.out.println("\nPromedio de vida de todos los soldados: " + promedioVida);
705     int sumaVidas = calcularSumaDeVidas(ejercito1);
706     System.out.println("\nEl nivel de vida de todo el ejército es: " + sumaVidas);
707     ordenamiento Burbuja(ejercito1);
708     System.out.println("\n-----");
709     for (Soldado soldado : ejercito1) {
710         int fila = soldado.getFila() - 1;
711         int columna = soldado.getColumna() - 1;
712         tablero[fila][columna] = soldado;
713     }
714     //Ejercito2
715     llenarSoldados2(ejercito2, x);
716     System.out.println("\nEjercito-2");
717     imprimirSoldados(ejercito2);
718     System.out.println("\nEl soldado con mayor vida es: ");
719     Soldado mayorV = soldadoConMayorVida(ejercito2);
720     System.out.println("Nombre: " + mayorV.getNombre());
721     System.out.println("Vida: " + mayorV.getNivelVida());
722     double promedioV = calcularPromedioDeVida(ejercito2);
723     double promedioV = calcularPromedioDeVida(ejercito2);
724     System.out.println("\nPromedio de vida de todos los soldados: " + promedioV);
725     int sumaV = calcularSumaDeVidas(ejercito2);
726     System.out.println("\nEl nivel de vida de todo el ejército es: " + sumaV);
727     ordenamientoSeleccion(ejercito2);
728     System.out.println("\nCantidad total de objetos Soldado creados: " + Soldado.getSoldCread());
729     for (Soldado soldado : ejercito2) {
730         int fila = soldado.getFila() - 1;
731         int columna = soldado.getColumna() - 1;
732         tablero[fila][columna] = soldado;
733     }
734     boolean juegoActivo = true;
735     while (juegoActivo) {
736         System.out.println("\nTablero de juego:");
737         System.out.println("\nCantidad de soldados en el Ejército 1: " + Soldado.getSEjer1());
738         System.out.println("Cantidad de soldados en el Ejército 2: " + Soldado.getSEjer2());
739         System.out.println("Cantidad de soldados en total: " + Soldado.getSoldCread());
740         imprimirTablero(tablero, ejercito1, ejercito2);
741         System.out.println("Turno del Jugador 1 - S ");
742         moverSoldado(sc, tablero, ejercito1, ejercito2);
743         if (ejercito2.isEmpty()) {
744             System.out.println("¡El Jugador 1 ha ganado!");
745             break;
746         }
747         System.out.println("Tablero de juego:");
748         System.out.println("\nCantidad de soldados en el Ejército 1: " + Soldado.getSEjer1());
749         System.out.println("Cantidad de soldados en el Ejército 2: " + Soldado.getSEjer2());
750         System.out.println("Cantidad de soldados en total: " + Soldado.getSoldCread());
751         imprimirTablero(tablero, ejercito1, ejercito2);
752         System.out.println("Turno del Jugador 2 - C ");
753         moverSoldado(sc, tablero, ejercito2, ejercito1);
754         if (ejercito1.isEmpty()) {
755             System.out.println("¡El Jugador 2 ha ganado!");
756             break;
757         }
758     }
759 }

```

	<p>UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p>Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLE-001</p>	<p>Página: 12</p>

videojuego (1) Java Application] C:\Users\ORIN\juegos\Downloads\compso java 2023

Menú:

1. Juego Rápido
2. Personalizado
3. Salir

Seleccione una opción: 1

Ejercito-1

Cantidad de soldados en el Ejército 1: 4

nombre: soldado0X1

fila: 2

columna: 1

vida: 4

nivel de Ataque: 1

nivel de Defensa: 5

velocidad: 5

nombre: soldado1X1

fila: 3

columna: 1

vida: 1

nivel de Ataque: 1

nivel de Defensa: 2

velocidad: 5

nombre: soldado2X1

fila: 5

columna: 2

vida: 2

nivel de Ataque: 1

nivel de Defensa: 1

velocidad: 5

nombre: soldado3X1

fila: 3

columna: 5

vida: 5

nivel de Ataque: 2

nivel de Defensa: 5

velocidad: 4



UNIVERSIDAD NACIONAL DE SAN AGUSTIN
FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS
ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA



Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación

Aprobación: 2022/03/01

Código: GUIA-PRLE-001

Página: 13

II. PRUEBA

	<p align="center">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		
Aprobación: 2022/03/01	Código: GUIA-PRLE-001	Página: 14

III. CUESTIONARIO:

REPOSITORIO: <https://github.com/UJKMjuandi/FundamentosP2>

CONCLUSIONES
<p>La expansión del programa para enfrentamientos entre jugadores humanos ha mejorado la experiencia del usuario mediante la introducción de métodos estratégicos. La selección de soldados, la planificación de movimientos y la gestión de turnos se han optimizado, resaltando la importancia de los atributos de la clase, como nivelAtaque y vive. La interfaz de usuario se ha mejorado, permitiendo una presentación clara de información. La gestión de eventos y resultados se ha afinado, utilizando atributos como vidaActual. La evaluación de fortalezas y debilidades se ha intensificado, destacando la relevancia de los atributos para estrategias tácticas, enriqueciendo así la experiencia de juego. En resumen, la sólida estructura de clase ha sido crucial para estas mejoras y para permitir una experiencia de juego más interactiva y estratégica.</p>

METODOLOGÍA DE TRABAJO
<ol style="list-style-type: none"> 1.- Leer los enunciados cuidadosamente para saber que requiere el problemas. 2.- verificar el diagrama uml puesto en la practica 3.-Empezar a hacer las clase soldado y la clase VideoJuego 4.- Realizar el Videojuego con los métodos apropiados.

	UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA	
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		
Aprobación: 2022/03/01	Código: GUIA-PRLE-001	Página: 15

REFERENCIAS Y BIBLIOGRAFÍA

[1] M. A. Lopez, E. Castro Gutierrez, *Fundamentos de la Programación 2 Topicos de Programación Orientada a Objetos*. Arequipa: UNSA, 2021

RUBRICA PARA EL CONTENIDO DEL INFORME Y DEMOSTRACIÓN

El alumno debe marcar o dejar en blanco en celdas de la columna Checklist si cumplió 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 los ítems.

El alumno debe autocalificarse en la columna Estudiante de acuerdo a la siguiente tabla:

Tabla 1: 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

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	2	
3. Código fuente	Hay porciones de código fuente importantes con numeración y explicaciones detalladas de sus funciones.	2	X	1	
4. Ejecución	Se incluyen ejecuciones/pruebas del código fuente explicadas gradualmente.	2	X	1	
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 están dentro de los plazos de fecha de entrega	2	X	0	

	<p style="text-align: center;">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p style="text-align: center;">Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLE-001</p>	<p>Página: 16</p>

	establecidos.				
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	3	
TOTAL		20		13	

Tabla 2: Rúbrica para contenido del Informe y demostración