	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 - Menú</i>				
NÚMERO DE PRÁCTICA:	<i>12</i>	AÑO LECTIVO:	<i>2023</i>	NRO. SEMESTRE:	<i>2do Semestre</i>
FECHA DE PRESENTACIÓN	<i>27/12/2023</i>	HORA DE PRESENTACIÓN	<i>13/00/00</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 Laboratorio12/src/Soldado.java
2 public class Soldado {
3     private String nombre;
4     private int nivelAtaque;
5     private int nivelDefensa;
6     private int nivelVida;
7     private int velocidad;
8     private String actitud;
9     private int vidaActual;
10    private boolean vive;
11    private int posFila;
12    private char posCol;
13    private char figura;
14    public Soldado(String n) {
15        nombre = n;
16    }
17    public int atacar() {
18        actitud = "ofensiva";
19        return avanzar();
20    }
21    public void defender() {
22        velocidad = 0;
23        actitud = "defensiva";
24    }
25    public int avanzar() {
26        return velocidad+1;
27    }
28    public void retroceder() {
29        defender();
30        actitud = "defensiva";
31        velocidad = velocidad - 1;
32    }
33    public void serAtacado() {
34        nivelVida = nivelVida - 1;
35    }
36    public void huir() {
37        actitud = "fuga";
38    }
39    public void morir() {
40        actitud = "muerto";
41        nivelVida = 0;
42    }
43    //sets
44    public void setAtaque(int ataque){
45        nivelAtaque = ataque;
46    }
47
48    public void setDefensa(int defensa){
49        nivelDefensa = defensa;
50    }
51    public void setNivVidAct(int vida){
52        vidaActual = vida;
53    }
54    public void setVidaActual(int a) {
55        nivelVida = a;
56    }
57    public void setFila(int n){
58        posFila = n;
59    }
60    public void setCol(char n){
61        posCol = n;
62    }
63    //cuatro constructores sobrecargados
64    public Soldado(String nom, char fig){
65        nombre = nom;
66        nivelAtaque = (int)(Math.random() * 5 + 1);
67        nivelDefensa = (int)(Math.random() * 5 + 1);
```

```

68     nivelVida = (int)(Math.random() * 5 + 1);
69     vidaActual = nivelVida;
70     velocidad = 0;
71     actitud = "Defensiva";
72     vive = true;
73     posFila = (int)(Math.random() * 10 + 1);
74     posCol = numCol();
75     figura = fig;
76 }
77 public Soldado(String nom, int nivAtaq, int nivDef, int vid, char fig){
78     nombre = nom;
79     nivelAtaque = nivAtaq;
80     nivelDefensa = nivDef;
81     nivelVida = vid;
82     vidaActual = vid;
83     velocidad = 0;
84     actitud = "Defensiva";
85     vive = true;
86     posFila = (int)(Math.random() * 10 + 1);
87     posCol = numCol();
88     figura = fig;
89 }
90 public Soldado(String nom, int nivAtaq, int nivDef, int nivVida, int nivAct, int vel,
91     String act, boolean vivir, int pFila, char pCol, char fig){
92     nombre = nom;
93     nivelAtaque = nivAtaq;
94     nivelDefensa = nivDef;
95     nivelVida = nivVida;
96     vidaActual = nivAct;
97     velocidad = vel;
98     actitud = act;
99     vive = vivir;
100    posFila = pFila;
101    posCol = pCol;
102    figura = fig;
103 }
104 public Soldado(String nom, int nivelVida, char fig){
105     nombre = nom;
106     nivelAtaque = (int)(Math.random() * 5 + 1);
107     nivelDefensa = (int)(Math.random() * 5 + 1);
108     this.nivelVida = nivelVida;
109     vidaActual = nivelVida;
110     velocidad = 0;
111     actitud = "Defensiva";
112     vive = true;
113     posFila = (int)(Math.random() * 10 + 1);
114     posCol = numCol();
115     figura = fig;
116 }
117 public static char numCol(){
118     String a = "abcdefghij";
119     int n = (int)(Math.random() * a.length());
120     char car = a.charAt(n);
121     return car;
122 }
123 public String toString(){
124     return "Nombre: " + nombre + " Vida:" + vidaActual + " Fila:" + posFila +
125         " Columna:" + posCol + " Actitud:" + actitud;
126 }
127 //gets
128 public String getNombre(){
129     return nombre;
130 }
131 public int getAtaque(){
132     return nivelAtaque;
133 }
134 public int getDefensa(){
135     return nivelDefensa;
136 }
137 public int getNivVidAct(){
138     return vidaActual;
139 }

```

	<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: 4</p>

```

140 public int getVida(){
141     return nivelVida;
142 }
143 public int getVelocidad(){
144     return velocidad;
145 }
146 public String getActitud(){
147     return actitud;
148 }
149 public boolean getVive(){
150     return vive;
151 }
152 public int getFila(){
153     return posFila;
154 }
155 public char getColumna(){
156     return posCol;
157 }
158 public char getFigura(){
159     return figura;
160 }
161 public void addVida(){
162     vidaActual++;
163 }
164 }
165 }
166

```

```
*Soldado.java  VideoJuego.java  Movimiento.java X
1 public class Movimiento {
2     public static int movFila(int fila, int mov){
3         switch(mov){
4             case 1: return fila -= 1;
5             case 2: return fila -= 1;
6             case 3: return fila -= 1;
7             case 5: return fila += 1;
8             case 6: return fila += 1;
9             case 7: return fila += 1;
10            default: return fila;
11        }
12    }
13    public static int restFila(int fila, int mov){
14        switch(mov){
15            case 1: return fila += 1;
16            case 2: return fila += 1;
17            case 3: return fila += 1;
18            case 5: return fila -= 1;
19            case 6: return fila -= 1;
20            case 7: return fila -= 1;
21            default: return fila;
22        }
23    }
24    public static int movColumna(int columna, int mov){
25        switch(mov){
26            case 3: return columna += 1;
27            case 4: return columna += 1;
28            case 5: return columna += 1;
29            case 1: return columna -= 1;
30            case 7: return columna -= 1;
31            case 8: return columna -= 1;
32            default: return columna;
33        }
34    }
35    public static int restColumna(int columna, int mov){
36        switch(mov){
37            case 3: return columna -= 1;
38            case 4: return columna -= 1;
39            case 5: return columna -= 1;
40            case 1: return columna += 1;
41            case 7: return columna += 1;
42            case 8: return columna += 1;
43            default: return columna;
44        }
45    }
46 }
```

[illegible]

```

87         case 2:
88             System.out.println("Eliminar soldado:");
89             eliminarSoldado(ejercitoElegido, tablero);
90             break;
91         case 3:
92             System.out.println("Clonar soldado:");
93             clonarSoldado(ejercitoElegido, tablero);
94             break;
95         case 4:
96             System.out.println("Modificar soldado:");
97             modificarSoldado(ejercitoElegido);
98             break;
99         case 5:
100             System.out.println("Comparar soldados:");
101             compararSoldados(ejercitoElegido);
102             break;
103         case 6:
104             System.out.println("Intercambiar soldados:");
105             intercambiarSoldados(ejercitoElegido);
106             break;
107         case 7:
108             System.out.println("Ver Soldado:");
109             verSoldado(ejercitoElegido);
110             break;
111         case 8:
112             System.out.println("Ver Ejercito:");
113             verEjercito(ejercitoElegido);
114             break;
115         case 9:
116             System.out.println("Sumar niveles:");
117             sumarNiveles(ejercitoElegido);
118             break;
119         case 10:
120             juego(ejercito1, ejercito2, tablero);
121         case 11:
122             System.out.println("Redirigiendo al menu principal");
123             validez = false;
124             break;
125         default:
126             System.out.println("Opcion invalida");
127     }
128 }
129 }

130 public static void addSoldado(ArrayList<Soldado> ejercito, char[][] tablero){
131     Scanner sc = new Scanner(System.in);
132     if(ejercito.size() == 10){
133         System.out.println("Ya no es posible agregar un soldado a este ejercito ya que cuenta con 10 soldados");
134     } else {
135         System.out.println("Ingrese el nombre");
136         String nombre = sc.next();
137         System.out.println("Ingrese el nivel de Vida");
138         int nivVida = sc.nextInt();
139         Soldado nuevo = new Soldado(nombre, nivVida, ejercito.get(0).getFigura());
140         ejercito.add(nuevo);
141         actTablero(ejercito, tablero);
142     }
143 }

144 public static void eliminarSoldado(ArrayList<Soldado> ejercito, char[][] tablero){
145     Scanner sc = new Scanner(System.in);
146     if(ejercito.size() == 1){
147         System.out.println("No es posible eliminar ya que este es el ultimo soldado de este ejercito");
148     } else {
149         System.out.println("Datos de los Soldados");
150         imprimirDatosEjercito(ejercito);
151         System.out.println("Que soldado desea eliminar");
152         int indice = sc.nextInt();
153         System.out.println("Se eliminara el soldado Mro " + indice);
154         System.out.println("Con datos: " + ejercito.get(indice));
155         ejercito.remove(indice);
156         actTablero(ejercito, tablero);
157     }
158 }

159 public static void clonarSoldado(ArrayList<Soldado> ejercito, char[][] tablero){
160     Scanner sc = new Scanner(System.in);
161     if(ejercito.size() == 10){
162         System.out.println("No es posible clonar porque el ejercito ya cuenta con 10 soldados");
163     } else {
164         System.out.println("Datos de los Soldados");
165         imprimirDatosEjercito(ejercito);
166         System.out.println("Que soldado desea clonar");
167         int indice = sc.nextInt();
168         Soldado original = ejercito.get(indice);
169         Soldado copia = new Soldado(original.getNombre(), original.getAtaque(), original.getDefensa(), original.getVida(), original.getNivVidAct(), original.getVelocidad(), o
170         ejercito.add(copia);
171         actTablero(ejercito, tablero);
172     }
173 }

174 public static void modificarSoldado(ArrayList<Soldado> ejercito){
175     Scanner sc = new Scanner(System.in);
176     int valor;
177     System.out.println("Datos de los Soldados");
178     imprimirDatosEjercito(ejercito);
179     System.out.println("Que soldado deseas modificar");
180     int indice = sc.nextInt();
181     System.out.println("Opcion 1: Nivel de Ataque\nOpcion 2: Nivel de defensa\nOpcion 3: Vida Actual");
182     System.out.println("Ingrese la opcion que desea modificar");
183     int opcion = sc.nextInt();

```

```

184 System.out.println("Ingrese la cantidad que quiere cambiar.");
185 valor = sc.nextInt();
186 if(opcion == 1){
187     if(valor <= 5 && valor>= 1){
188         System.out.println("Cambio realizado");
189         ejercito.get(indice).setAtaque(valor);
190     } else {
191         System.out.println("No es posible realizar el cambio porque rebasa los limites");
192     }
193 } else if(opcion == 2){
194     if(valor <= 5 && valor>= 1){
195         System.out.println("Cambio realizado");
196         ejercito.get(indice).setDefensa(valor);
197     } else {
198         System.out.println("No es posible realizar el cambio porque rebasa los limites");
199     }
200 } else if(opcion == 3){
201     if(valor <= 5 && valor>= 1){
202         System.out.println("Cambio realizado");
203         ejercito.get(indice).setNivVidAct(valor);
204     } else {
205         System.out.println("No es posible realizar el cambio porque rebasa los limites");
206     }
207 } else{
208     System.out.println("Opcion invalida");
209 }
210 }
211 public static void compararSoldados(ArrayList<Soldado> ejercito){
212     Scanner sc = new Scanner(System.in);
213     System.out.println("Datos de los soldados");
214     imprimirDatosEjercito(ejercito);
215     System.out.println("Escoja el indice del 1er soldado");
216     int primero = sc.nextInt();
217     System.out.println("Escoja el indice del 2do soldado");
218     int segundo = sc.nextInt();
219     Soldado uno = ejercito.get(primero);
220     Soldado dos = ejercito.get(segundo);
221     if(comparar(uno, dos)){
222         System.out.println("Son iguales");
223     } else {
224         System.out.println("No son iguales");
225     }
226 }
227 public static boolean comparar(Soldado uno, Soldado dos){
228     return uno.getNombre().equals(dos.getNombre()) && uno.getAtaque() == dos.getAtaque() && uno.getDefensa() == dos.getDefensa() && uno.getVive() == dos.getVive() && uno.getV
229 }
230 public static void intercambiarSoldados(ArrayList<Soldado> ejercito){
231     Scanner sc = new Scanner(System.in);
232     System.out.println("Datos de los soldados");
233     imprimirDatosEjercito(ejercito);
234     System.out.println("Escoja el indice del 1er soldado");
235     int primero = sc.nextInt();
236     System.out.println("Escoja el indice del 2do soldado");
237     int segundo = sc.nextInt();
238     Soldado original = ejercito.get(primero);
239     Soldado copia = new Soldado(original.getNombre(), original.getAtaque(), original.getDefensa(), original.getVida(), original.getNivVidAct(), original.getVelocidad(), origi
240     ejercito.set(primero, ejercito.get(segundo));
241     ejercito.set(segundo, copia);
242     System.out.println("Intercambio realizado");
243 }
244 public static void verSoldado(ArrayList<Soldado> lista){
245     Scanner sc = new Scanner(System.in);
246     int indice;
247     System.out.println("Ingrese el nombre que esta buscando");
248     String nombre = sc.next();
249     if(nombreUbicado(lista, nombre)){
250         for(int i = 0; i < lista.size(); i++){
251             if(lista.get(i).getNombre().equals(nombre)){
252                 indice = i;
253                 System.out.println(lista.get(i));
254                 break;
255             }
256         }
257     } else {
258         System.out.println("No se encontro a nadie con ese nombre");
259     }
260 }
261 public static void verEjercito(ArrayList<Soldado> ejercito){
262     System.out.println("Datos de los soldados");
263     imprimirDatosEjercito(ejercito);
264 }
265 public static boolean nombreUbicado(ArrayList<Soldado> lista, String nombre){
266     for(int i = 0; i < lista.size(); i++){
267         if(lista.get(i).getNombre().equals(nombre)){
268             return true;
269         }
270     }

```



```

270     }
271     return false;
272 }
273 public static ArrayList<Soldado> elegido(ArrayList<Soldado> ejercito1, ArrayList<Soldado> ejercito2){
274     Scanner sc = new Scanner(System.in);
275     System.out.println("Opcion 1 : Ejercito *\nOpcion 2: Ejercico $");
276     int opcion = sc.nextInt();
277     if(opcion == 1){
278         return ejercito1;
279     } else {
280         return ejercito2;
281     }
282 }
283 public static void sumarNiveles(ArrayList<Soldado> ejercito){
284     int valorVidaT = 0;
285     int valorAtaqueT = 0;
286     int valorDefensaT = 0;
287     int valorVelocidadT = 0;
288     for(Soldado m: ejercito){
289         valorVidaT += m.getVida();
290         valorAtaqueT += m.getAtaque();
291         valorDefensaT += m.getDefensa();
292         valorVelocidadT += m.getVelocidad();
293     }
294     System.out.println("El valor de vida total es : " + valorVidaT + "\nEl valor de ataque total es : " + valorAtaqueT +
295     "\nEl valor de defensa total es : " + valorDefensaT + "\n El valor de velocidad total es : " + valorVelocidadT);
296 }
297 public static void juego(ArrayList<Soldado> ejercito1, ArrayList<Soldado> ejercito2, char[][] tablero){
298     Scanner sc = new Scanner(System.in);
299     boolean validez = true;
300     String coordenada;
301     int movimiento, fila, columna;
302     String jugar = "";
303     int i = 0;
304     while(validez){
305         if(ejercito1.size() == 0 || ejercito2.size() == 0 || jugar.equals("NO")){
306             validez = false;
307             break;
308         } else if(i % 2 == 0){
309             System.out.println("Turno del Ejercito 1");
310             coordenada = ingresar(ejercito1.get(0).getFigura(), tablero);
311             movimiento = ingresarMovimiento(coordenada);
312             movimientoJugado(coordenada, movimiento, ejercito1, ejercito2, tablero);
313             imprimirTablero(tablero);
314         } else {
315             System.out.println("Turno del Ejercito 2");
316             coordenada = ingresar(ejercito2.get(0).getFigura(), tablero);
317             movimiento = ingresarMovimiento(coordenada);
318             movimientoJugado(coordenada, movimiento, ejercito2, ejercito1, tablero);
319             imprimirTablero(tablero);
320         }
321         System.out.println("Desea seguir jugando?");
322         jugar = sc.next();
323         jugar = jugar.toUpperCase();
324         i++;
325     }
326     if(ejercito1.size() == 0){
327         System.out.println("Salio victorioso el ejercito 2");
328         for(Soldado n: ejercito2){
329             System.out.println(n);
330         }
331     } else if(ejercito2.size() == 0){
332         System.out.println("Salio victorioso el ejercito 1");
333         for(Soldado m: ejercito1){
334             System.out.println(m);
335         }
336     } else {
337         System.out.println("Se cancelo el juego");
338     }
339 }
340 public static void movimientoJugado(String coordenada, int movimiento, ArrayList<Soldado> usuario, ArrayList<Soldado> contrincante, char[][] tablero){
341     int fila_act, columna_act, fila_mov, columna_mov, posAct;
342     fila_act = Integer.parseInt(coordenada.substring(1, coordenada.length())) - 1;
343     columna_act = nroColumna(coordenada.charAt(0));
344     fila_mov = Movimiento.movFila(fila_act, movimiento);
345     columna_mov = Movimiento.movColumna(columna_act, movimiento);
346     if(tablero[fila_mov][columna_mov] == usuario.get(0).getFigura()){
347         System.out.println("Posicion ocupada por una figura de tu mismo ejercito");
348     } else if(tablero[fila_mov][columna_mov] == '-'){
349         posAct = buscadorPosicion(usuario, coordenada);
350         usuario.get(posAct).setFila(fila_mov + 1);
351         usuario.get(posAct).setCol(conversionBusqueda(columna_mov));
352         tablero[fila_act][columna_act] = '-';
353         tablero[fila_mov][columna_mov] = usuario.get(0).getFigura();
354     } else {
355         System.out.println("Enemigo hallado");
356         batalla(coordenada, movimiento, usuario, contrincante, tablero);
357     }
358 }
359 public static void batalla(String coordenada, int movimiento, ArrayList<Soldado> usuario, ArrayList<Soldado> contrincante, char[][] tablero){
360     int fila_act, columna_act, fila_mov, columna_mov, posAct, us, cont, vidaAd;
361     fila_act = Integer.parseInt(coordenada.substring(1, coordenada.length())) - 1;
362     columna_act = nroColumna(coordenada.charAt(0));
363     fila_mov = Movimiento.movFila(fila_act, movimiento);

```

```

364     columna_mov = Movimiento.movColumna(columna_act, movimiento);
365     String movRealizado = conversionBusqueda(columna_mov) + String.valueOf(fila_mov + 1);
366     us = buscadorPosicion(usuario, coordenada);
367     usuario.get(us).atacar();
368     cont = buscadorPosicion(contrincante, movRealizado);
369     contrincante.get(cont).defender();
370     System.out.println("Jugador actual : " + usuario.get(us).getNivVidAct());
371     System.out.println("Jugador enemigo : " + contrincante.get(cont).getNivVidAct());
372     int total = usuario.get(us).getNivVidAct() + contrincante.get(cont).getNivVidAct();
373     double probabilidad1 = usuario.get(us).getNivVidAct() * 100 / total;
374     probabilidad1 = Math.round(probabilidad1 * 10.0) / 10.0;
375     System.out.println("Posibilidad de porcentaje del Jugador actual: " + probabilidad1 + "%");
376     System.out.println("Posibilidad de porcentaje del Jugador enemigo: " + (100 - probabilidad1) + "%");
377     int nroRandom = (int)(Math.random() * 100 + 1);
378     System.out.println("El numero aleatorio fue de " + nroRandom);
379     if(0 <= nroRandom && nroRandom <= probabilidad1){
380         contrincante.remove(cont);
381         usuario.get(us).setFila(fila_mov + 1);
382         usuario.get(us).setCol(conversionBusqueda(columna_mov));
383         tablero[fila_mov][columna_mov] = usuario.get(us).getFigura();
384         tablero[fila_act][columna_act] = '-';
385         usuario.get(us).addVida();
386     } else {
387         usuario.remove(us);
388         tablero[fila_act][columna_act] = '-';
389         contrincante.get(cont).addVida();
390     }
391 }
392 public static String ingresar(char n, char[][] tablero){
393     String coordenada = "";
394     int fila, columna;
395     boolean validez = true;
396     Scanner sc = new Scanner(System.in);
397     while(validez){
398         System.out.println("1 2 3\n8 x 4\n7 6 5");
399         System.out.println("Ingrese la coordenada");
400         coordenada = sc.next();
401         fila = Integer.parseInt(coordenada.substring(1, coordenada.length()));
402         columna = nroColumna(coordenada.charAt(0));
403         System.out.println(fila + " " + columna);
404         if(tablero[fila - 1][columna] == n){
405             break;
406         } else {
407             System.out.println("Coordenada incorrecta");
408         }
409     }
410     return coordenada;
411 }
412 public static int ingresarMovimiento(String coordenada){
413     int movimiento = 0;
414     boolean validez = true;
415     int fila, columna;
416     Scanner sc = new Scanner(System.in);
417     while(validez){
418         System.out.println("Ingresar Movimiento: ");
419         movimiento = sc.nextInt();
420         fila = Integer.parseInt(coordenada.substring(1, coordenada.length())) - 1;
421         columna = nroColumna(coordenada.charAt(0));
422         fila = Movimiento.movFila(fila, movimiento);
423         columna = Movimiento.movColumna(columna, movimiento);
424         System.out.println(fila + " Movimiento aplicado " + columna);
425         if(validezMovimiento(fila, columna)){
426             break;
427         } else {
428             System.out.println("Movimiento invalido");
429             fila = Movimiento.restFila(fila, movimiento);
430             columna = Movimiento.restColumna(columna, movimiento);
431         }
432     }
433     return movimiento;
434 }
435 public static boolean validezMovimiento(int fila, int columna){
436     return fila <= 9 && fila >= 0 && columna <= 9 && columna >= 0;
437 }
438 public static void imprimirDatosEjercito(ArrayList<Soldado> ejercito){
439     for(int i = 0; i < ejercito.size(); i++){
440         System.out.println((i + 1) + " " + ejercito.get(i));
441     }
442 }
443 public static void datosEjercito(int n, ArrayList<Soldado> ejercito, char fig){
444     int nroSoldados = (int)(Math.random() * 10 + 1);
445     System.out.println("El ejercito " + n + " tiene un total de " + nroSoldados + " soldados");
446     for(int i = 0; i < nroSoldados; i++){
447         String nombre = "Soldado " + (i + 1) + "X" + n;
448         ejercito.add(new Soldado(nombre, fig));
449     }
450 }
451 public static void impTablero(char[][] tablero){
452     System.out.println("El tablero es :");
453     for(int x = 0; x < tablero.length; x++){
454         for(int y = 0; y < tablero[x].length; y++){
455             System.out.print(tablero[x][y]);
456         }
457         System.out.println();
458     }
459 }

```

	<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 style="text-align: center;">Aprobación: 2022/03/01</p>	<p style="text-align: center;">Código: GUIA-PRLE-001</p>	<p style="text-align: right;">Página: 11</p>

```

460 public static void actTablero(ArrayList<Soldado> ejercito, char[][] tablero){
461     int fila;
462     char columna;
463     for(int i = 0; i < ejercito.size(); i++){
464         fila = ejercito.get(i).getFila();
465         columna = ejercito.get(i).getColumna();
466         while(tablero[fila - 1][nroColumna(columna)] != '-'){
467             fila = (int)(Math.random() * 10 + 1);
468             columna = Soldado.numCol();
469         }
470         ejercito.get(i).setFila(fila);
471         ejercito.get(i).setCol(columna);
472         tablero[fila - 1][nroColumna(columna)] = ejercito.get(i).getFigura();
473     }
474 }
475 public static int nroColumna(char n){
476     switch(n){
477         case 'A': return 0;
478         case 'B': return 1;
479         case 'C': return 2;
480         case 'D': return 3;
481         case 'E': return 4;
482         case 'F': return 5;
483         case 'G': return 6;
484         case 'H': return 7;
485         case 'I': return 8;
486         case 'J': return 9;
487         default: return 0;
488     }
489 }
490 public static char conversionBusqueda(int m){
491     switch(m){
492         case 0: return 'A';
493         case 1: return 'B';
494         case 2: return 'C';
495         case 3: return 'D';
496         case 4: return 'E';
497         case 5: return 'F';
498         case 6: return 'G';
499         case 7: return 'H';
500         case 8: return 'I';
501         case 9: return 'J';
502         default: return 0;
503     }
504 }
505 public static int buscadorPosicion(ArrayList<Soldado> ejercito, String posicion){
506     for(int i = 0; i < ejercito.size(); i++){
507         if(posicion.charAt(0) == ejercito.get(i).getColumna() && Integer.parseInt(posicion.substring(1, posicion.length())) == ejercito.get(i).getFila()){
508             return i;
509         }
510     }
511     return 0;
512 }
513 }

```

II. PRUEB

	<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: 12</p>

```

VideoJuego (1) [Java Application] C:\Users\UJKMjuandi\Downloads\eclipse-jav
El ejercito 1 tiene un total de 8 soldados
El ejercito 2 tiene un total de 1 soldados
El tablero es :
*-----
-----
*-----
*-----
*-----
*-----
*-----
&-----
*-----
*-----
Ingrese una opcion
Opcion 1: Juego Rapido
Opcion 2: Juego Personalizado
Opcion 3: Salir
2
Se inicio una partida personalizada
1
¿Que ejercito deseas aplicar modificacion?
Opcion 1 : Ejercito *
Opcion 2: Ejercico $
Opcion 1: Ingresar Soldado
Opcion 2: Eliminar Soldado
Opcion 3: Clonar Soldado
Opcion 4: Modificar Soldado
Opcion 5: Comparar Soldados
Opcion 6: Intercambiar Soldados
Opcion 7: Ver Soldado
Opcion 8: Ver ejercito
Opcion 9: Suma de Niveles
Opcion 10: Iniciar juego
Opcion 11: Salir al menu principal
Ingresa la opcion que desea realizar:

```

III. CUESTIONARIO:

	<p align="center">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p 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 align="right">Página: 13</p>

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

CONCLUSIONES

En cuanto a la importancia de la creación de un menú para el juego, se ha implementado un sistema intuitivo que permite a los jugadores acceder fácilmente a diferentes opciones, como la selección de modos de juego, la gestión de perfiles y la personalización de configuraciones. Este menú proporciona una experiencia de usuario cohesionada y eficiente, mejorando la accesibilidad y la inmersión en el juego.

Adicionalmente, se han incorporado métodos para manejar eventos y resultados durante el enfrentamiento humano vs humano, asegurando una evaluación precisa de las acciones, cálculos precisos de resultados y una actualización coherente del estado del juego. Este enfoque contribuye a una experiencia de juego sin problemas y libre de errores, consolidando la calidad del enfrentamiento entre jugadores humanos.


Considerando las complejidades estratégicas, la implementación de métodos para evaluar fortalezas y debilidades de los soldados, así como la planificación y ejecución de estrategias tácticas, brinda a los jugadores la oportunidad de participar en una experiencia de juego más rica, desafiante y plena.

METODOLOGÍA DE TRABAJO

- 1.- Leer los enunciados cuidadosamente para saber que requiere el problemas.*
- 2.- Seguir las indicaciones para la implementación del menu*
- 3.-programar lo que se mostrara en la consola*
- 4.- Realizar el Videojuego con los métodos apropiados.*

REFERENCIAS Y BIBLIOGRAFÍA

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

	<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

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		Pun- tos	Che- cklist	Estu- dian- te	Profe- sor
1. GitHub	Hay enlace URL activo del directorio para el laboratorio hacia su repositorio GitHub con código fuente terminado y fácil de revisar.	2	X	2	
2. Commits	Hay capturas de pantalla de los commits más importantes con sus explicaciones detalladas. (El profesor puede preguntar para refrendar calificación).	4	X	3	
3. Código fuente	Hay porciones de código fuente importantes con numeración y explicaciones detalladas de sus funciones.	2	X	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 establecidos.	2	X	0	
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	4	X	3	

	<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: 15</p>

	impecable. (El profesor puede preguntar para refrendar calificación).				
TOTAL		20		14	

Tabla 2: Rúbrica para contenido del Informe y demostración