

Informe de Laboratorio 12

Tema: Clase Soldado - Menú

Nota

Estudiante	Escuela	Asignatura
Christian Mestas Zegarra cmestasz@unsa.edu.pe	Escuela Profesional de Ingeniería de Sistemas	Fundamentos de la Programación 2 Semestre: II Código: 1701213

Laboratorio	Tema	Duración
12	Clase Soldado - Menú	04 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2023 - B	Del 06 Diciembre 2023	Al 11 Diciembre 2023

1. Tarea

- **Item 1:** Puede reutilizar todo el código del laboratorio 11, pero ahora el objetivo es gestionar los ejércitos autogenerados.
- **Item 2:** Al ejecutar el videojuego, el programa deberá dar las opciones:
 1. Juego rápido (tal cual como en el laboratorio 11) Al acabar el juego mostrar las opciones de volver a jugar y de volver al menú principal. También se deberá tener la posibilidad de cancelar el juego actual en cualquier momento, permitiendo escoger entre empezar un juego totalmente nuevo o salir al menú principal.
 2. Juego personalizado: permite gestionar ejércitos. Primero se generan los 2 ejércitos con sus respectivos soldados y se muestran sus datos. Luego se tendrá que escoger cuál de los 2 ejércitos se va a gestionar, después se mostrarán las siguientes opciones:
 - 1) Crear Soldado: permitirá crear un nuevo soldado personalizado y añadir al final del ejército (recordar que límite es de 10 soldados por ejército)
 - 2) Eliminar Soldado (no debe permitir un ejército vacío)
 - 3) Clonar Soldado (crea una copia exacta del soldado) y se añade al final del ejército (recordar que límite es de 10 soldados por ejército)
 - 4) Modificar Soldado (con submenú para cambiar alguno de los atributos nivelAtaque, nivelDefensa, vidaActual)
 - 5) Comparar Soldados (verifica si atributos: nombre, nivelAtaque, nivelDefensa, vidaActual y vive son iguales)
 - 6) Intercambiar Soldados (intercambia 2 soldados en sus posiciones en la estructura de datos del ejército)
 - 7) Ver soldado (Búsqueda por nombre)

- 8) Ver ejército
 - 9) Sumar niveles (usando Method-Call Chaining), calcular las sumatorias de nivelVida, nivelAtaque, nivelDefensa, velocidad de todos los soldados de un ejército 1. Por ejemplo, si ejército tendría 3 soldados: 2. `s=s1.sumar(s2).sumar(s3)`; 3. `s` es un objeto Soldado nuevo que contendría las sumatorias de los 4 atributos indicados de los 3 soldados. Ningún soldado cambia sus valores
 - 10) Jugar (se empezará el juego con los cambios realizados) y con las mismas opciones de la opción 1 del menú principal.
 - 11) Volver (muestra el menú principal) Después de escoger alguna de las opciones 1) a 9) se podrá volver a elegir uno de los ejércitos y se mostrarán las opciones 1) a 11)
3. Salir

2. Equipos, materiales y temas utilizados

- Sistema Operativo Microsoft Windows 10 Pro 64 bits
- Visual Studio Code 1.82.2
- Java Development Kit 17.0.1
- Git 2.41.0.windows.1
- Windows PowerShell 5.1.19041.3031
- Cuenta en GitHub con el correo institucional.
- Programación Orientada a Objetos
- HashMap de Objetos
- ArrayList de Objetos
- Ordenamientos burbuja y por selección.

3. URL de Repositorio Github

- URL del Repositorio GitHub para clonar o recuperar.
- <https://github.com/cmestasz/fp2-23b.git>
- URL para el laboratorio 12 en el Repositorio GitHub.
- <https://github.com/cmestasz/fp2-23b/tree/main/fase02/lab12>

4. Actividades con el repositorio GitHub

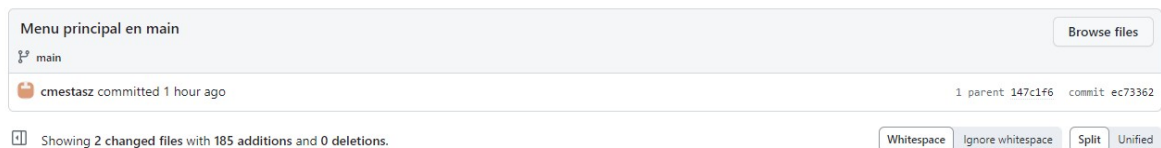
commits.bash

```
1 # CREACION DE CARPETAS Y ARCHIVOS COMO PLANTILLA
2 $ mkdir lab07
3 $ code lab08/Soldado.java
4 $ code lab08/VideoJuego5.java
5 $ git add .
6 # MENU PRINCIPAL Y LOS METODOS A USAR, CLASE SOLDADO
7 $ git commit -m "Menu principal en main"
8 [main ec73362] Menu principal en main
9 2 files changed, 185 insertions(+)
10 create mode 100644 fase02/lab12/Soldado.java
11 create mode 100644 fase02/lab12/VideoJuego9.java
12 $ git add .
13 # JUEGO RAPIDO, YA DESARROLLADO EN EL LABORATORIO ANTERIOR
14 $ git commit -m "Juego rapido (Laboratorio anterior)"
15 [main 7ce4e90] Juego rapido (Laboratorio anterior)
16 1 file changed, 296 insertions(+)
17 $ git add .
18 # MENU DE JUEGO PERSONALIZADO Y LOS METODOS A USAR
19 $ git commit -m "Base del menu del Juego Personalizado"
20 [main 922afbe] Base del menu del Juego Personalizado
21 1 file changed, 86 insertions(+)
22 $ git add .
23 # METODO QUE PERMITE CREAR UN SOLDADO
24 $ git commit -m "Metodo crearSoldado"
25 [main 7aa0571] Metodo crearSoldado
26 1 file changed, 34 insertions(+), 8 deletions(-)
27 $ git add .
28 # METODO QUE PERMITE ELIMINAR UN SOLDADO
29 $ git commit -m "Metodo eliminarSoldado"
30 [main 360384d] Metodo eliminarSoldado
31 1 file changed, 18 insertions(+)
32 $ git add .
33 # METODO QUE PERMITE COPIAR UN SOLDADO A OTRA POSICION
34 $ git commit -m "Metodo clonarSoldado"
35 [main 52db412] Metodo clonarSoldado
36 1 file changed, 26 insertions(+)
37 $ git add .
38 # METODO QUE PERMITE MODIFICAR LOS ATRIBUTOS DE UN SOLDADO
39 $ git commit -m "Metodo modificarSoldado"
40 [main fcbcc3d] Metodo modificarSoldado
41 1 file changed, 32 insertions(+)
42 $ git add .
43 # METODO QUE PERMITE COMPARAR LOS ATRIBUTOS DE DOS SOLDADOS
44 $ git commit -m "Metodo compararSoldados"
45 [main 08fe8fa] Metodo compararSoldados
46 1 file changed, 31 insertions(+)
47 $ git add .
48 #METODO QUE PERMITE INTERCAMBIAR LA POSICION DE DOS SOLDADOS
49 $ git commit -m "Metodo intercambiarSoldados"
50 [main b5f210a] Metodo intercambiarSoldados
51 1 file changed, 31 insertions(+)
52 $ git add .
```

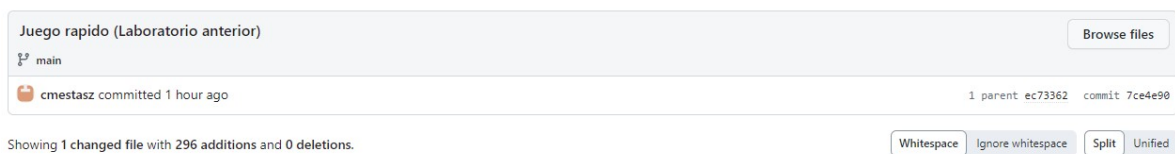
```

53 # METODO QUE MUESTRA UN SOLDADO POR NOMBRE
54 $ git commit -m "Metodo verSoldado"
55 [main a386c0d] Metodo verSoldado
56 1 file changed, 9 insertions(+)
57 # PUSH SECUNDARIO
58 $ git push
59 Enumerating objects: 60, done.
60 Counting objects: 100% (60/60), done.
61 Delta compression using up to 4 threads
62 Compressing objects: 100% (56/56), done.
63 Writing objects: 100% (56/56), 8.76 KiB | 1.75 MiB/s, done.
64 Total 56 (delta 32), reused 0 (delta 0), pack-reused 0
65 remote: Resolving deltas: 100% (32/32), completed with 3 local objects.
66 To https://github.com/cmestasz/fp2-23b.git
67 ce3d609..a386c0d main -> main
68 $ git add .
69 # METODO QUE MUESTRA LA SUMA DE ATRIBUTOS DEL EJERCITO
70 $ git commit -m "Metodo sumarNiveles"
71 [main 593e33d] Metodo sumarNiveles
72 1 file changed, 8 insertions(+)
73 # PUSH FINAL
74 $ git push
75 Enumerating objects: 9, done.
76 Counting objects: 100% (9/9), done.
77 Delta compression using up to 4 threads
78 Compressing objects: 100% (5/5), done.
79 Writing objects: 100% (5/5), 589 bytes | 589.00 KiB/s, done.
80 Total 5 (delta 3), reused 0 (delta 0), pack-reused 0
81 remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
82 To https://github.com/cmestasz/fp2-23b.git
83 a386c0d..593e33d main -> main

```



Primer Commit.



Segundo Commit.

Base del menu del Juego Personalizado
main
cmestasz committed 49 minutes ago
1 parent 7ce4e90 commit 922afbe

[Browse files](#)

Showing 1 changed file with 86 additions and 0 deletions.

[Whitespace](#) [Ignore whitespace](#) [Split](#) [Unified](#)

Tercer Commit.

Metodo crearSoldado
main
cmestasz committed 33 minutes ago
1 parent 922afbe commit 7aa0571

[Browse files](#)

Showing 1 changed file with 34 additions and 8 deletions.

[Whitespace](#) [Ignore whitespace](#) [Split](#) [Unified](#)

Cuarto Commit.

Metodo eliminarSoldado
main
cmestasz committed 29 minutes ago
1 parent 7aa0571 commit 360384d

[Browse files](#)

Showing 1 changed file with 18 additions and 0 deletions.

[Whitespace](#) [Ignore whitespace](#) [Split](#) [Unified](#)

Quinto Commit.

Metodo clonarSoldado
main
cmestasz committed 23 minutes ago
1 parent 360384d commit 52db412

[Browse files](#)

Showing 1 changed file with 26 additions and 0 deletions.

[Whitespace](#) [Ignore whitespace](#) [Split](#) [Unified](#)

Sexto Commit.

Metodo modificarSoldado
main
cmestasz committed 17 minutes ago
1 parent 52db412 commit fcbcc3d

[Browse files](#)

Showing 1 changed file with 32 additions and 0 deletions.

[Whitespace](#) [Ignore whitespace](#) [Split](#) [Unified](#)

Septimo Commit.

Metodo compararSoldados
main
cmestasz committed 12 minutes ago
1 parent fcbcc3d commit 08fe8fa


[Browse files](#)


Showing 1 changed file with 31 additions and 0 deletions.

[Whitespace](#) [Ignore whitespace](#) [Split](#) [Unified](#)

Octavo Commit.

Metodo intercambiarSoldados [Browse files](#)


 main


 **cmestasz** committed 6 minutes ago 1 parent 08fe0fa commit b5f210a

Showing 1 changed file with 31 additions and 0 deletions. [Whitespace](#) [Ignore whitespace](#) [Split](#) [Unified](#)

Noveno Commit.

Metodo verSoldado [Browse files](#)


 main


 **cmestasz** committed 3 minutes ago 1 parent b5f210a commit a386c0d

Showing 1 changed file with 9 additions and 0 deletions. [Whitespace](#) [Ignore whitespace](#) [Split](#) [Unified](#)

Decimo Commit.

Metodo sumarNiveles [Browse files](#)

 main

 **cmestasz** committed 2 minutes ago 1 parent a386c0d commit 593e33d

Showing 1 changed file with 8 additions and 0 deletions. [Whitespace](#) [Ignore whitespace](#) [Split](#) [Unified](#)

Decimo Primer Commit.

5. Código desarrollado

Soldado.java

```
1 package fase02.lab12;
2
3 public class Soldado {
4     private String nombre;
5     private int equipo;
6     private int ataque;
7     private int defensa;
8     private int vidaInicial;
9     private int vidaActual;
10    private int velocidad;
11    private String actitud;
12    private boolean vive;
13
14    public Soldado(String nombreN, int equipoN, int ataqueN, int defensaN, int vidaInicialN) {
15        nombre = nombreN;
16        equipo = equipoN;
17        ataque = ataqueN;
18        defensa = defensaN;
19        vidaInicial = vidaInicialN;
20        vidaActual = vidaInicial;
21        actitud = "defensiva";
22        vive = true;
23    }
24
25    public Soldado(String nombreN, int equipoN, int ataqueN, int defensaN) {
26        this(nombreN, equipoN, ataqueN, defensaN, 5);
27    }
28
29    public Soldado(String nombreN, int equipoN, int vidaInicialN) {
30        this(nombreN, equipoN, 5, 5, vidaInicialN);
31    }
32
33    public Soldado() {
34
35    }
36
37    public void setVidaActual(int vidaActualN) {
38        vidaActual = vidaActualN;
39    }
40
41    public void setAtaque(int ataque) {
42        this.ataque = ataque;
43    }
44
45    public void setDefensa(int defensa) {
46        this.defensa = defensa;
47    }
48
49    public void setVelocidad(int velocidad) {
50        this.velocidad = velocidad;
51    }
52
```



```
53     public int getVidaActual() {
54         return vidaActual;
55     }
56
57     public String getNombre() {
58         return nombre;
59     }
60
61     public int getEquipo() {
62         return equipo;
63     }
64
65     public int getAtaque() {
66         return ataque;
67     }
68
69     public int getDefensa() {
70         return defensa;
71     }
72
73     public int getVelocidad() {
74         return velocidad;
75     }
76
77     public Soldado getCopia() {
78         return new Soldado(nombre, equipo, ataque, defensa, vidaInicial);
79     }
80
81     public boolean vive() {
82         return vive;
83     }
84
85     public void atacar() {
86         actitud = "ofensiva";
87         avanzar();
88     }
89
90     public void defender() {
91         actitud = "defensiva";
92         parar();
93     }
94
95     public void huir() {
96         actitud = "fuga";
97         velocidad = 2;
98     }
99
100    public void avanzar() {
101        velocidad = 1;
102    }
103
104    public void parar() {
105        velocidad = 0;
106    }
107
108    public void retroceder() {
```

```
109     if (velocidad > 0)
110         defender();
111     else
112         velocidad = -1;
113 }
114
115 public void serAtacado(int d) {
116     vidaActual -= d;
117     if (vidaActual <= 0)
118         morir();
119 }
120
121 public void morir() {
122     vive = false;
123 }
124
125 public void mejorar() {
126     vidaActual++;
127 }
128
129 public Soldado sumar(Soldado otro) {
130     Soldado nuevo = new Soldado();
131     nuevo.setVidaActual(vidaActual + otro.getVidaActual());
132     nuevo.setAtaque(ataque + otro.getAtaque());
133     nuevo.setDefensa(defensa + otro.getDefensa());
134     nuevo.setVelocidad(velocidad + otro.getVelocidad());
135     return nuevo;
136 }
137
138 public String toString() {
139     return nombre + ": Vida: " + vidaActual + ". At: " + ataque + ". Df: " + defensa + ".
140         Vl: " + velocidad;
141 }
```

- Clase que guarda nombre y vida del soldado.
- Posee tanto setters como getters para todos los atributos.
- Permite crear una copia con el metodo getCopia().
- Permite crear un nuevo soldado con la suma de los atributos de otros soldados con el metodo sumar().
- Posee el metodo toString() para poder imprimir el objeto.

VideoJuego5.java

```
1 package fase02.lab12;
2
3 import java.util.Scanner;
4 import java.util.ArrayList;
5 import java.util.HashMap;
6 import java.util.Random;
7
8 public class VideoJuego9 {
9     private static final int ESCALA = 10;
10    private static Random random;
11    private static Scanner sc = new Scanner(System.in);
12
13    public static void main(String[] args) {
14        int opcion = 3;
15        do {
16            System.out.println("1. Juego Rapido\n2. Juego Personalizado\n3. Salir");
17            System.out.print("Opcion: ");
18            opcion = sc.nextInt();
19            sc.nextLine();
20            System.out.println();
21            String continuar = "N";
22            do {
23                switch (opcion) {
24                    case 1:
25                        juegoRapido();
26                        break;
27                    case 2:
28                        juegoPersonalizado();
29                        break;
30                }
31                if (opcion != 3) {
32                    System.out.print("\nVolver a jugar? (S/N): ");
33                    continuar = sc.nextLine();
34                }
35                } while (continuar.equalsIgnoreCase("S"));
36        } while (opcion != 3);
37    }
38
39    private static void juegoPersonalizado() {
40        int semilla = new Random().nextInt(1000000);
41        random = new Random(semilla);
42        System.out.println(semilla);
43        HashMap<String, Soldado> mapaSoldados = new HashMap<String, Soldado>();
44        ArrayList<Soldado> listaSoldados1 = new ArrayList<Soldado>();
45        ArrayList<Soldado> listaSoldados2 = new ArrayList<Soldado>();
46        inicializarSoldados(mapaSoldados, listaSoldados1, 1);
47        inicializarSoldados(mapaSoldados, listaSoldados2, 2);
48        int opcion;
49        do {
50            imprimirTablero(mapaSoldados);
51            System.out.println("\nEjercito 1:");
52            imprimirSoldados(listaSoldados1);
53            System.out.println("\nEjercito 2:");
54            imprimirSoldados(listaSoldados2);
55            System.out.print("\nEjercito a gestionar (1/2): ");
```

```
56         int equipo = sc.nextInt();
57         ArrayList<Soldado> seleccionado = equipo == 1 ? listaSoldados1 : listaSoldados2;
58         System.out.println(
59             "1. Crear soldado\n2. Eliminar soldado\n3. Clonar soldado\n4. Modificar\n5. Comparar\n6. Intercambiar\n7. Ver\n8. Ver ejercito\n9. Sumar niveles\n10. Jugar\n11. Volver");
60         System.out.print("Opcion: ");
61         opcion = sc.nextInt();
62         sc.nextLine();
63         System.out.println();
64         switch (opcion) {
65             case 1:
66                 crearSoldado(mapaSoldados, seleccionado, equipo);
67                 break;
68             case 2:
69                 eliminarSoldado(mapaSoldados, seleccionado, equipo);
70                 break;
71             case 3:
72                 clonarSoldado(mapaSoldados, seleccionado);
73                 break;
74             case 4:
75                 modificarSoldado(mapaSoldados, seleccionado);
76                 break;
77             case 5:
78                 compararSoldados(mapaSoldados, seleccionado);
79                 break;
80             case 6:
81                 intercambiarSoldados(mapaSoldados, seleccionado);
82                 break;
83             case 7:
84                 verSoldado(seleccionado);
85                 break;
86             case 8:
87                 imprimirSoldados(seleccionado);
88                 break;
89             case 9:
90                 sumarNiveles(seleccionado);
91                 break;
92             case 10:
93                 realizarCombates(mapaSoldados, listaSoldados1, listaSoldados2);
94                 break;
95             case 11:
96                 return;
97         }
98         System.out.println();
99     } while (opcion < 10);
100 }

101
102 private static void crearSoldado(HashMap<String, Soldado> mapaSoldados,
103     ArrayList<Soldado> listaSoldados,
104     int equipo) {
105     if (listaSoldados.size() < 10) {
106         System.out.print("Nombre: ");
107         String nombre = sc.nextLine();
108         System.out.print("Vida: ");
109         int vida = sc.nextInt();
```

```
109     System.out.print("Ataque: ");
110     int ataque = sc.nextInt();
111     System.out.print("Defensa: ");
112     int defensa = sc.nextInt();
113     sc.nextLine();
114     int filaI, columnaI;
115     do {
116         System.out.print("Fila: ");
117         String fila = sc.nextLine();
118         filaI = Integer.parseInt(fila) - 1;
119         System.out.print("Columna: ");
120         String columna = sc.nextLine();
121         columnaI = charToInt(columna.charAt(0)) - 1;
122     } while (!coordenadaValida(mapaSoldados, filaI, columnaI)
123             || seleccionOcupada(mapaSoldados, filaI, columnaI));
124     Soldado soldado = new Soldado(nombre, equipo, ataque, defensa, vida);
125     mapaSoldados.put(generarLlave(filaI, columnaI), soldado);
126     listaSoldados.add(soldado);
127 } else {
128     System.out.println("Ejercito lleno!");
129 }
130 }
131
132 private static void eliminarSoldado(HashMap<String, Soldado> mapaSoldados,
133     ArrayList<Soldado> listaSoldados,
134     int equipo) {
135     if (listaSoldados.size() > 1) {
136         int filaI, columnaI;
137         do {
138             System.out.print("Fila: ");
139             String fila = sc.nextLine();
140             filaI = Integer.parseInt(fila) - 1;
141             System.out.print("Columna: ");
142             String columna = sc.nextLine();
143             columnaI = charToInt(columna.charAt(0)) - 1;
144         } while (!coordenadaValida(mapaSoldados, filaI, columnaI) ||
145                 !seleccionValida(mapaSoldados, filaI, columnaI, equipo));
146         String llave = generarLlave(filaI, columnaI);
147         Soldado soldado = mapaSoldados.get(llave);
148         listaSoldados.remove(soldado);
149         mapaSoldados.remove(llave);
150     } else {
151         System.out.println("No se puede tener un ejercito vacio!");
152     }
153 }
154
155 private static void clonarSoldado(HashMap<String, Soldado> mapaSoldados,
156     ArrayList<Soldado> listaSoldados) {
157     if (listaSoldados.size() < 10) {
158         int filaI, columnaI, filaF, columnaF;
159         do {
160             System.out.print("Fila de origen: ");
161             String fila = sc.nextLine();
162             filaI = Integer.parseInt(fila) - 1;
```

```
163         columnaI = charToInt(columna.charAt(0)) - 1;
164     } while (!coordenadaValida(mapaSoldados, filaI, columnaI)
165             || !seleccionLlena(mapaSoldados, filaI, columnaI));
166     do {
167         System.out.print("Fila de destino: ");
168         String fila = sc.nextLine();
169         filaF = Integer.parseInt(fila) - 1;
170         System.out.print("Columna de destino: ");
171         String columna = sc.nextLine();
172         columnaF = charToInt(columna.charAt(0)) - 1;
173     } while (!coordenadaValida(mapaSoldados, filaF, columnaF)
174             || seleccionOcupada(mapaSoldados, filaF, columnaF));
175     Soldado soldado = mapaSoldados.get(generarLlave(filaI, columnaI)).getCopia();
176     mapaSoldados.put(generarLlave(filaF, columnaF), soldado);
177     listaSoldados.add(soldado);
178 } else {
179     System.out.println("Ejercito lleno!");
180 }
181 }
182
183 private static void modificarSoldado(HashMap<String, Soldado> mapaSoldados,
184     ArrayList<Soldado> listaSoldados) {
185     int filaI, columnaI;
186     do {
187         System.out.print("Fila: ");
188         String fila = sc.nextLine();
189         filaI = Integer.parseInt(fila) - 1;
190         System.out.print("Columna: ");
191         String columna = sc.nextLine();
192         columnaI = charToInt(columna.charAt(0)) - 1;
193     } while (!coordenadaValida(mapaSoldados, filaI, columnaI)
194             || !seleccionLlena(mapaSoldados, filaI, columnaI));
195     Soldado soldado = mapaSoldados.get(generarLlave(filaI, columnaI));
196     if (listaSoldados.contains(soldado)) {
197         System.out.println("1. Ataque\n2. Defensa\n3. Vida");
198         System.out.print("Opcion: ");
199         int opcion = sc.nextInt();
200         System.out.print("Nuevo valor: ");
201         int valor = sc.nextInt();
202         sc.nextLine();
203         switch (opcion) {
204             case 1:
205                 soldado.setAtaque(valor);
206                 break;
207             case 2:
208                 soldado.setDefensa(valor);
209                 break;
210             case 3:
211                 soldado.setVidaActual(valor);
212                 break;
213         }
214     } else {
215         System.out.println("El soldado no pertenece a tu ejercito!");
216     }
217 }
```

```
218 private static void compararSoldados(HashMap<String, Soldado> mapaSoldados,
219     ArrayList<Soldado> listaSoldados) {
220     int filaI, columnaI, filaF, columnaF;
221     do {
222         System.out.print("Fila soldado 1: ");
223         String fila = sc.nextLine();
224         filaI = Integer.parseInt(fila) - 1;
225         System.out.print("Columna: ");
226         String columna = sc.nextLine();
227         columnaI = charToInt(columna.charAt(0)) - 1;
228     } while (!coordenadaValida(mapaSoldados, filaI, columnaI)
229         || !seleccionLlena(mapaSoldados, filaI, columnaI));
230     do {
231         System.out.print("Fila soldado 2: ");
232         String fila = sc.nextLine();
233         filaF = Integer.parseInt(fila) - 1;
234         System.out.print("Columna: ");
235         String columna = sc.nextLine();
236         columnaF = charToInt(columna.charAt(0)) - 1;
237     } while (!coordenadaValida(mapaSoldados, filaF, columnaF)
238         || !seleccionLlena(mapaSoldados, filaF, columnaF));
239     Soldado soldado1 = mapaSoldados.get(generarLlave(filaI, columnaI));
240     Soldado soldado2 = mapaSoldados.get(generarLlave(filaF, columnaF));
241     if (soldado1.getNombre().equals(soldado2.getNombre()))
242         System.out.println("Nombres iguales");
243     if (soldado1.getAtaque() == soldado2.getAtaque())
244         System.out.println("Ataques iguales");
245     if (soldado1.getDefensa() == soldado2.getDefensa())
246         System.out.println("Defensas iguales");
247     if (soldado1.getVidaActual() == soldado2.getVidaActual())
248         System.out.println("Vidas iguales");
249     if (soldado1.vive() == soldado2.vive())
250         System.out.println("Estados vitales iguales");
251 }
252
253 private static void intercambiarSoldados(HashMap<String, Soldado> mapaSoldados,
254     ArrayList<Soldado> listaSoldados) {
255     int filaI, columnaI, filaF, columnaF;
256     do {
257         System.out.print("Fila soldado 1: ");
258         String fila = sc.nextLine();
259         filaI = Integer.parseInt(fila) - 1;
260         System.out.print("Columna soldado 1: ");
261         String columna = sc.nextLine();
262         columnaI = charToInt(columna.charAt(0)) - 1;
263     } while (!coordenadaValida(mapaSoldados, filaI, columnaI)
264         || !seleccionLlena(mapaSoldados, filaI, columnaI));
265     do {
266         System.out.print("Fila soldado 2: ");
267         String fila = sc.nextLine();
268         filaF = Integer.parseInt(fila) - 1;
269         System.out.print("Columna soldado 2: ");
270         String columna = sc.nextLine();
271         columnaF = charToInt(columna.charAt(0)) - 1;
272     } while (!coordenadaValida(mapaSoldados, filaF, columnaF)
273         || !seleccionLlena(mapaSoldados, filaF, columnaF));
```

```
272     String llaveI = generarLlave(filaI, columnaI);
273     String llaveF = generarLlave(filaF, columnaF);
274     Soldado soldado1 = mapaSoldados.get(llaveI);
275     Soldado soldado2 = mapaSoldados.get(llaveF);
276     if (listaSoldados.contains(soldado1) && listaSoldados.contains(soldado2)) {
277         mapaSoldados.remove(llaveI);
278         mapaSoldados.remove(llaveF);
279         mapaSoldados.put(llaveI, soldado2);
280         mapaSoldados.put(llaveF, soldado1);
281     } else {
282         System.out.println("Uno o ambos soldados no pertenecen a tu ejercito!");
283     }
284 }
285
286 private static void verSoldado(ArrayList<Soldado> listaSoldados) {
287     System.out.print("Nombre: ");
288     String nombre = sc.nextLine();
289     for (Soldado soldado : listaSoldados) {
290         if (soldado.getNombre().equals(nombre)) {
291             System.out.println(soldado);
292             return;
293         }
294     }
295     System.out.println("Soldado no encontrado!");
296 }
297
298 private static void sumarNiveles(ArrayList<Soldado> listaSoldados) {
299     Soldado suma = new Soldado();
300     for (Soldado soldado : listaSoldados) {
301         suma = suma.sumar(soldado);
302     }
303     System.out.printf("Vida total: %d\n", suma.getVidaActual());
304     System.out.printf("Ataque total: %d\n", suma.getAtaque());
305     System.out.printf("Defensa total: %d\n", suma.getDefensa());
306     System.out.printf("Velocidad total: %d\n", suma.getVelocidad());
307 }
308
309 private static void juegoRapido() {
310     int semilla = new Random().nextInt(1000000);
311     random = new Random(semilla);
312     System.out.println(semilla);
313     HashMap<String, Soldado> mapaSoldados = new HashMap<String, Soldado>();
314     ArrayList<Soldado> listaSoldados1 = new ArrayList<Soldado>();
315     ArrayList<Soldado> listaSoldados2 = new ArrayList<Soldado>();
316     System.out.println("Inicia la batalla!");
317     inicializarSoldados(mapaSoldados, listaSoldados1, 1);
318     inicializarSoldados(mapaSoldados, listaSoldados2, 2);
319     imprimirTablero(mapaSoldados);
320     System.out.printf("Soldado con mayor vida del ejercito 1: %s\n",
321         soldadoMayorVida(listaSoldados1));
321     System.out.printf("Promedio de puntos de vida del ejercito 1: %f\n",
322         promedioPuntosVida(listaSoldados1));
322     System.out.println("\nSoldados por orden de creacion:");
323     imprimirSoldados(listaSoldados1);
324     System.out.println("\nSoldados ordenados por burbuja");
325     ordenarSoldadosBurbuja(listaSoldados1);
```



```
326     imprimirSoldados(listaSoldados1);
327     System.out.println();
328     System.out.printf("Soldado con mayor vida del ejercito 2: %s%n",
329                       soldadoMayorVida(listaSoldados2));
329     System.out.printf("Promedio de puntos de vida del ejercito 2: %f%n",
330                       promedioPuntosVida(listaSoldados2));
330     System.out.println("\nSoldados por orden de creacion:");
331     imprimirSoldados(listaSoldados2);
332     System.out.println("\nSoldados ordenados por seleccion");
333     ordenarSoldadosSeleccion(listaSoldados2);
334     imprimirSoldados(listaSoldados2);
335     System.out.println();
336     realizarCombates(mapaSoldados, listaSoldados1, listaSoldados2);
337 }
338
339 public static void inicializarSoldados(HashMap<String, Soldado> mapaSoldados,
340                                       ArrayList<Soldado> listaSoldados,
341                                       int equipo) {
342     int cantidad = random.nextInt(10) + 1;
343     for (int i = 0; i < cantidad; i++) {
344         String nombre = "Soldado" + i + "X" + equipo;
345         int vida = random.nextInt(5) + 1;
346         int ataque = random.nextInt(5) + 1;
347         int defensa = random.nextInt(5) + 1;
348         int fila, columna;
349         do {
350             fila = random.nextInt(ESCALA);
351             columna = random.nextInt(ESCALA);
352         } while (mapaSoldados.containsKey(generarLlave(fila, columna)));
353         Soldado soldado = new Soldado(nombre, equipo, ataque, defensa, vida);
354         mapaSoldados.put(generarLlave(fila, columna), soldado);
355         listaSoldados.add(soldado);
356     }
357 }
358
359 public static void imprimirTablero(HashMap<String, Soldado> mapaSoldados) {
360     System.out.print(generarEncabezado(mapaSoldados));
361     String separacion = generarSeparacion(mapaSoldados);
362     for (int i = 0; i < ESCALA; i++) {
363         System.out.print(separacion);
364         System.out.print(generarFila(mapaSoldados, i));
365     }
366     System.out.print(separacion);
367 }
368
369 public static String generarEncabezado(HashMap<String, Soldado> mapaSoldados) {
370     String encabezado = "\t";
371     for (int i = 0; i < ESCALA; i++)
372         encabezado += (" " + intToChar(i + 1) + " ");
373     encabezado += "\n";
374     return encabezado;
375 }
376
377 public static String generarSeparacion(HashMap<String, Soldado> mapaSoldados) {
378     String fila = "\t";
379     for (int i = 0; i < ESCALA; i++)
```

```
379         fila += "-----";
380     fila += "-\n";
381     return fila;
382 }
383
384 public static String generarFila(HashMap<String, Soldado> mapaSoldados, int f) {
385     String fila = (f + 1) + "\t";
386     for (int i = 0; i < ESCALA; i++) {
387         fila += "| ";
388         Soldado soldado = mapaSoldados.get(generarLlave(f, i));
389         if (soldado != null) {
390             String nombre = soldado.getNombre();
391             if (nombre.startsWith("Soldado"))
392                 fila += "So" + nombre.charAt(nombre.length() - 3);
393             else
394                 fila += nombre.substring(0, 3);
395             fila += "~" + soldado.getEquipo();
396         } else {
397             fila += "   ";
398         }
399         fila += " ";
400     }
401     fila += "\n";
402     return fila;
403 }
404
405 public static Soldado soldadoMayorVida(ArrayList<Soldado> soldados) {
406     int idx = 0;
407     for (int i = 1; i < soldados.size(); i++) {
408         if (soldados.get(i).getVidaActual() > soldados.get(idx).getVidaActual())
409             idx = i;
410     }
411     return soldados.get(idx);
412 }
413
414 public static double promedioPuntosVida(ArrayList<Soldado> soldados) {
415     int suma = sumaPuntosVida(soldados);
416     return 1.0 * suma / soldados.size();
417 }
418
419 public static void imprimirSoldados(ArrayList<Soldado> soldados) {
420     for (Soldado soldado : soldados)
421         System.out.println(soldado);
422 }
423
424 public static void ordenarSoldadosBurbuja(ArrayList<Soldado> soldados) {
425     for (int i = 0; i < soldados.size() - 1; i++) {
426         for (int j = 0; j < soldados.size() - i - 1; j++) {
427             int vida1 = soldados.get(j).getVidaActual();
428             int vida2 = soldados.get(j + 1).getVidaActual();
429             if (vida1 < vida2)
430                 intercambiar(soldados, j, j + 1);
431         }
432     }
433 }
434
```

```
435 public static void ordenarSoldadosSeleccion(ArrayList<Soldado> soldados) {
436     for (int i = 0; i < soldados.size() - 1; i++) {
437         int idx = i;
438         for (int j = i + 1; j < soldados.size(); j++) {
439             int vida1 = soldados.get(j).getVidaActual();
440             int vida2 = soldados.get(idx).getVidaActual();
441             if (vida1 > vida2)
442                 idx = j;
443         }
444         intercambiar(soldados, i, idx);
445     }
446 }
447
448 public static void realizarCombates(HashMap<String, Soldado> mapaSoldados,
449     ArrayList<Soldado> listaSoldados1,
450     ArrayList<Soldado> listaSoldados2) {
451     int turno = 1;
452     int ganador = -1;
453     while (ganador == -1) {
454         imprimirTablero(mapaSoldados);
455         if (!realizarTurno(mapaSoldados, listaSoldados1, listaSoldados2, turno % 2))
456             return;
457         ganador = verificarGanador(listaSoldados1, listaSoldados2);
458         turno++;
459     }
460     System.out.printf("\nHa ganado el ejercito %d!\n", ganador);
461     System.out.println("Tablero final:");
462     imprimirTablero(mapaSoldados);
463 }
464
465 public static boolean realizarTurno(HashMap<String, Soldado> mapaSoldados,
466     ArrayList<Soldado> listaSoldados1,
467     ArrayList<Soldado> listaSoldados2, int jugador) {
468     int filaI, columnaI, filaF, columnaF;
469     System.out.printf("Ejercito %d\n", jugador);
470     do {
471         System.out.print("Ingrese fila (vacío para cancelar el juego actual): ");
472         String fila = sc.nextLine();
473         if (fila.equals(""))
474             return false;
475         filaI = Integer.parseInt(fila) - 1;
476         System.out.print("Ingrese columna: ");
477         String columna = sc.nextLine();
478         columnaI = charToInt(columna.charAt(0)) - 1;
479     } while (!coordenadaValida(mapaSoldados, filaI, columnaI)
480         || !seleccionValida(mapaSoldados, filaI, columnaI, jugador));
481     do {
482         System.out.print("Ingrese direccion (Punto cardinal): ");
483         String direccion = sc.nextLine();
484         filaF = filaI;
485         columnaF = columnaI;
486         if (direccion.contains("N"))
487             filaF--;
488         else if (direccion.contains("S"))
489             filaF++;
490         if (direccion.contains("E"))
```

```
489         columnaF++;
490         else if (direccion.contains("0"))
491             columnaF--;
492     } while (!coordenadaValida(mapaSoldados, filaF, columnaF)
493             || !destinoValido(mapaSoldados, filaF, columnaF, jugador));
494     String llaveI = generarLlave(filaI, columnaI);
495     String llaveF = generarLlave(filaF, columnaF);
496     if (mapaSoldados.get(llaveF) != null && mapaSoldados.get(llaveF).getEquipo() !=
497         jugador) {
498         realizarPelea(mapaSoldados, listaSoldados1, listaSoldados2, llaveI, llaveF,
499             jugador);
500     } else {
501         mapaSoldados.put(llaveF, mapaSoldados.get(llaveI));
502         mapaSoldados.remove(llaveI);
503     }
504     return true;
505 }
506
507 public static void realizarPelea(HashMap<String, Soldado> mapaSoldados,
508     ArrayList<Soldado> listaSoldados1,
509     ArrayList<Soldado> listaSoldados2, String llaveI, String llaveF, int jugador) {
510     Soldado soldadoAtaca = mapaSoldados.get(llaveI);
511     Soldado soldadoDefiende = mapaSoldados.get(llaveF);
512     System.out.printf("Ocurre una batalla entre %s y %s!\n", soldadoAtaca.getNombre(),
513         soldadoDefiende.getNombre());
514     int vidaAtaca = soldadoAtaca.getVidaActual();
515     int vidaDefiende = soldadoDefiende.getVidaActual();
516     double probabilidad = 1.0 * vidaAtaca / (vidaAtaca + vidaDefiende);
517     System.out.printf("%s tiene %.3f%% probabilidades de vencer!\n",
518         soldadoAtaca.getNombre(), probabilidad * 100);
519     System.out.printf("%s tiene %.3f%% probabilidades de vencer!\n",
520         soldadoDefiende.getNombre(),
521         (1 - probabilidad) * 100);
522     if (probabilidad >= Math.random()) {
523         System.out.printf("Gana el soldado %s!\n", soldadoAtaca.getNombre());
524         soldadoAtaca.mejorar();
525         mapaSoldados.remove(llaveI);
526         mapaSoldados.put(llaveF, soldadoAtaca);
527         if (jugador == 1)
528             listaSoldados2.remove(soldadoDefiende);
529         else if (jugador == 2)
530             listaSoldados1.remove(soldadoDefiende);
531     } else {
532         System.out.printf("Gana el soldado %s!\n", soldadoDefiende.getNombre());
533         soldadoDefiende.mejorar();
534         mapaSoldados.remove(llaveI);
535         if (jugador == 1)
536             listaSoldados1.remove(soldadoAtaca);
537         else if (jugador == 2)
538             listaSoldados2.remove(soldadoDefiende);
539     }
540 }
541
542 public static boolean coordenadaValida(HashMap<String, Soldado> mapaSoldados, int fila,
543     int columna) {
544     if (fila >= 0 && fila < ESCALA && columna >= 0 && columna < ESCALA)
```

```
538         return true;
539     System.out.println("Coordenada no valida.");
540     return false;
541 }
542
543 public static boolean seleccionValida(HashMap<String, Soldado> mapaSoldados, int fila,
544     int columna, int equipo) {
545     String llave = generarLlave(fila, columna);
546     if (mapaSoldados.get(llave) != null && mapaSoldados.get(llave).getEquipo() == equipo)
547         return true;
548     System.out.println("Seleccion no valida.");
549     return false;
550 }
551
552 public static boolean seleccionLlena(HashMap<String, Soldado> mapaSoldados, int fila, int
553     columna) {
554     String llave = generarLlave(fila, columna);
555     if (mapaSoldados.get(llave) != null)
556         return true;
557     System.out.println("Seleccion vacia.");
558     return false;
559 }
560
561 public static boolean seleccionOcupada(HashMap<String, Soldado> mapaSoldados, int fila,
562     int columna) {
563     String llave = generarLlave(fila, columna);
564     if (mapaSoldados.get(llave) == null)
565         return false;
566     System.out.println("Seleccion ya ocupada.");
567     return true;
568 }
569
570 public static boolean destinoValido(HashMap<String, Soldado> mapaSoldados, int fila, int
571     columna, int equipo) {
572     String llave = generarLlave(fila, columna);
573     if (mapaSoldados.get(llave) == null || mapaSoldados.get(llave).getEquipo() != equipo)
574         return true;
575     System.out.println("Destino no valido.");
576     return false;
577 }
578
579 public static int verificarGanador(ArrayList<Soldado> listaSoldados1, ArrayList<Soldado>
580     listaSoldados2) {
581     if (listaSoldados2.isEmpty())
582         return 1;
583     else if (listaSoldados1.isEmpty())
584         return 2;
585     return -1;
586 }
587
588 public static int sumaPuntosVida(ArrayList<Soldado> soldados) {
589     int suma = 0;
590     for (int i = 0; i < soldados.size(); i++)
591         suma += soldados.get(i).getVidaActual();
592     return suma;
593 }
```

```
589
590 public static void intercambiar(ArrayList<Soldado> soldados, int i, int j) {
591     Soldado t = soldados.get(i);
592     soldados.set(i, soldados.get(j));
593     soldados.set(j, t);
594 }
595
596 public static String generarLlave(int fila, int columna) {
597     return fila + "," + columna;
598 }
599
600 public static char intToChar(int n) {
601     return (char) (n + 'A' - 1);
602 }
603
604 public static int charToInt(char c) {
605     return (int) (c - 'A' + 1);
606 }
607 }
```

- Método juegoRapido() contiene la batalla, para permitir el comportamiento iterativo.
- Método inicializarSoldados() crea a los soldados, los ubica en el tablero y los guarda en hashmaps de soldados por separado.
- Método imprimirTablero() imprime el tablero con ayuda de los métodos auxiliares generarEncabezado(), generarSeparacion() y generarFila(), ubicando a los soldados por su numero y ejercito.
- Método soldadoMayorVida() retorna el soldado con mayor vida de un ejército.
- Método promedioPuntosVida() retorna el promedio de los puntos de vida de todos los soldados de un ejército.
- Método imprimirSoldados() imprime los soldados del ejército.
- Método ordenarSoldadosBurbuja() ordena los soldados por vida de mayor a menor, usando ordenamiento burbuja.
- Método ordenarSoldadosSeleccion() ordena los soldados por vida de mayor a menor, usando ordenamiento selección.
- Se reusan los métodos para mostrar los datos de ambos ejércitos.
- Método juegoPersonalizado() contiene un menú que permite modificar el juego antes de iniciar.
- Método crearSoldado() permite crear un soldado.
- Método eliminarSoldado() permite eliminar un soldado.
- Método clonarSoldado() permite clonar un soldado y ponerlo en otra posición.
- Método modificarSoldado() permite modificar los atributos de un soldado.
- Método compararSoldados() permite comparar los atributos de dos soldados.
- Método intercambiarSoldados() permite intercambiar los atributos de dos soldados.
- Método verSoldado() permite mostrar un soldado por nombre.

- Método `sumarNiveles()` muestra la suma de atributos de todo el ejército.
- Método `realizarCombates()` se encarga de realizar el ciclo de juego, permitiendo a los jugadores seleccionar sus soldados, moverlos y atacar, hasta que quede un ganador.

6. Ejecución del código

VideoJuego9.java

```

1  1. Juego Rapido
2  2. Juego Personalizado
3  3. Salir
4  Opcion: 2
5
6  27230
7      A      B      C      D      E      F      G      H      I      J
8  -----
9  1      |      |      |      |      |      |      |      |      |
10 -----
11 2      |      |      |      |      |      |      |      |      |
12 -----
13 3      |      |      |      |      |      |      |      |      |
14 -----
15 4      |      |      |      |      |      |      |      | So0~1 |
16 -----
17 5      |      |      |      |      | So5~1 |      |      |      |
18 -----
19 6      |      |      |      |      |      |      |      |      |
20 -----
21 7      |      |      |      |      |      |      | So2~1 |      | So8~1 |
22 -----
23 8      |      |      |      |      | So1~1 |      |      |      | So3~1 |
24 -----
25 9      |      |      |      |      |      |      | So6~1 |      | So0~2 |
26 -----
27 10     | So7~1 |      |      |      |      |      |      |      | So4~1 |
28 -----
29
30 Ejercito 1:
31 Soldado0X1: Vida: 1. At: 5. Df: 4. Vl: 0
32 Soldado1X1: Vida: 1. At: 4. Df: 3. Vl: 0
33 Soldado2X1: Vida: 2. At: 2. Df: 1. Vl: 0
34 Soldado3X1: Vida: 4. At: 1. Df: 5. Vl: 0
35 Soldado4X1: Vida: 2. At: 2. Df: 4. Vl: 0
36 Soldado5X1: Vida: 3. At: 3. Df: 3. Vl: 0
37 Soldado6X1: Vida: 4. At: 2. Df: 3. Vl: 0
38 Soldado7X1: Vida: 5. At: 5. Df: 1. Vl: 0
39 Soldado8X1: Vida: 1. At: 1. Df: 5. Vl: 0
40
41 Ejercito 2:
42 Soldado0X2: Vida: 4. At: 1. Df: 5. Vl: 0
43
44 Ejercito a gestionar (1/2): 1
45 1. Crear soldado
46 2. Eliminar soldado
47 3. Clonar soldado
48 4. Modificar soldado
49 5. Comparar soldados
50 6. Intercambiar soldados
51 7. Ver soldado
52 8. Ver ejercito

```


53 9. Sumar niveles

54 10. Jugar

55 11. Volver

56 Opcion: 1

57

58 Nombre: Pepe

59 Vida: 5

60 Ataque: 4

61 Defensa: 5

62 Fila: 6

63 Columna: C

64

	A	B	C	D	E	F	G	H	I	J
65										
66										
67	1									
68										
69	2									
70										
71	3									
72										
73	4								So0~1	
74										
75	5					So5~1				
76										
77	6				Pep~1					
78										
79	7						So2~1		So8~1	
80										
81	8					So1~1			So3~1	
82										
83	9						So6~1		So0~2	
84										
85	10		So7~1							So4~1
86										

87

88

Ejercito 1:

89 Soldado0X1: Vida: 1. At: 5. Df: 4. Vl: 0

90 Soldado1X1: Vida: 1. At: 4. Df: 3. Vl: 0

91 Soldado2X1: Vida: 2. At: 2. Df: 1. Vl: 0

92 Soldado3X1: Vida: 4. At: 1. Df: 5. Vl: 0

93 Soldado4X1: Vida: 2. At: 2. Df: 4. Vl: 0

94 Soldado5X1: Vida: 3. At: 3. Df: 3. Vl: 0

95 Soldado6X1: Vida: 4. At: 2. Df: 3. Vl: 0

96 Soldado7X1: Vida: 5. At: 5. Df: 1. Vl: 0

97 Soldado8X1: Vida: 1. At: 1. Df: 5. Vl: 0

98 Pepe: Vida: 5. At: 4. Df: 5. Vl: 0

99

100 Ejercito 2:

101 Soldado0X2: Vida: 4. At: 1. Df: 5. Vl: 0

102

103 Ejercito a gestionar (1/2): 1

104 1. Crear soldado

105 2. Eliminar soldado

106 3. Clonar soldado

107 4. Modificar soldado

108 5. Comparar soldados

109 6. Intercambiar soldados

110 7. Ver soldado

111 8. Ver ejercito

112 9. Sumar niveles

113 10. Jugar

114 11. Volver

115 Opcion: 2

116

117 Fila: 10

118 Columna: A

119

	A	B	C	D	E	F	G	H	I	J
1										
2										
3										
4									So0~1	
5					So5~1					
6			Pep~1							
7							So2~1		So8~1	
8					So1~1				So3~1	
9							So6~1		So0~2	
10										So4~1

141

142

143 Ejercito 1:

144 Soldado0X1: Vida: 1. At: 5. Df: 4. Vl: 0

145 Soldado1X1: Vida: 1. At: 4. Df: 3. Vl: 0

146 Soldado2X1: Vida: 2. At: 2. Df: 1. Vl: 0

147 Soldado3X1: Vida: 4. At: 1. Df: 5. Vl: 0

148 Soldado4X1: Vida: 2. At: 2. Df: 4. Vl: 0

149 Soldado5X1: Vida: 3. At: 3. Df: 3. Vl: 0

150 Soldado6X1: Vida: 4. At: 2. Df: 3. Vl: 0

151 Soldado8X1: Vida: 1. At: 1. Df: 5. Vl: 0

152 Pepe: Vida: 5. At: 4. Df: 5. Vl: 0

153

154 Ejercito 2:

155 Soldado0X2: Vida: 4. At: 1. Df: 5. Vl: 0

156

157 Ejercito a gestionar (1/2): 1

158 1. Crear soldado

159 2. Eliminar soldado

160 3. Clonar soldado

161 4. Modificar soldado

162 5. Comparar soldados

163 6. Intercambiar soldados

164 7. Ver soldado

165 8. Ver ejercito

166 9. Sumar niveles

167 10. Jugar

168 11. Volver

169 Opcion: 3

170

171 Fila de origen: 6

172 Columna de origen: C

173 Fila de destino: 5

174 Columna de destino: C

175

	A	B	C	D	E	F	G	H	I	J
1										
2										
3										
4									So0~1	
5			Pep~1		So5~1					
6			Pep~1							
7							So2~1		So8~1	
8					So1~1				So3~1	
9							So6~1		So0~2	
10										So4~1

197

198

199 Ejercito 1:

200 Soldado0X1: Vida: 1. At: 5. Df: 4. Vl: 0

201 Soldado1X1: Vida: 1. At: 4. Df: 3. Vl: 0

202 Soldado2X1: Vida: 2. At: 2. Df: 1. Vl: 0

203 Soldado3X1: Vida: 4. At: 1. Df: 5. Vl: 0

204 Soldado4X1: Vida: 2. At: 2. Df: 4. Vl: 0

205 Soldado5X1: Vida: 3. At: 3. Df: 3. Vl: 0

206 Soldado6X1: Vida: 4. At: 2. Df: 3. Vl: 0

207 Soldado8X1: Vida: 1. At: 1. Df: 5. Vl: 0

208 Pepe: Vida: 5. At: 4. Df: 5. Vl: 0

209 Pepe: Vida: 5. At: 4. Df: 5. Vl: 0

210

211 Ejercito 2:

212 Soldado0X2: Vida: 4. At: 1. Df: 5. Vl: 0

213

214 Ejercito a gestionar (1/2): 1

215 1. Crear soldado

216 2. Eliminar soldado

217 3. Clonar soldado

218 4. Modificar soldado

219 5. Comparar soldados

220 6. Intercambiar soldados

221 7. Ver soldado
222 8. Ver ejercito
223 9. Sumar niveles
224 10. Jugar
225 11. Volver
226 Opcion: 4
227

228 Fila: 5
229 Columna: E

230 1. Ataque
231 2. Defensa
232 3. Vida

233 Opcion: 3
234 Nuevo valor: 5
235

	A	B	C	D	E	F	G	H	I	J
236										
237										
238	1									
239										
240	2									
241										
242	3									
243										
244	4								So0~1	
245										
246	5			Pep~1		So5~1				
247										
248	6			Pep~1						
249										
250	7						So2~1		So8~1	
251										
252	8					So1~1			So3~1	
253										
254	9						So6~1		So0~2	
255										
256	10									So4~1
257										
258										

259 Ejercito 1:
260 Soldado0X1: Vida: 1. At: 5. Df: 4. Vl: 0
261 Soldado1X1: Vida: 1. At: 4. Df: 3. Vl: 0
262 Soldado2X1: Vida: 2. At: 2. Df: 1. Vl: 0
263 Soldado3X1: Vida: 4. At: 1. Df: 5. Vl: 0
264 Soldado4X1: Vida: 2. At: 2. Df: 4. Vl: 0
265 Soldado5X1: Vida: 5. At: 3. Df: 3. Vl: 0
266 Soldado6X1: Vida: 4. At: 2. Df: 3. Vl: 0
267 Soldado8X1: Vida: 1. At: 1. Df: 5. Vl: 0
268 Pepe: Vida: 5. At: 4. Df: 5. Vl: 0
269 Pepe: Vida: 5. At: 4. Df: 5. Vl: 0
270

271 Ejercito 2:
272 Soldado0X2: Vida: 4. At: 1. Df: 5. Vl: 0
273

274 Ejercito a gestionar (1/2): 1
275 1. Crear soldado
276 2. Eliminar soldado

277 3. Clonar soldado
278 4. Modificar soldado
279 5. Comparar soldados
280 6. Intercambiar soldados
281 7. Ver soldado
282 8. Ver ejercito
283 9. Sumar niveles
284 10. Jugar
285 11. Volver
286 Opcion: 5
287

288 Fila soldado 1: 5
289 Columna: C
290 Fila soldado 2: 6
291 Columna: C
292 Nombres iguales
293 Ataques iguales
294 Defensas iguales
295 Vidas iguales
296 Estados vitales iguales
297

	A	B	C	D	E	F	G	H	I	J
299	-----									
300	1									
301	-----									
302	2									
303	-----									
304	3									
305	-----									
306	4								So0~1	
307	-----									
308	5			Pep~1		So5~1				
309	-----									
310	6			Pep~1						
311	-----									
312	7						So2~1		So8~1	
313	-----									
314	8					So1~1			So3~1	
315	-----									
316	9						So6~1		So0~2	
317	-----									
318	10									So4~1
319	-----									

320
321 Ejercito 1:
322 Soldado0X1: Vida: 1. At: 5. Df: 4. Vl: 0
323 Soldado1X1: Vida: 1. At: 4. Df: 3. Vl: 0
324 Soldado2X1: Vida: 2. At: 2. Df: 1. Vl: 0
325 Soldado3X1: Vida: 4. At: 1. Df: 5. Vl: 0
326 Soldado4X1: Vida: 2. At: 2. Df: 4. Vl: 0
327 Soldado5X1: Vida: 5. At: 3. Df: 3. Vl: 0
328 Soldado6X1: Vida: 4. At: 2. Df: 3. Vl: 0
329 Soldado8X1: Vida: 1. At: 1. Df: 5. Vl: 0
330 Pepe: Vida: 5. At: 4. Df: 5. Vl: 0
331 Pepe: Vida: 5. At: 4. Df: 5. Vl: 0
332

333 Ejercito 2:
334 Soldado0X2: Vida: 4. At: 1. Df: 5. Vl: 0

335
336 Ejercito a gestionar (1/2): 1

- 337 1. Crear soldado
- 338 2. Eliminar soldado
- 339 3. Clonar soldado
- 340 4. Modificar soldado
- 341 5. Comparar soldados
- 342 6. Intercambiar soldados
- 343 7. Ver soldado
- 344 8. Ver ejercito
- 345 9. Sumar niveles
- 346 10. Jugar
- 347 11. Volver

348 Opcion: 6

349
350 Fila soldado 1: 7
351 Columna soldado 1: G
352 Fila soldado 2: 7
353 Columna soldado 2: I

	A	B	C	D	E	F	G	H	I	J
1										
2										
3										
4									So0~1	
5			Pep~1		So5~1					
6			Pep~1							
7							So8~1		So2~1	
8					So1~1				So3~1	
9							So6~1		So0~2	
10										So4~1

378 Ejercito 1:

379 Soldado0X1: Vida: 1. At: 5. Df: 4. Vl: 0
380 Soldado1X1: Vida: 1. At: 4. Df: 3. Vl: 0
381 Soldado2X1: Vida: 2. At: 2. Df: 1. Vl: 0
382 Soldado3X1: Vida: 4. At: 1. Df: 5. Vl: 0
383 Soldado4X1: Vida: 2. At: 2. Df: 4. Vl: 0
384 Soldado5X1: Vida: 5. At: 3. Df: 3. Vl: 0
385 Soldado6X1: Vida: 4. At: 2. Df: 3. Vl: 0
386 Soldado8X1: Vida: 1. At: 1. Df: 5. Vl: 0
387 Pepe: Vida: 5. At: 4. Df: 5. Vl: 0
388 Pepe: Vida: 5. At: 4. Df: 5. Vl: 0

389
390 Ejercito 2:
391 Soldado0X2: Vida: 4. At: 1. Df: 5. Vl: 0
392

393 Ejercito a gestionar (1/2): 1

- 394 1. Crear soldado
- 395 2. Eliminar soldado
- 396 3. Clonar soldado
- 397 4. Modificar soldado
- 398 5. Comparar soldados
- 399 6. Intercambiar soldados
- 400 7. Ver soldado
- 401 8. Ver ejercito
- 402 9. Sumar niveles
- 403 10. Jugar
- 404 11. Volver
- 405 Opcion: 7
- 406

407 Nombre: Soldado4X1

408 Soldado4X1: Vida: 2. At: 2. Df: 4. Vl: 0

	A	B	C	D	E	F	G	H	I	J
1										
2										
3										
4									So0~1	
5			Pep~1		So5~1					
6			Pep~1							
7							So8~1		So2~1	
8					So1~1				So3~1	
9							So6~1		So0~2	
10										So4~1

433 Ejercito 1:

434 Soldado0X1: Vida: 1. At: 5. Df: 4. Vl: 0
435 Soldado1X1: Vida: 1. At: 4. Df: 3. Vl: 0
436 Soldado2X1: Vida: 2. At: 2. Df: 1. Vl: 0
437 Soldado3X1: Vida: 4. At: 1. Df: 5. Vl: 0
438 Soldado4X1: Vida: 2. At: 2. Df: 4. Vl: 0
439 Soldado5X1: Vida: 5. At: 3. Df: 3. Vl: 0
440 Soldado6X1: Vida: 4. At: 2. Df: 3. Vl: 0
441 Soldado8X1: Vida: 1. At: 1. Df: 5. Vl: 0
442 Pepe: Vida: 5. At: 4. Df: 5. Vl: 0
443 Pepe: Vida: 5. At: 4. Df: 5. Vl: 0
444

445 Ejercito 2:
446 Soldado0X2: Vida: 4. At: 1. Df: 5. Vl: 0

447
448 Ejercito a gestionar (1/2): 1

- 449 1. Crear soldado
- 450 2. Eliminar soldado
- 451 3. Clonar soldado
- 452 4. Modificar soldado
- 453 5. Comparar soldados
- 454 6. Intercambiar soldados
- 455 7. Ver soldado
- 456 8. Ver ejercito
- 457 9. Sumar niveles
- 458 10. Jugar
- 459 11. Volver
- 460 Opcion: 8

461
462 Soldado0X1: Vida: 1. At: 5. Df: 4. Vl: 0
463 Soldado1X1: Vida: 1. At: 4. Df: 3. Vl: 0
464 Soldado2X1: Vida: 2. At: 2. Df: 1. Vl: 0
465 Soldado3X1: Vida: 4. At: 1. Df: 5. Vl: 0
466 Soldado4X1: Vida: 2. At: 2. Df: 4. Vl: 0
467 Soldado5X1: Vida: 5. At: 3. Df: 3. Vl: 0
468 Soldado6X1: Vida: 4. At: 2. Df: 3. Vl: 0
469 Soldado8X1: Vida: 1. At: 1. Df: 5. Vl: 0
470 Pepe: Vida: 5. At: 4. Df: 5. Vl: 0
471 Pepe: Vida: 5. At: 4. Df: 5. Vl: 0

	A	B	C	D	E	F	G	H	I	J
474	-----									
475	1									
476		-----								
477	2									
478		-----								
479	3									
480		-----								
481	4								So0~1	
482		-----								
483	5			Pep~1		So5~1				
484		-----								
485	6			Pep~1						
486		-----								
487	7						So8~1		So2~1	
488		-----								
489	8					So1~1			So3~1	
490		-----								
491	9						So6~1		So0~2	
492		-----								
493	10								So4~1	
494		-----								

495
496 Ejercito 1:
497 Soldado0X1: Vida: 1. At: 5. Df: 4. Vl: 0
498 Soldado1X1: Vida: 1. At: 4. Df: 3. Vl: 0
499 Soldado2X1: Vida: 2. At: 2. Df: 1. Vl: 0
500 Soldado3X1: Vida: 4. At: 1. Df: 5. Vl: 0

501 Soldado4X1: Vida: 2. At: 2. Df: 4. Vl: 0
502 Soldado5X1: Vida: 5. At: 3. Df: 3. Vl: 0
503 Soldado6X1: Vida: 4. At: 2. Df: 3. Vl: 0
504 Soldado8X1: Vida: 1. At: 1. Df: 5. Vl: 0
505 Pepe: Vida: 5. At: 4. Df: 5. Vl: 0
506 Pepe: Vida: 5. At: 4. Df: 5. Vl: 0

507
508 Ejercito 2:

509 Soldado0X2: Vida: 4. At: 1. Df: 5. Vl: 0

510

511 Ejercito a gestionar (1/2): 1

512 1. Crear soldado

513 2. Eliminar soldado

514 3. Clonar soldado

515 4. Modificar soldado

516 5. Comparar soldados

517 6. Intercambiar soldados

518 7. Ver soldado

519 8. Ver ejercito

520 9. Sumar niveles

521 10. Jugar

522 11. Volver

523 Opcion: 9

524

525 Vida total: 30

526 Ataque total: 28

527 Defensa total: 38

528 Velocidad total: 0

529

	A	B	C	D	E	F	G	H	I	J
530										
531										
532	1									
533										
534	2									
535										
536	3									
537										
538	4								So0~1	
539										
540	5			Pep~1		So5~1				
541										
542	6			Pep~1						
543										
544	7						So8~1		So2~1	
545										
546	8					So1~1			So3~1	
547										
548	9						So6~1		So0~2	
549										
550	10									So4~1
551										

552

553 Ejercito 1:

554 Soldado0X1: Vida: 1. At: 5. Df: 4. Vl: 0

555 Soldado1X1: Vida: 1. At: 4. Df: 3. Vl: 0

556 Soldado2X1: Vida: 2. At: 2. Df: 1. Vl: 0

557 Soldado3X1: Vida: 4. At: 1. Df: 5. Vl: 0
558 Soldado4X1: Vida: 2. At: 2. Df: 4. Vl: 0
559 Soldado5X1: Vida: 5. At: 3. Df: 3. Vl: 0
560 Soldado6X1: Vida: 4. At: 2. Df: 3. Vl: 0
561 Soldado8X1: Vida: 1. At: 1. Df: 5. Vl: 0
562 Pepe: Vida: 5. At: 4. Df: 5. Vl: 0
563 Pepe: Vida: 5. At: 4. Df: 5. Vl: 0

564

565 Ejercito 2:

566 Soldado0X2: Vida: 4. At: 1. Df: 5. Vl: 0

567

568 Ejercito a gestionar (1/2): 1

569 1. Crear soldado

570 2. Eliminar soldado

571 3. Clonar soldado

572 4. Modificar soldado

573 5. Comparar soldados

574 6. Intercambiar soldados

575 7. Ver soldado

576 8. Ver ejercito

577 9. Sumar niveles

578 10. Jugar

579 11. Volver

580 Opcion: 10

581

582

	A	B	C	D	E	F	G	H	I	J
584	-----									
585 1										
586	-----									
587 2										
588	-----									
589 3										
590	-----									
591 4									So0~1	
592	-----									
593 5			Pep~1		So5~1					
594	-----									
595 6			Pep~1							
596	-----									
597 7							So8~1		So2~1	
598	-----									
599 8					So1~1				So3~1	
600	-----									
601 9							So6~1		So0~2	
602	-----									
603 10										So4~1
604	-----									

605 Ejercito 1

606 Ingrese fila (vacio para cancelar el juego actual): 10

607 Ingrese columna: J

608 Ingrese direccion (Punto cardinal): NO

609 Ocurre una batalla entre Soldado4X1 y Soldado0X2!

610 Soldado4X1 tiene 33.333% probabilidades de vencer!

611 Soldado0X2 tiene 66.667% probabilidades de vencer!

612 Gana el soldado Soldado4X1!

613

614 Ha ganado el ejercito 1!

615 Tablero final:

	A	B	C	D	E	F	G	H	I	J
1										
2										
3										
4									So0~1	
5			Pep~1		So5~1					
6			Pep~1							
7							So8~1		So2~1	
8					So1~1				So3~1	
9							So6~1		So4~1	
10										

638

639

640 Volver a jugar? (S/N): N

641 1. Juego Rapido

642 2. Juego Personalizado

643 3. Salir

644 Opcion: 3

7. Estructura de laboratorio 12

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

```
lab12/  
|--- Soldado.java  
|--- VideoJuego9.java  
|--- commits.bash  
|--- ejec01.bash  
|--- Informe.tex  
|--- Informe.pdf  
|--- img  
|   |--- logo_abet.png  
|   |--- logo_episunsa.png  
|   |--- logo_unsa.jpg  
|   |--- commit01.jpg  
|   |--- commit02.jpg  
|   |--- commit03.jpg  
|   |--- commit04.jpg  
|   |--- commit05.jpg  
|   |--- commit06.jpg  
|   |--- commit07.jpg  
|   |--- commit08.jpg  
|   |--- commit09.jpg  
|   |--- commit10.jpg  
|   |--- commit11.jpg
```

8. Rúbricas

8.1. Entregable Informe

Tipo de Informe

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

8.2. Rúbrica para el contenido del Informe y demostración

- El alumno debe marcar o dejar en blanco en celdas de la columna **Checklist** si cumple con el ítem correspondiente.
- Si un alumno supera la fecha de entrega, su calificación será sobre la nota mínima aprobatoria, siempre y cuando cumpla con todos los ítems.
- El alumno debe autocalificarse en la columna **Estudiante** de acuerdo a la siguiente tabla:

Niveles de desempeño

Puntos	Nivel			
	Insatisfactorio 25 %	En Proceso 50 %	Satisfactorio 75 %	Sobresaliente 100 %
2.0	0.5	1.0	1.5	2.0
4.0	1.0	2.0	3.0	4.0

Rúbrica para contenido del Informe y demostración

Contenido y demostración		Puntos	Checklist	Estudiante	Profesor
1. GitHub	Hay enlace URL activo del directorio para el laboratorio hacia su repositorio GitHub con código fuente terminado y fácil de revisar.	2	X	2	
2. Commits	Hay capturas de pantalla de los commits más importantes con sus explicaciones detalladas. (El profesor puede preguntar para refrendar calificación).	4	X	3	
3. Código fuente	Hay porciones de código fuente importantes con numeración y explicaciones detalladas de sus funciones.	2	X	2	
4. Ejecución	Se incluyen ejecuciones/pruebas del código fuente explicadas gradualmente.	2	X	1.5	
5. Pregunta	Se responde con completitud a la pregunta formulada en la tarea. (El profesor puede preguntar para refrendar calificación).	2	X	2	
6. Fechas	Las fechas de modificación del código fuente estan dentro de los plazos de fecha de entrega establecidos.	2	X	2	
7. Ortografía	El documento no muestra errores ortográficos.	2	X	1.5	
8. Madurez	El Informe muestra de manera general una evolución de la madurez del código fuente, explicaciones puntuales pero precisas y un acabado impecable. (El profesor puede preguntar para refrendar calificación).	4	X	4	
Total		20		18	

9. Referencias

- Aedo, M. y Castro, E. (2021). FUNDAMENTOS DE PROGRAMACIÓN 2 - Tópicos de Programación Orientada a Objetos. Editorial UNSA.