Informe de Laboratorio 22
Tema: Laboratorio 22

Nota

Estudiante	Escuela	Asignatura
Victor Mamani Anahua vmamanian@unsa.edu.pe	Escuela Profesional de Ingeniería de Sistemas	Fundamentos de la Programación II Semestre: II Código: 20230489

Laboratorio	Tema	Duración
22	Laboratorio 22	04 horas

Semestre académico	Fecha de inicio	Fecha de entrega
2023 - B	Del 15 Enero 2024	Al 22 Enero 2024

1. Tarea

- Cree una versión del videojuego de estrategia usando componentes básicos GUI: Etiquetas, botones, cuadros de texto, JOptionPane, Color.
- Además, utilizar componentes avanzados GUI: Layouts, JPanel, áreas de texto, checkbox, botones de radio y combobox.
- Considerar nivel estratégico y táctico.
- Considerar hasta las unidades especiales de los reinos.
- Hacerlo iterativo.

2. Equipos, materiales y temas utilizados

- Sistema Operativo Ubuntu GNU Linux 23 lunar 64 bits Kernell 6.2.v
- Visual Studio Code.
- VIM 9.0.
- OpenJDK 64-Bits 19.0.7.
- Git 2.39.2.
- Cuenta en GitHub con el correo institucional.
- Programación Orientada a Objetos.
- Actividades del Laboratorio 22.

3. URL de Repositorio Github

- URL del Repositorio GitHub para clonar o recuperar.
- `https://github.com/VictorMA18/fp2-23b.git`
- URL para el laboratorio 22 en el Repositorio GitHub.
- `https://github.com/VictorMA18/fp2-23b/tree/main/Fase03/Lab22`

4. Actividades del Laboratorio 22

4.1. Ejercicio Soldado(Herencias)

- En esta seccion solo reutilizamos las Herencias de la clase Soldado.
- El codigo y el commit seria el siguiente:

Listing 1: Commit

```
$ git commit -m "Agregando la clase Soldado y Espadachin para poder hacer el juego
    bueno solo en la clase espadachin usamos la herencia que nos deja la clase Soldado
    y tambien creamos la funcion muroEscudo() la cual devuelve como mensaje el uso de
    esta habilidad defensiva y los getters y setters"
```

Listing 2: Las lineas de codigos de la clase Espadachin creada:

```
public class Espadachin extends Soldado{
   private int swordlth;
   public Espadachin(String name , int attacklevel, int defenselevel, int lifelevel,
       int speed, String attitude ,boolean lives, int row, String column, int swordlth){
      super(name, attacklevel, defenselevel, lifelevel, speed, attitude, lives, row,
          column);
      this.swordlth = swordlth;
   }
   public void muroEscudo(){
      System.out.println("Usted uso la habilidad muro de Escudos");
   }
   public int getSwordlth(){
      return swordlth;
   }
   public void setSwordlth(int n){
      this.swordlth = n;
   }
}
```

Listing 3: Las lineas de codigos de la clase Caballero creada:

```
public class Caballero extends Soldado{
   private boolean montar;
   private String arma;
   public Caballero(){
   }
   public Caballero(String name , int attacklevel, int defenselevel, int lifelevel, int
       speed, String attitude ,boolean lives, int row, String column,boolean montar){
```

```
        super(name, attacklevel, defenselevel, lifelevel, speed, attitude, lives, row,
            column);
        this.montar = montar;
    }
    public void montar(){
        if(!this.montar){
            this.arma = "Lanza";
            this.embestir();
        }
    }
    public void desmontar(){
        if (this.montar) {
            this.arma = "Espada";
        }
    }
    public void embestir(){
        if(!montar){
            this.atacar();
            this.atacar();
        }else{
            this.atacar();
            this.atacar();
            this.atacar();
        }
    }
    public String getArma(){
        return arma;
    }
}
```

Listing 4: Las lineas de codigos de la clase Arquero creada:

```
public class Arquero extends Soldado{
    private int flechas;
    public Arquero(){

    }
    public Arquero(String name , int attacklevel, int defenselevel, int lifelevel, int
        speed, String attitude ,boolean lives, int row, String column, int flechas){
        super(name, attacklevel, defenselevel, lifelevel, speed, attitude, lives, row,
            column);
        this.flechas = flechas;
    }
    public void disparar(){
        if(this.flechas == 0){
            System.out.println("El arquero ya tiene flechas para poder disparar");
        }else{
            this.flechas = flechas - 1;
            this.atacar();
        }
    }
    public void setFlechas(int n){
        this.flechas = n;
    }
    public int getFlechas(){
        return flechas;
```

```
    }
}
```

Listing 5: Las lineas de codigos de la clase Lancero creada:

```
public class Lancero extends Soldado{
   private int lancelth;
   public Lancero(){
   }
   public Lancero(String name , int attacklevel, int defenselevel, int lifelevel, int
       speed, String attitude ,boolean lives, int row, String column, int lancelth){
      super(name, attacklevel, defenselevel, lifelevel, speed, attitude, lives, row,
          column);
      this.lancelth = lancelth;
   }
   public void schiltrom(){
      this.setDefenseLevel(this.getDefenseLevel() + 1);
      System.out.println("El lancero uso el schiltrom su nivel de defensa subio 1
          punto");
   }
   public void setLancelth(int n){
      this.lancelth = n;
   }
   public int getLancelth(){
      return lancelth;
   }
}
```

Listing 6: Las lineas de codigos de la clase Soldado creada:

```
// Laboratorio Nro 22 - Ejercicio Soldado
// Autor: Mamani Anahua Victor Narciso
// Colaboro:
// Tiempo:
import java.util.*;
public class Soldado { //CREAMOS LA CLASE SOLDODADO PARA PODER USAR UN ARREGLO
    BIDIMENSIONAL DONDE NECESITAMOS LA VIDA , EL NOMBRE DEL SOLDADO Y TAMBIEN SU
    POSICION COMO LA FILA Y LA COLUMNA

   private String name;
   private int lifeactual;
   private int row;
   private String column;
   private int attacklevel;
   private int defenselevel;
   private int lifelevel;
   private int speed;
   private String attitude;
   private boolean lives;

   Random rdm = new Random();

   //Anadiendo metodo que nos permita que un arreglo tenga datos nulos si este esta
       vacio
   public Soldado(){
```

```
        this.name = "";
        this.row = 0;
        this.column = "";
        this.attacklevel = 0;
        this.defenselevel = 0;
        this.lifelevel = 0;
        this.lifeactual = 0;
        this.speed = 0;
        this.attitude = "";
        this.lives = false;
    }

    //Constructor
    public Soldado(String name, int health, int row, String column){
        this.name = name;
        this.lifeactual = health;
        this.lifelevel = health;
        this.lifeactual = health;
        this.row = row;
        this.column = column;
        this.lives = true;

        //YA QUE ESTOS DATOS SERIAN ALEATORIOS YA QUE SE ESTARIA CREANDO EL SOLDADO
            TENDRIAMOS DATOS QUE SERIAN COMO ATTACKLEVEL DEFENSELEVEL EL CUAL TENDRIAN
            QUE SER ALEATORIOS
        this.attacklevel = rdm.nextInt(5) + 1;
        this.defenselevel = rdm.nextInt(5) + 1;

    }

    //Constructor para los diferentes niveles como de vidad defensa ataque velocidad
    public Soldado(String name , int attacklevel, int defenselevel, int lifelevel, int
         speed, String attitude ,boolean lives, int row, String column) {
        this.name = name;
        this.attacklevel = attacklevel;
        this.defenselevel = defenselevel;
        this.lifeactual = lifelevel;
        this.lifelevel = lifelevel;
        this.speed = speed;
        this.lives = lives;
        this.row = row;
        this.column = column;
        this.attitude = attitude;
    }

    //Metodos necesarios como avanzar defender huir al seratacado al retroceder
    public void advance(){
        this.speed = getSpeed() + 1;
        System.out.println("El soldado " + this.name + "avanzo");
    }
    public void defense(){
        this.speed = 0;
        this.attitude = "DEFENSIVA";
        System.out.println("El soldado " + this.name + "esta defendiendo");
    }
    public void flee(){
```

```
        this.speed = getSpeed() + 2;
        this.attitude = "HUYE";
        System.out.println("El soldado " + this.name + "esta huyendo");
    }
    public void back(){
        System.out.println("El soldado " + this.name + "esta retrocediendo");
        if(this.speed == 0){
            this.speed = rdm.nextInt(5) - 5;
        }else{
            if(this.speed > 0){
                this.speed = 0;
                this.attitude = "DEFENSIVA";
            }
        }
    }
    public void atacar(){
        this.speed += 1;
        this.attitude = "Atacar";
        this.lifeactual += 1;
        if(this.lifeactual == 0){
            morir();
        }
    }
    public void attack(Soldado soldier){
        if(this.getLifeActual() > soldier.getLifeActual()){
            int life = this.getLifeActual() - soldier.getLifeActual();
            this.setLifeActual(life);
            this.setLifeLevel(life);
            soldier.lives = false;
            soldier.morir();
            System.out.println(this.name + " asesino al soldado " + soldier.name);
        }else if(soldier.getLifeActual() > this.getLifeActual()){
            int life = soldier.getLifeActual() - this.getLifeActual();
            this.lives = false;
            this.morir();
            soldier.setLifeActual(life);
            soldier.setLifeLevel(life);
            System.out.println(soldier.name + " asesino al soldado " + this.name);
        }else{
            this.lives = false;
            this.morir();
            soldier.lives = false;
            soldier.morir();
            System.out.println("los 2 soldados se asesinaron");
        }
    }
    public void morir(){
        this.lives = false;
        this.attitude = "SOLDADO MUERTO";
    }

    // Metodos mutadores
    public void setName(String n){
        name = n;
    }
    public void setLifeActual(int p){
```

```
        lifeactual = p;
    }
    public void setRow(int b){
        row = b;
    }
    public void setColumn(String c){
        column = c;
    }
    public void setAttackLevel(int attacklevel) {
        this.attacklevel = attacklevel;
    }
    public void setDefenseLevel(int defenselevel) {
        this.defenselevel = defenselevel;
    }
    public void setLifeLevel(int lifelevel){
        this.lifelevel = lifelevel;
    }
    public void setSpeed(int speed) {
        this.speed = speed;
    }
    public void setAttitude(String attitude) {
        this.attitude = attitude;
    }
    public void setLives(boolean lives) {
        this.lives = lives;
    }

    // Metodos accesores
    public String getName(){
        return name;
    }
    public int getLifeActual(){
        return lifeactual;
    }
    public int getRow(){
        return row;
    }
    public String getColumn(){
        return column;
    }
    public int getAttackLevel() {
        return attacklevel;
    }
    public int getDefenseLevel() {
        return defenselevel;
    }
    public int getLifeLevel(){
        return lifelevel;
    }
    public int getSpeed() {
        return speed;
    }
    public String getAttitude() {
        return attitude;
    }
    public boolean getLives() {
```

```
        return lives;
    }

    // Completar con otros metodos necesarios
    public String toString(){ //CREAMOS ESTE METODO PARA IMPRIMIR LOS DATOS DEl OBJETO
        String join = "\nNombre: " + getName() + "\nVida: " + getLifeActual() + "\nFila: "
            + getRow() + "\nColumna: " + getColumn() + "\nNivel de ataque: " +
            getAttackLevel() + "\nNivel de Defensa: " + getDefenseLevel() + "\nNivel de
            vida: " + getLifeLevel() + "\nVelocidad: " + getSpeed() + "\nActitud: " +
            getAttitude() + "\nEstado: " + getLives(); //Agregamos un espaciador para
            poder separar
        return join;
    }

}
```

4.2. Ejercicio Ejercito y Mapa

- En esta seccion En la clase Ejercito creamos un contructor null y constructor el cual reciba parametros para reconocer a este le agregamos los metodos viewsoldiers() , longerlife() , averagelife(), rankingInsercionLife() y tambien sus getters y setters esto con el fin que nos ayude a simplificar algunas funciones las cuales sean representadas mejor con una clase Ejercito.
- Por parte de clase Mapa integramos 2 objetos de la clase Ejercito las cuales nos ayudan con sus metodos para ver la informacion de la batalla entre los 2 ejercitos.
- El codigo y el commit seria el siguiente:

Listing 7: Commit

```
$ git commit -m "Creamos una clase Ejercito la cual nos va a poder ayudar al momento de
    querer datos de este como los soldados , el soldado de mayor vida promedio y
    tambien tenenemos en la clase Mapa la cual creamos objetos de esta clase Ejercito
    las cuales vamos a poder usar sus metodos"
```

Listing 8: Las lineas de codigos de la clase Ejercito creada:

```
import java.util.*;
class Ejercito {
    private ArrayList<ArrayList<Soldado>> army;
    private String kingdom;
    public Ejercito(){
    }
    public Ejercito(ArrayList<ArrayList<Soldado>> army, String kingdom){
        this.army = army;
        this.kingdom = kingdom;
    }
    public void viewSoldiers(String armyespe, int num, ArrayList<ArrayList<Soldado>>
        army){
        int numbersoldiers = 0;
        System.out.println("El Ejercito " + armyespe + " del " + num + " ejercito sus
            soldados son :");
        for(int i = 0; i < 10; i++){ //ITERACION
            for(int j = 0; j < 10 ; j++){//ITERACION
```

```
            if(army.get(i).get(j) != null){
               System.out.println("\n**********************************");
               System.out.println("El " + (numbersoldiers + 1) + " soldado es: ");
               System.out.println(army.get(i).get(j).toString());
               numbersoldiers++;
            }
         }
      }
   }
   public void longerLife(ArrayList<ArrayList<Soldado>> army, String kingdom){
      int mayor = 0;//METODO CREADO PARA PODER PERMITIRNOS A CONOCER EL SOLDADO CON
          MAYOR VIDA DE CADA EJERCITO
      Soldado soldier = null;
      for(int i = 0; i < army.size(); i++){
         for(int j = 0; j < army.get(i).size(); j++){
            if(army.get(i).get(j) != null){ //COMPROBACION QUE HACEMOS PARA PODER DECIR
                QUE EL CASILLERO DONDE ESTAMOS ES UN SOLDADO QUE EXISTE
               if(army.get(i).get(j).getLifeActual() > mayor){ //COMPARAMOS PUNTOS DE
                   VIDA DE CADA SOLDADO PARA VER QUIEN ES EL MAYOR
                  mayor = army.get(i).get(j).getLifeActual();
                  soldier = army.get(i).get(j);
               }
            }
         }
      }
      System.out.println("El soldado con mayor vida del Ejercito " + kingdom + " es: ");
      System.out.println(soldier.toString());//IMPRIMIMOS SUS DATOS PARA PODER VER DE
          QUE SOLDADO SE TRATA
      System.out.println("**********************************");
   }
   public double averageLife(ArrayList<ArrayList<Soldado>> army , String kingdom){
      int sum = 0;
      int count = 0;
      System.out.println("El promedio de puntos de vida del Ejercito " + kingdom + " es:
          ");
      for(int i = 0; i < 10; i++){
         for(int j = 0; j < 10; j++){ //ITERACION LA CUAL NOS AYUDA A PASAR POR TODOS
             LOS SOLDADOS DE CADA EJERCITO
            if(army.get(i).get(j) != null){
               sum += army.get(i).get(j).getLifeActual();
               count++;
            }
         }
      }
      if(sum != 0){
         double avg = sum / (count * 1.0);
         System.out.println(avg); // DAMOS A CONOCER EL PROMEDIO DE VIDA DE CADA EJERCITO
         System.out.println("**********************************");
         return avg;
      }else{
         double avg = 0;
         System.out.println(avg); // DAMOS A CONOCER EL PROMEDIO DE VIDA DE CADA EJERCITO
         System.out.println("**********************************");
         return avg;
      }
   }
```

```
public void rankingInsercionLife(ArrayList<ArrayList<Soldado>> army , String
    kingdom){
   System.out.println("\nEl Ejercito " + kingdom + " ordenando por metodo insercion:
       ");
   int count = 0;
   for(int i = 0; i < 10; i++){
      for(int j = 0; j < 10; j++){ //ITERACION LA CUAL NOS AYUDA A PASAR POR TODOS
          LOS SOLDADOS DE CADA EJERCITO
         if(army.get(i).get(j) != null){
            count++;
         }
      }
   }
   System.out.println("-----------------------------------------");
   System.out.println("Mostrando Ranking del Ejercito " + kingdom + " ..... //////
       --->");
   Soldado[] soldados = new Soldado[count];
   int x = 0;
   for(int i = 0; i < 10; i++){
      for(int j = 0; j < 10; j++){ //ITERACION LA CUAL NOS AYUDA A PASAR POR TODOS
          LOS SOLDADOS AL ARRAY SOLDADO PARA PODER USAR EL USO DEL METODO DE
          ORDENACION INSERCION
         if(army.get(i).get(j) != null){
            if(count - count + x == count){
               break;
            }else{
               soldados[count - count + x] = army.get(i).get(j); //LA MISMA LOGICA QUE
                   EL ANTERIOR METODO SOLO QUE EN ESTE LO USARIAMOS DE MANERA
                   DIFERENTE YA QUE ESTE SERIA DE FORMA DE INSERCION
            }
            x++;
         }
      }
   }
   int n = soldados.length;
   for (int i = 1; i < n; i++) {
      Soldado actual = soldados[i];
      int j = i - 1;
      while (j >= 0 && soldados[j].getLifeActual() < actual.getLifeActual()) {
          //ORDENAMOS EL EJERCITO RESPECTIVAMENTE MEDIANTE EL METODO QUE NOS OFRECE
          INSERCION EL CUAL ES ESTE CODIGO
         soldados[j + 1] = soldados[j];
         j--;
      }
      soldados[j + 1] = actual;
   }
   for(int i = 0; i < soldados.length; i++){
      System.out.print("\n" + "Puesto " + (i + 1));
      System.out.println(soldados[i].toString()); //PUBLICAMOS RESULTADOS
      System.out.println("------------------");
   }
   System.out.println("**********************************");
}
public void setArmy(ArrayList<ArrayList<Soldado>> army){
   this.army = army;
}
```

```
    public void setKingdom(String kingdom){
       this.kingdom = kingdom;
    }
    public String getKingdom(){
       return kingdom;
    }
}
```

Listing 9: Las lineas de codigos de la clase Mapa creada:

```
import java.text.DecimalFormat;
import java.util.*;

public class Mapa {
   Scanner sc = new Scanner(System.in);
   Random rdm = new Random();
   private String territory;
   private ArrayList<ArrayList<Soldado>> board;
   private Ejercito army1e;
   private Ejercito army2e;
   private String[] typesterritory = {"bosque", "campo abierto", "montana", "desierto",
       "playa"};
   private String[] kingdoms = {"Inglaterra", "Francia", "Sacro", "Castilla", "Aragon",
       "Moros"};
   public Mapa(){
      this.board = fillboard();
   }
   public void iniciarJuego() {
      menuBatalla();
      int n = sc.nextInt();
      while (n == 1) {
         String kingdom1 = kingdoms[rdm.nextInt(6)];
         String kingdom2 = kingdoms[rdm.nextInt(6)];
         ArrayList<ArrayList<Soldado>> army1 = fillarray(kingdom1, 1);
         ArrayList<ArrayList<Soldado>> army2 = fillarray(kingdom2, 2);
         army1e = new Ejercito(army1,kingdom1);
         army2e = new Ejercito(army2,kingdom2);
         territory = typesterritory[rdm.nextInt(5)];
         System.out.println("\n**********************************");
         System.out.println("El tipo de territorio es: " + territory);
         System.out.println("\n**********************************");
         bonificacion(army1, territory, kingdom1);
         bonificacion(army2, territory, kingdom2);
         army1e.viewSoldiers(kingdom1, 1, army1);
         army2e.viewSoldiers(kingdom2, 2, army2);
         viewBoard(army1, army2);
         army1e.longerLife(army1, kingdom1);
         army2e.longerLife(army2, kingdom2);
         double avg1 = army1e.averageLife(army1, kingdom1);
         double avg2 = army2e.averageLife(army2, kingdom2);
         army1e.rankingInsercionLife(army1, kingdom1);
         army2e.rankingInsercionLife(army2, kingdom2);
         viewBoard(army1, army2);
         resultBattleInfo(army1, kingdom1 , 1);
         resultBattleInfo(army2, kingdom2 , 2);
         int sum1 = resultBattleSum(army1, kingdom1, 1);
```

```
            int sum2 = resultBattleSum(army2, kingdom2, 2);
            double sumtotal = (sum1 * 1.0) + (sum2 * 1.0);
            resbattle(sumtotal, sum1, sum2, 1, 2, kingdom1, kingdom2);
            volveraJugar();
            n = sc.nextInt();
        }
    }
    public static ArrayList<ArrayList<Soldado>> fillboard(){
        ArrayList<ArrayList<Soldado>> army = new ArrayList<ArrayList<Soldado>>();
        for(int i = 0; i < 10; i++){ //ITERACION
            army.add(new ArrayList<Soldado>()); //LLENAMOS NUESTROS ARRAYLIST BIDIMENSIONAL
                CON CADA FILA PARA QUE CUMPLAN CON ESTRUCTURA DEL TABLERO
            for(int j = 0; j < 10 ; j++){//ITERACION
                army.get(i).add(null); // LLENAMOS CADA FILA DEL ARRAYLIST CON UN OBJETO
                    SOLDADO CON TAL QUE ESTE SEA NULL PARA QUE SEPA QUE ESTE TIENE UNA
                    CASILLA PERO NO HAY NADIE TODAVIA SE PUEDE LLENAR
            }
        }
        return army;
    }
    public static ArrayList<ArrayList<Soldado>> fillarray(String armyespe, int num){
        Random rdm = new Random();
        ArrayList<ArrayList<Soldado>> army = new ArrayList<ArrayList<Soldado>>();
        int numbersoldiers = rdm.nextInt(10) + 1; //NUMERO DE SOLDADOS ALEATORIOS ENTRE 1
            A 10 SOLDADOS
        for(int i = 0; i < 10; i++){ //ITERACION
            army.add(new ArrayList<Soldado>()); //LLENAMOS NUESTROS ARRAYLIST BIDIMENSIONAL
                CON CADA FILA PARA QUE CUMPLAN CON ESTRUCTURA DEL TABLERO
            for(int j = 0; j < 10 ; j++){//ITERACION
                army.get(i).add(null); // LLENAMOS CADA FILA DEL ARRAYLIST CON UN OBJETO
                    SOLDADO CON TAL QUE ESTE SEA NULL PARA QUE SEPA QUE ESTE TIENE UNA
                    CASILLA PERO NO HAY NADIE TODAVIA SE PUEDE LLENAR
            }
        }
        System.out.println("El Ejercito " + armyespe + " tiene " + numbersoldiers + "
            soldados : " );
        System.out.println("");
        for(int i = 0; i < numbersoldiers; i++){ //LLENAMOS CASILLAS CON CADA SOLDADO
            CREADO ALEATORIAMENTE
            Soldado soldado = getRandomSoldado();
            String name = "Soldado" + i + "X" + num;
            //System.out.println(name); PRUEBA QUE SE HIZO PARA VER LOS NOMBRES
            int health = rdm.nextInt(5) + 1;
            int row = rdm.nextInt(10) + 1;
            int speed = rdm.nextInt(5) + 1;
            String column = String.valueOf((char)(rdm.nextInt(10) + 65)); //REUTILIZAMOS
                CODIGO DEL ANTERIOR ARCHIVO VIDEOJUEGO2.JAVA YA QUE TENDRIAN LA MISMA
                FUNCIONALIDAD
            //System.out.println(army.get(row - 1).get((int)column.charAt(0) - 65)); PRUEBA
                QUE SE HIZO PARA COMPROBAR SI EL OBJETO SE ESTABA DANDO O NO CAPAZ NI
                EXISTIA
            int lifelevel, defenselevel = 0;
            int attacklevel = 0;
            if (soldado instanceof Espadachin) {
                name = "Espadachin" + i + "X" + num;
                lifelevel = rdm.nextInt(3) + 8;
```

```
                attacklevel = 10;
                defenselevel = 8;
                soldado.setName(name);
                soldado.setAttackLevel(attacklevel);
                soldado.setDefenseLevel(defenselevel);
                soldado.setLifeLevel(lifelevel);
                soldado.setRow(row);
                soldado.setColumn(column);
                if(army.get(row - 1).get((int)column.charAt(0) - 65) == null){
                    System.out.println("Registrando al " + (i + 1) + " soldado del Ejercito "
                        + armyespe + "");
                    army.get(row - 1).set((int)column.charAt(0) - 65, new Espadachin(name,
                        attacklevel, defenselevel, lifelevel, speed, "Espadachin", true, row,
                        column, attacklevel));
                    army.get(row - 1).get((int)column.charAt(0) - 65).setSpeed(speed);
                    System.out.println(army.get(row - 1).get((int)column.charAt(0) -
                        65).toString());
                    System.out.println("----------------------------------");
                }else{
                    i -= 1; //NOS AYUDARIA CON LOS SOLDADOS QUE SE REPITEN EN EL MISMO
                        CASILLERO CON TAL QUE NO DEBERIA CONTAR
                }
            } else if (soldado instanceof Arquero) {
                name = "Arquero" + i + "X" + num;
                attacklevel = 7;
                defenselevel = 3;
                lifelevel = rdm.nextInt(3) + 3;
                soldado.setName(name);
                soldado.setAttackLevel(attacklevel);
                soldado.setDefenseLevel(defenselevel);
                soldado.setLifeLevel(lifelevel);
                soldado.setRow(row);
                soldado.setColumn(column);
                if(army.get(row - 1).get((int)column.charAt(0) - 65) == null){
                    System.out.println("Registrando al " + (i + 1) + " soldado del Ejercito "
                        + armyespe + "");
                    army.get(row - 1).set((int)column.charAt(0) - 65, new Arquero(name,
                        attacklevel, defenselevel, lifelevel, speed, "Arquero", true, row,
                        column, attacklevel));
                    army.get(row - 1).get((int)column.charAt(0) - 65).setSpeed(speed);
                    System.out.println(army.get(row - 1).get((int)column.charAt(0) -
                        65).toString());
                    System.out.println("----------------------------------");
                }else{
                    i -= 1; //NOS AYUDARIA CON LOS SOLDADOS QUE SE REPITEN EN EL MISMO
                        CASILLERO CON TAL QUE NO DEBERIA CONTAR
                }
            } else if (soldado instanceof Caballero) {
                name = "Caballero" + i + "X" + num;
                attacklevel = 13;
                defenselevel = 7;
                lifelevel = rdm.nextInt(3) + 10;
                soldado.setName(name);
                soldado.setAttackLevel(attacklevel);
                soldado.setDefenseLevel(defenselevel);
                soldado.setLifeLevel(lifelevel);
```

```
                soldado.setRow(row);
                soldado.setColumn(column);
                if(army.get(row - 1).get((int)column.charAt(0) - 65) == null){
                   System.out.println("Registrando al " + (i + 1) + " soldado del Ejercito "
                       + armyespe + "");
                   army.get(row - 1).set((int)column.charAt(0) - 65, new Caballero(name,
                       attacklevel, defenselevel, lifelevel, speed, "Caballero", true, row,
                       column, false));
                   army.get(row - 1).get((int)column.charAt(0) - 65).setSpeed(speed);
                   System.out.println(army.get(row - 1).get((int)column.charAt(0) -
                       65).toString());
                   System.out.println("---------------------------------");
                }else{
                   i -= 1; //NOS AYUDARIA CON LOS SOLDADOS QUE SE REPITEN EN EL MISMO
                       CASILLERO CON TAL QUE NO DEBERIA CONTAR
                }
             } else if (soldado instanceof Lancero) {
                name = "Lancero" + i + "X" + num;
                attacklevel = 5;
                defenselevel = 10;
                lifelevel = rdm.nextInt(3) + 5;
                soldado.setName(name);
                soldado.setAttackLevel(attacklevel);
                soldado.setDefenseLevel(defenselevel);
                soldado.setLifeLevel(lifelevel);
                soldado.setRow(row);
                soldado.setColumn(column);
                if(army.get(row - 1).get((int)column.charAt(0) - 65) == null){
                   System.out.println("Registrando al " + (i + 1) + " soldado del Ejercito "
                       + armyespe + "");
                   army.get(row - 1).set((int)column.charAt(0) - 65, new Lancero(name,
                       attacklevel, defenselevel, lifelevel, speed, "Lancero", true, row,
                       column, attacklevel));
                   army.get(row - 1).get((int)column.charAt(0) - 65).setSpeed(speed);
                   System.out.println(army.get(row - 1).get((int)column.charAt(0) -
                       65).toString());
                   System.out.println("---------------------------------");
                }else{
                   i -= 1; //NOS AYUDARIA CON LOS SOLDADOS QUE SE REPITEN EN EL MISMO
                       CASILLERO CON TAL QUE NO DEBERIA CONTAR
                }
             }
          }
          System.out.println("**********************************");
          return army;
       }
       public static void menuBatalla(){
          System.out.println("-----------------------------------------");
          System.out.println("--             MENU                --");
          System.out.println("-----------------------------------------");
          System.out.println(" SELECCIONE UN NUMERO PARA PODER EMPEZAR O TERMINAR");
          System.out.println(" 1 : JUGAR");
          System.out.println(" 2 : NO JUGAR");
       }
       public static void viewBoard(ArrayList<ArrayList<Soldado>> army1,
           ArrayList<ArrayList<Soldado>> army2){ //EN ESTE METODO DEMOSTRAREMOS LA TABLA
```

```
        REUTILIZAREMOS CODIGOS DE ANTERIORES LABORATORIOS PARA PODER HACER LA BASE DE
        ESTE TABLERO
      System.out.println("\nMostrando tabla de posicion ... --");
      System.out.println("Leyenda: Ejercito1 --> 1#1 | Ejercito2 --> 2#2");
          //RECONOCIMIENTO PARA LOS EJERCITOS Y POSICION DE SUS SOLDADOS
      System.out.println("\n \t A\t  B\t  C\t  D\t  E\t  F\t  G\t  H\t  I\t  J"); //
          RECONOCIMIENTO PARA CADA UBICACION DE CADA SOLDADO EN EL TABLERO POR PARTE DE
          LAS COLUMNAS
      System.out.println("\t_______________________________________________________________________
      for(int i = 0; i < 10; i++ ){
         System.out.print((i + 1) + "\t"); // RECONOCIMIENTO PARA CADA UBICACION DE CADA
             SOLDADO EN EL TABLERO POR PARTE DE LAS FILAS
           for(int j = 0; j < 10; j++){
                if(army1.get(i).get(j) != null && army2.get(i).get(j) != null){
                    //CREAMOS UN IF PARA QUE ESTE NOS AYUDE A SABER QUIEN DE ESTOS
                    SOLDADOS SE OCUPARA DEL CASILLERO EL CUAL DONDE ESTAN PELEANDO
                  if(army1.get(i).get(j).getLifeActual() >
                      army2.get(i).get(j).getLifeActual()){
                     army1.get(i).get(j).setLifeActual(army1.get(i).get(j).getLifeActual()
                         - army2.get(i).get(j).getLifeActual()); //Cambiamos
                     army2.get(i).set(j, null);
                     System.out.print("| 1" + obtenerInicial(army1.get(i).get(j)) + "1
                         ");
                  }else if(army2.get(i).get(j).getLifeActual() >
                      army1.get(i).get(j).getLifeActual()){
                     army2.get(i).get(j).setLifeActual(army2.get(i).get(j).getLifeActual()
                         - army1.get(i).get(j).getLifeActual());
                     army1.get(i).set(j, null);;
                     System.out.print("| 2" + obtenerInicial(army2.get(i).get(j)) + "2
                         ");
                  }else{
                     army2.get(i).set(j, null);
                     army1.get(i).set(j, null);
                     System.out.print("| " + "" + "  ");
                  }
                }else if(army1.get(i).get(j) != null){
                     System.out.print("| 1" + obtenerInicial(army1.get(i).get(j)) + "1 ");
                }else if(army2.get(i).get(j) != null){
                     System.out.print("| 2" + obtenerInicial(army2.get(i).get(j)) + "2 ");
                }else{
                     System.out.print("| " + " " + "  ");
                }
           }
           System.out.println("|");
           System.out.println("\t|_______|_______|_______|_______|_______|_______|_______|_______|_
      }
      System.out.println("\n*********************************");
   }
   public static Soldado getRandomSoldado() {
      Random rdm = new Random();
      int tipoSoldado = rdm.nextInt(4);
      switch (tipoSoldado) {
         case 0:
            return new Espadachin();
         case 1:
            return new Arquero();
```

```
            case 2:
               return new Lancero();
            case 3:
               return new Caballero();
            default:
               return new Espadachin();
         }
      }
      public static String obtenerInicial(Soldado soldado) {
         if (soldado instanceof Espadachin) {
            return "E";
         } else if (soldado instanceof Arquero) {
            return "A";
         } else if (soldado instanceof Caballero) {
            return "C";
         } else if (soldado instanceof Lancero) {
            return "L";
         } else {
            return "S";
         }
      }
      public void bonificacion(ArrayList<ArrayList<Soldado>> army, String territory ,
          String kingdom) {
         for(int i = 0; i < 10; i++){ //ITERACION
            for(int j = 0; j < 10 ; j++){//ITERACION
               if(army.get(i).get(j) != null){
                  if(kingdom.equals("Inglaterra") && territory.equals("bosque")){
                     army.get(i).get(j).setLifeLevel(army.get(i).get(j).getLifeLevel() + 1);
                  }else if(kingdom.equals("Francia") && territory.equals("campo abierto")){
                     army.get(i).get(j).setLifeLevel(army.get(i).get(j).getLifeLevel() + 1);
                  }else if((kingdom.equals("Castilla") || kingdom.equals("Aragon")) &&
                       territory.equals("montana")){
                     army.get(i).get(j).setLifeLevel(army.get(i).get(j).getLifeLevel() + 1);
                  }else if(kingdom.equals("Moros") && territory.equals("desierto")){
                     army.get(i).get(j).setLifeLevel(army.get(i).get(j).getLifeLevel() + 1);
                  }else if(kingdom.equals("Sacro") && (territory.equals("desierto") ||
                       territory.equals("playa") || territory.equals("campo abierto"))){
                     army.get(i).get(j).setLifeLevel(army.get(i).get(j).getLifeLevel() + 1);
                  }
               }
            }
         }
      }
      public static void resultBattleInfo(ArrayList<ArrayList<Soldado>> army, String
          kingdom, int n){
         System.out.println("Ejercito " + n + " : " + kingdom);
         int numbersoldiers = 0;
         int numberespadachines = 0;
         int numbercaballeros = 0;
         int numberlanceros = 0;
         int numberarqueros = 0;
         for(int i = 0; i < 10; i++){ //ITERACION
            for(int j = 0; j < 10 ; j++){//ITERACION
               if(army.get(i).get(j) != null){
                  numbersoldiers++;
                  if(army.get(i).get(j) instanceof Espadachin){
```

```
                    numberespadachines++;
                }else if(army.get(i).get(j) instanceof Caballero){
                    numbercaballeros++;
                }else if(army.get(i).get(j) instanceof Lancero){
                    numberlanceros++;
                }else if(army.get(i).get(j) instanceof Arquero){
                    numberarqueros++;
                }
             }
          }
       }
       System.out.println("Cantidad total de soldados creados: " + numbersoldiers);
       System.out.println("Espadachines: " + numberespadachines);
       System.out.println("Arqueros: " + numberarqueros);
       System.out.println("Caballeros: " + numbercaballeros);
       System.out.println("Lanceros: " + numberlanceros + "\n");
    }
    public static int resultBattleSum(ArrayList<ArrayList<Soldado>> army, String
        kingdom, int n){
       int sum = 0;
       for(int i = 0; i < 10; i++){ //ITERACION
          for(int j = 0; j < 10 ; j++){//ITERACION
             if(army.get(i).get(j) != null){
                sum += army.get(i).get(j).getLifeLevel();
             }
          }
       }
       System.out.println("Ejercito " + n + ": " + kingdom + ": " + sum);
       return sum;
    }
    public static void resbattle(double sumtotal , int sum1, int sum2 , int n1, int n2 ,
        String kingdom1, String kingdom2){
       DecimalFormat df = new DecimalFormat("#.##");
       String numero1 = df.format(((sum1 / sumtotal) * 100));
       String numero2 = df.format(((sum2 / sumtotal) * 100));
       if(sum1 > sum2){
          System.out.println("El ganador es el ejercito " + n1 + " de: " + kingdom1 +".
              Ya que al generar los porcentajes de probabilidad de victoria basada en los
              niveles de vida de sus soldados y aplicando un experimento aleatorio salio
              vencedor. (Aleatorio generado : "+ numero1 + ")");
       }else if (sum2 > sum1){
          System.out.println("El ganador es el ejercito " + n2 + " de: " + kingdom2 +".
              Ya que al generar los porcentajes de probabilidad de victoria basada en los
              niveles de vida de sus soldados y aplicando un experimento aleatorio salio
              vencedor. (Aleatorio generado : "+ numero2 + ")");
       }else{
          System.out.println("El resultado de la batalla es Empate");
       }
    }
    public static void volveraJugar(){
       System.out.println("\n*********************************");
       System.out.println(" DESEA VOLVER A JUGAR");
       System.out.println(" 1 : JUGAR");
       System.out.println(" 2 : NO JUGAR");
    }
 }
```

4.3. Estructura de laboratorio 22

- El contenido que se entrega en este laboratorio22 es el siguiente:

```
\Lab22
```

5. Rúbricas

5.1. Entregable Informe

Tabla 1: Tipo de Informe

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

5.2. Rúbrica para el contenido del Informe y demostración

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

Tabla 2: Niveles de desempeño

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

Tabla 3: Rúbrica para contenido del Informe y demostración

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

6. Referencias

- https://drive.google.com/drive/folders/1_yO46UOaxs7uKVK7nrrkcNwybnk_ZJXJ