# Informe de Laboratorio 22
## Tema: Laboratorio 22

| Nota |
|---|
| |

| Estudiante | Escuela | Asignatura |
|---|---|---|
| Victor Mamani Anahua<br>vmamanian@unsa.edu.pe | Escuela Profesional de Ingeniería de Sistemas | Fundamentos de la Programación II<br>Semestre: II<br>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. 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