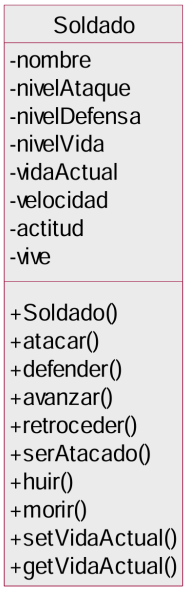
Universidad Nacional de San Agustín
Escuela Profesional de Ingeniería de Sistemas
Fundamentos de Programación II
Práctica de Laboratorio N° 9:
Definición de Clases de Usuario
Clase Soldado

Nombre: Jhonatan Benjamin Mamani Céspedes **CUI: 20232188**

Link de GitHub: https://github.com/JBenjamin01/fp2-24b/tree/main/Laboratorio

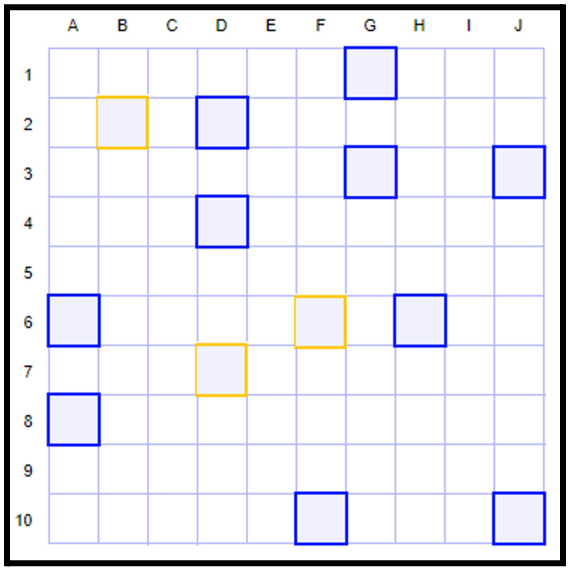
Usar como base el diagrama de clases UML siguiente:



1. Crear 3 constructores sobrecargados.
2. La actitud puede ser defensiva, ofensiva, fuga. Dicha actitud varía cuando el soldado defiende, ataca o huye respectivamente.
3. Al atacar el soldado avanza, al avanzar aumenta su velocidad en 1. Al defender el soldado se para. Al huir aumenta su velocidad en 2. Al retroceder, si su velocidad es mayor que 0, entonces primero para y su actitud es defensiva, y si su velocidad es 0 entonces disminuirá a valores negativos. Al ser atacado su vida actual disminuye y puede llegar incluso a morir.
4. Crear los atributos y métodos extra que considere necesarios.
5. Tendrá 2 Ejércitos. Usar la estructura de datos que considere más adecuada. Inicializar el tablero con n soldados aleatorios entre 1 y 10 para cada Ejército. Cada soldado tendrá un nombre autogenerado: Soldado0X1, Soldado1X1, etc., un valor de puntos de vida autogenerado aleatoriamente [1..5], la fila y columna también autogenerados aleatoriamente (no puede haber 2 soldados en el mismo cuadrado). Nivel de ataque y de defensa son aleatorios [1..5]. Se debe mostrar el tablero con todos los soldados creados (usar caracteres como | _ y otros) y distinguir los de un ejército de los del otro ejército.

Además de los datos del Soldado con mayor vida de cada ejército, el promedio de puntos de vida de todos los soldados creados por ejército, los datos de todos los soldados por ejército en el orden que fueron creados y un ranking de poder de todos los soldados creados por ejército (del que tiene más nivel de vida al que tiene menos) usando 2 diferentes algoritmos de ordenamiento. Finalmente, que muestre qué ejército ganará la batalla (indicar la métrica usada para decidir al ganador de la batalla). Hacerlo un programa iterativo.

6. Crear el diagrama de clases UML completo



Clase Soldado.java:

```java
public class Soldado {
    private String nombre;
    private int puntosVida;
    private int fila;
    private char columna;
    private int nivelAtaque;
    private int nivelDefensa;
    private int velocidad;
    private String actitud;
    private boolean vive;

    // Constructores sobrecargados del Soldado
    public Soldado(String nombre, int puntosVida, int fila, char columna, int nivelAtaque, int nivelDefensa) {
        this.nombre = nombre;
        this.puntosVida = puntosVida;
        this.fila = fila;
        this.columna = columna;
        this.nivelAtaque = nivelAtaque;
        this.nivelDefensa = nivelDefensa;
        this.velocidad = 0;
        this.actitud = "Defensiva";  // Estado inicial
        this.vive = true; // Aqui dejo que por defecto, el soldado está vivo
    }
    public Soldado(String nombre, int puntosVida, int fila, char columna) {
        this.nombre = nombre;
        this.puntosVida = puntosVida;
        this.fila = fila;
        this.columna = columna;
        this.nivelAtaque = 0;
        this.nivelDefensa = 0;
        this.velocidad = 0;
        this.actitud = "Defensiva";
        this.vive = true;
    }
```

```java
    public Soldado(String nombre) {
        this(nombre, (int)(Math.random() * 5) + 1, (int)(Math.random() * 10), (char)('A' + (int)(Math.random() * 10)),
            (int)(Math.random() * 5) + 1, (int)(Math.random() * 5) + 1);
    }
    public Soldado() {
        this("Soldado" + (int)(Math.random() * 100));
    }

    // Getters
    public int getPuntosVida() {
        return puntosVida;
    }
    public int getNivelAtaque() {
        return nivelAtaque;
    }
    public int getNivelDefensa() {
        return nivelDefensa;
    }
    public int getVelocidad() {
        return velocidad;
    }
    public String getActitud() {
        return actitud;
    }
    public boolean isVivo() {
        return vive;
    }

    // Métodos del UML de referencia
    public void atacar() {
        velocidad += 1;
        actitud = "Ofensiva";
        System.out.println(nombre + " ha atacado, su velocidad es ahora " + velocidad);
    }
    public void defender() {
        actitud = "Defensiva";
        System.out.println(nombre + " está en modo defensivo.");
    }
    public void huir() {
        velocidad += 2;
        actitud = "Fuga";
        System.out.println(nombre + " está huyendo, su velocidad es ahora " + velocidad);
    }
    public void avanzar() {
        velocidad += 1;
        System.out.println(nombre + " avanza, su velocidad es ahora " + velocidad);
    }
```

```java
    public void retroceder() {
        if (velocidad > 0) {
            velocidad = 0;
            actitud = "Defensiva";
            System.out.println(nombre + " se ha detenido, velocidad actual: " + velocidad);
        } else {
            velocidad -= 1;
            System.out.println(nombre + " ha retrocedido, velocidad negativa: " + velocidad);
        }
    }
    public void serAtacado(int daño) {
        recibirAtaque(daño);
    }
    public void recibirAtaque(int daño) {
        puntosVida -= daño;
        if (puntosVida <= 0) {
            puntosVida = 0;
            System.out.println(nombre + " ha muerto.");
        } else {
            System.out.println(nombre + " ha recibido " + daño + " de daño. Vida restante: " + puntosVida);
        }
    }
    public void morir() {
        vive = false;
        System.out.println(nombre + " ha muerto.");
    }
```

```java

    @Override
    public String toString() {
        return "Nombre: " + nombre
            + " | Vida: " + puntosVida
            + " | Ataque: " + nivelAtaque
            + " | Defensa: " + nivelDefensa
            + " | Velocidad: " + velocidad
            + " | Actitud: " + actitud
            + " | Posición: " + fila + columna
            + " | Vive: " + (vive ? "Si" : "No");
    }
}
```

Clase Videojuego.java:

```java
// LABORATORIO N° 9
// AUTOR: JHONATAN BENJAMIN MAMANI CÉSPEDES
// TIEMPO: 79 MINUTOS
import java.util.*;
public class VideoJuego {
    public static void main(String[] args) {
        ArrayList<ArrayList<Soldado>> tablero = new ArrayList<>();
        ArrayList<Soldado> e1 = new ArrayList<>();
        ArrayList<Soldado> e2 = new ArrayList<>();

        inicializarTablero(tablero);
        inicializarEjercitos(e1, e2, tablero);

        mostrarTablero(tablero, e1, e2);

        System.out.println("\nDatos del ejército 1:");
        mostrarDatosEjercito(e1);
        System.out.println("\nDatos del ejército 2:");
        mostrarDatosEjercito(e2);

        Soldado soldadoMayorVidaE1 = obtenerSoldadoMayorVida(e1);
        Soldado soldadoMayorVidaE2 = obtenerSoldadoMayorVida(e2);

        System.out.println("\nSoldado con mayor vida del ejército 1:\n" + soldadoMayorVidaE1);
        System.out.println("\nSoldado con mayor vida del ejército 2:\n" + soldadoMayorVidaE2);

        double promedioVidaE1 = calcularPromedioVida(e1);
        double promedioVidaE2 = calcularPromedioVida(e2);

        System.out.println("\nPromedio de puntos de vida del ejército 1: " + promedioVidaE1);
        System.out.println("Promedio de puntos de vida del ejército 2: " + promedioVidaE2);

        ordenamientoInsertionSort(e1);
        System.out.println("\nRanking de soldados del ejército 1 (ordenado por puntos de vida de forma decreciente):");
        mostrarDatosEjercito(e1);


        ordenamientoBubbleSort(e2);
        System.out.println("\nRanking de soldados del ejército 2 (ordenado por puntos de vida de forma decreciente):");
        mostrarDatosEjercito(e2);


        String resultadoBatalla = determinarGanador(e1, e2);
        System.out.println("\nResultado de la batalla:");
        System.out.println(resultadoBatalla);
    }
```

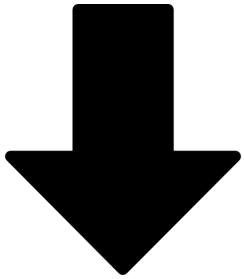
```java

    public static void inicializarTablero(ArrayList<ArrayList<Soldado>> tablero) {
        for (int i = 0; i < 10; i++) {
            ArrayList<Soldado> fila = new ArrayList<>();
            for (int j = 0; j < 10; j++)
                fila.add(null);
            tablero.add(fila);
        }
    }
```

```java
    public static void inicializarEjercitos(ArrayList<Soldado> e1, ArrayList<Soldado> e2,
            ArrayList<ArrayList<Soldado>> tablero) {
        for (int i = 0; i < 2; i++) {
            int n = (int) (Math.random() * 10) + 1;
            for (int j = 0; j < n; j++) {
                int fila, columna;
                do {
                    fila = (int) (Math.random() * 10);
                    columna = (int) (Math.random() * 10);
                } while (tablero.get(fila).get(columna) != null);

                String nombre = "Soldado" + (i + 1) + "X" + (j + 1);
                int puntosVida = (int) (Math.random() * 5) + 1;
                Soldado soldado = new Soldado(nombre, puntosVida, fila + 1, (char) ('A' + columna));

                if (i == 0) {
                    e1.add(soldado);
                    tablero.get(fila).set(columna, soldado);
                } else {
                    e2.add(soldado);
                    tablero.get(fila).set(columna, soldado);
                }
            }
        }
    }

    public static void mostrarTablero(ArrayList<ArrayList<Soldado>> tablero, ArrayList<Soldado> e1,
                                      ArrayList<Soldado> e2) {
        System.out.println("Tablero de la batalla:"
                           + "\nLas unidades del ejército 1 estarán con sus puntos de vida entre corchetes ([x])."
                           + "\nLas del ejército 2 con sus puntos de vida entre signos menor y mayor que (<x>):");
        System.out.println("\n           A     B     C     D     E     F     G     H     I     J");
        System.out.println();
        System.out.println("        ------------------------------------------------------------");

        for (int i = 0; i < tablero.size(); i++) {
            System.out.print(i + 1 + "\t| ");
            for (int j = 0; j < tablero.get(i).size(); j++) {
                Soldado soldado = tablero.get(i).get(j);
                if (soldado == null)
                    System.out.print("    | ");
                else
                    if (e1.contains(soldado))
                        System.out.print("[" + soldado.getPuntosVida() + "] | ");
                    else if (e2.contains(soldado))
                        System.out.print("<" + soldado.getPuntosVida() + "> | ");
                    else
                        System.out.print(soldado.getPuntosVida() + " | ");
```

```java

            }
            System.out.println();
            System.out.println("        ------------------------------------------------------------");
        }
    }

    public static void mostrarDatosEjercito(ArrayList<Soldado> ejercito) {
        for (Soldado s : ejercito)
            System.out.println(s);
    }

    public static Soldado obtenerSoldadoMayorVida(ArrayList<Soldado> ejercito) {
        Soldado mayorVida = null;
        int maxPuntosVida = Integer.MIN_VALUE;
        for (Soldado soldado : ejercito)
            if (soldado.getPuntosVida() > maxPuntosVida) {
                maxPuntosVida = soldado.getPuntosVida();
                mayorVida = soldado;
            }
        return mayorVida;
    }

    public static double calcularPromedioVida(ArrayList<Soldado> ejercito) {
        double total = 0;
        for (Soldado soldado : ejercito)
            total += soldado.getPuntosVida();
        return total / ejercito.size();
    }

```

```java
    public static void ordenamientoInsertionSort(ArrayList<Soldado> ejercito) {
        int n = ejercito.size();
        for (int i = 1; i < n; ++i) {
            Soldado key = ejercito.get(i);
            int j = i - 1;

            while (j >= 0 && ejercito.get(j).getPuntosVida() < key.getPuntosVida()) {
                ejercito.set(j + 1, ejercito.get(j));
                j = j - 1;
            }
            ejercito.set(j + 1, key);
        }
    }
    public static void ordenamientoBubbleSort(ArrayList<Soldado> ejercito) {
        int n = ejercito.size();
        for (int i = 0; i < n - 1; i++)
            for (int j = 0; j < n - i - 1; j++)
                if (ejercito.get(j).getPuntosVida() < ejercito.get(j + 1).getPuntosVida()){
                    Soldado temp = ejercito.get(j);
                    ejercito.set(j, ejercito.get(j + 1));
                    ejercito.set(j + 1, temp);
                }
    }

    public static String determinarGanador(ArrayList<Soldado> e1, ArrayList<Soldado> e2) {
        int puntosE1 = 0;
        int puntosE2 = 0;

        for (Soldado soldado : e1)
            puntosE1 += soldado.getPuntosVida();

        for (Soldado soldado : e2)
            puntosE2 += soldado.getPuntosVida();

        if (puntosE1 > puntosE2)
            return "El ejército 1 ha ganado la batalla. La suma total de sus puntos de vida es " + puntosE1
                    + " superando en " + (puntosE1 - puntosE2) + " puntos al ejército 2.";
        else if (puntosE2 > puntosE1)
            return "El ejército 2 ha ganado la batalla. La suma total de sus puntos de vida es " + puntosE2
                    + " superando en " + (puntosE2 - puntosE1) + " puntos al ejército 1.";
        else
            return "La batalla ha terminado en empate. Ambos ejércitos tienen " + puntosE1 + " puntos de vida en total.";
    }
}
```

Los cambios más importantes hechos en este laboratorio son principalmente adaptar los ejércitos a su forma de ArrayList como en el laboratorio 7 para volver a tener los ordenamientos más eficientes, por otro lado, agregué los nuevos atributos y métodos a la clase Soldado adaptando sus valores por defecto desde los constructores y su posterior muestra desde el toString que se usa también en el mismo videojuego, así se ve a continuación desde la consola:



Consola:

```
PS C:\Users\jhona\OneDrive\Documentos\University\Universidad Nacional de San Agustín\2nd Year\Segundo Semestre\Fundamen
ersity\Universidad Nacional de San Agustín\2nd Year\Segundo Semestre\Fundamentos de la Programación 2 - A\Laboratorio\P
-cp' 'C:\Users\jhona\AppData\Roaming\Code\User\workspaceStorage\2b402101b08498c28ad9cea985b5fbbd\redhat.java\jdt_ws\Prá
Tablero de la batalla:
Las unidades del ejército 1 estarán con sus puntos de vida entre corchetes ([x]).
Las del ejército 2 con sus puntos de vida entre signos menor y mayor que (<x>):

          A     B     C     D     E     F     G     H     I     J

        ---------------------------------------------------------
1       |     |     |     |     |     | [1] |     |     |     |     |
        ---------------------------------------------------------
2       |     |     | [5] |     |     | [1] |     |     |     |     |
        ---------------------------------------------------------
3       |     |     |     | [3] |     |     |     |     |     |     |
        ---------------------------------------------------------
4       |     |     |     |     |     | [1] |     |     |     |     |
        ---------------------------------------------------------
5       |     |     |     | <1> |     |     |     |     |     |     |
        ---------------------------------------------------------
6       |     |     |     |     |     |     |     |     |     |     |
        ---------------------------------------------------------
7       |     |     |     |     |     |     |     |     |     | [1] |
        ---------------------------------------------------------
8       |     |     |     | <3> | [4] |     |     |     |     |     |
        ---------------------------------------------------------
9       |     |     |     |     |     |     |     |     |     |     |
        ---------------------------------------------------------
10      |     |     |     | <5> |     |     |     |     |     |     |
        ---------------------------------------------------------

Datos del ejército 1:
Nombre: Soldado1X1 | Vida: 5 | Ataque: 0 | Defensa: 0 | Velocidad: 0 | Actitud: Defensiva | Posición: 2C | Vive: Si
Nombre: Soldado1X2 | Vida: 1 | Ataque: 0 | Defensa: 0 | Velocidad: 0 | Actitud: Defensiva | Posición: 1F | Vive: Si
Nombre: Soldado1X3 | Vida: 1 | Ataque: 0 | Defensa: 0 | Velocidad: 0 | Actitud: Defensiva | Posición: 4F | Vive: Si
Nombre: Soldado1X4 | Vida: 1 | Ataque: 0 | Defensa: 0 | Velocidad: 0 | Actitud: Defensiva | Posición: 7J | Vive: Si
Nombre: Soldado1X5 | Vida: 3 | Ataque: 0 | Defensa: 0 | Velocidad: 0 | Actitud: Defensiva | Posición: 3D | Vive: Si
Nombre: Soldado1X6 | Vida: 4 | Ataque: 0 | Defensa: 0 | Velocidad: 0 | Actitud: Defensiva | Posición: 8E | Vive: Si
Nombre: Soldado1X7 | Vida: 1 | Ataque: 0 | Defensa: 0 | Velocidad: 0 | Actitud: Defensiva | Posición: 2F | Vive: Si
```

```
Nombre: Soldado1X1 | Vida: 5 | Ataque: 0 | Defensa: 0 | Velocidad: 0 | Actitud: Defensiva | Posición: 2C | Vive: Si
Nombre: Soldado1X2 | Vida: 1 | Ataque: 0 | Defensa: 0 | Velocidad: 0 | Actitud: Defensiva | Posición: 1F | Vive: Si
Nombre: Soldado1X3 | Vida: 1 | Ataque: 0 | Defensa: 0 | Velocidad: 0 | Actitud: Defensiva | Posición: 4F | Vive: Si
Nombre: Soldado1X4 | Vida: 1 | Ataque: 0 | Defensa: 0 | Velocidad: 0 | Actitud: Defensiva | Posición: 7J | Vive: Si
Nombre: Soldado1X5 | Vida: 3 | Ataque: 0 | Defensa: 0 | Velocidad: 0 | Actitud: Defensiva | Posición: 3D | Vive: Si
Nombre: Soldado1X6 | Vida: 4 | Ataque: 0 | Defensa: 0 | Velocidad: 0 | Actitud: Defensiva | Posición: 8E | Vive: Si
Nombre: Soldado1X7 | Vida: 1 | Ataque: 0 | Defensa: 0 | Velocidad: 0 | Actitud: Defensiva | Posición: 2F | Vive: Si

Datos del ejército 2:
Nombre: Soldado2X1 | Vida: 1 | Ataque: 0 | Defensa: 0 | Velocidad: 0 | Actitud: Defensiva | Posición: 5D | Vive: Si
Nombre: Soldado2X2 | Vida: 3 | Ataque: 0 | Defensa: 0 | Velocidad: 0 | Actitud: Defensiva | Posición: 8D | Vive: Si
Nombre: Soldado2X3 | Vida: 5 | Ataque: 0 | Defensa: 0 | Velocidad: 0 | Actitud: Defensiva | Posición: 10D | Vive: Si

Soldado con mayor vida del ejército 1:
Nombre: Soldado1X1 | Vida: 5 | Ataque: 0 | Defensa: 0 | Velocidad: 0 | Actitud: Defensiva | Posición: 2C | Vive: Si

Soldado con mayor vida del ejército 2:
Nombre: Soldado2X3 | Vida: 5 | Ataque: 0 | Defensa: 0 | Velocidad: 0 | Actitud: Defensiva | Posición: 10D | Vive: Si

Promedio de puntos de vida del ejército 1: 2.2857142857142856
Promedio de puntos de vida del ejército 2: 3.0

Ranking de soldados del ejército 1 (ordenado por puntos de vida de forma decreciente):
Nombre: Soldado1X1 | Vida: 5 | Ataque: 0 | Defensa: 0 | Velocidad: 0 | Actitud: Defensiva | Posición: 2C | Vive: Si
Nombre: Soldado1X6 | Vida: 4 | Ataque: 0 | Defensa: 0 | Velocidad: 0 | Actitud: Defensiva | Posición: 8E | Vive: Si
Nombre: Soldado1X5 | Vida: 3 | Ataque: 0 | Defensa: 0 | Velocidad: 0 | Actitud: Defensiva | Posición: 3D | Vive: Si
Nombre: Soldado1X2 | Vida: 1 | Ataque: 0 | Defensa: 0 | Velocidad: 0 | Actitud: Defensiva | Posición: 1F | Vive: Si
Nombre: Soldado1X3 | Vida: 1 | Ataque: 0 | Defensa: 0 | Velocidad: 0 | Actitud: Defensiva | Posición: 4F | Vive: Si
Nombre: Soldado1X4 | Vida: 1 | Ataque: 0 | Defensa: 0 | Velocidad: 0 | Actitud: Defensiva | Posición: 7J | Vive: Si
Nombre: Soldado1X7 | Vida: 1 | Ataque: 0 | Defensa: 0 | Velocidad: 0 | Actitud: Defensiva | Posición: 2F | Vive: Si

Ranking de soldados del ejército 2 (ordenado por puntos de vida de forma decreciente):
Nombre: Soldado2X3 | Vida: 5 | Ataque: 0 | Defensa: 0 | Velocidad: 0 | Actitud: Defensiva | Posición: 10D | Vive: Si
Nombre: Soldado2X2 | Vida: 3 | Ataque: 0 | Defensa: 0 | Velocidad: 0 | Actitud: Defensiva | Posición: 8D | Vive: Si
Nombre: Soldado2X1 | Vida: 1 | Ataque: 0 | Defensa: 0 | Velocidad: 0 | Actitud: Defensiva | Posición: 5D | Vive: Si

Resultado de la batalla:
El ejército 1 ha ganado la batalla. La suma total de sus puntos de vida es 16 superando en 7 puntos al ejército 2.
```

Diagrama de clases UML (hecho con Plant UML en VS Code):