
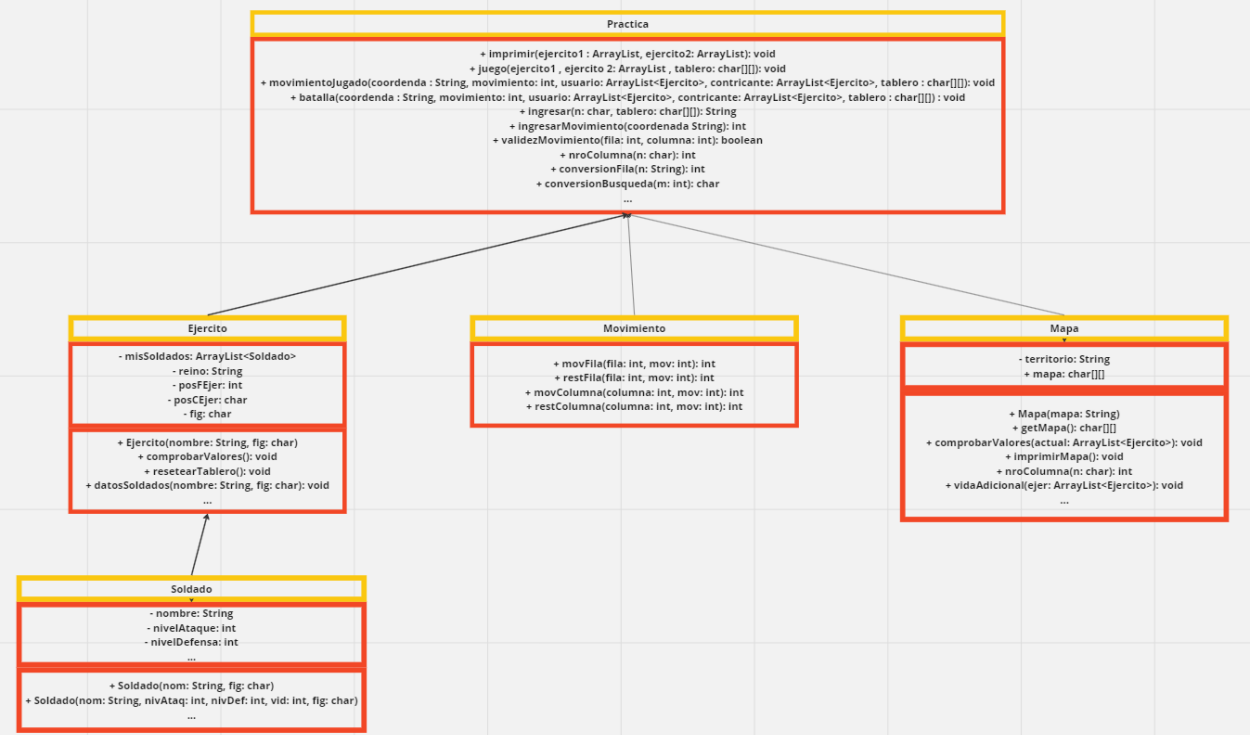
	<p style="text-align: center;">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p style="text-align: center;">Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLE-001</p>	<p>Página: 1</p>

INFORME DE LABORATORIO

INFORMACIÓN BÁSICA					
ASIGNATURA:	Fundamentos de la Programación 2				
TÍTULO DE LA PRÁCTICA:	Laboratorio 17 – Clase Ejercito / Soldado / Mapa				
NÚMERO DE PRÁCTICA:	17	AÑO LECTIVO:	1ro	NRO. SEMESTRE:	2
FECHA DE PRESENTACIÓN	17/01/2023	HORA DE PRESENTACIÓN	2:30 pm		
INTEGRANTE (s): Cuno Cahuari Armando Steven				NOTA:	

SOLUCIÓN Y RESULTADOS
<p>I. SOLUCIÓN DE EJERCICIOS/PROBLEMAS</p> <p>Link GitHub: https://github.com/SteveArms/LaboratorioFPIL.git</p>  <pre> classDiagram class Practica { +imprimir(ejercito1 : ArrayList, ejercito2: ArrayList): void + juego(ejercito1 , ejercito 2: ArrayList , tablero : char[][]): void + movimientojugado(coordenada : String, movimiento: int, usuario: ArrayList<Ejercito>, contricante: ArrayList<Ejercito>, tablero : char[][]): void + batalla(coordenada : String, movimiento: int, usuario: ArrayList<Ejercito>, contricante: ArrayList<Ejercito>, tablero : char[][]): void + ingresar(n: char, tablero: char[][]): String + ingresarMovimiento(coordenada String): int + validezMovimiento(fila: int, columna: int): boolean + nroColumna(n: char): int + conversionFila(n: String): int + conversionBusqueda(m: int): char } class Ejercito { - misSoldados: ArrayList<Soldado> - reino: String - posEjer: int - posCEjer: char - fig: char + Ejercito(nombre: String, fig: char) + comprobarValores(): void + resetearTablero(): void + datosSoldados(nombre: String, fig: char): void } class Soldado { - nombre: String - nivelAtaque: int - nivelDefensa: int + Soldado(nom: String, fig: char) + Soldado(nom: String, nivAtaq: int, nivDef: int, vid: int, fig: char) } class Movimiento { + movFila(fila: int, mov: int): int + restFila(fila: int, mov: int): int + movColumna(columna: int, mov: int): int + restColumna(columna: int, mov: int): int } class Mapa { - territorio: String + mapa: char[][] + Mapa(mapa: String) + getMapa(): char[][] + comprobarValores(actual: ArrayList<Ejercito>): void + imprimirMapa(): void + nroColumna(n: char): int + vidaAdicional(ejer: ArrayList<Ejercito>): void } Practica --> Ejercito Practica --> Movimiento Practica --> Mapa Ejercito --> Soldado </pre>

El Laboratorio nos plantea un juego el cual interactúen 2 personas las cuales podrán mover a sus ejércitos correspondientes teniendo una metrica para cuando se de una batalla entre ya sea el Ejercito del jugador actual y del enemigo.

Para ello fue necesario métodos los cuales hagan el ingreso de los datos correspondientes a cada ArrayList de Ejercito.

```
//Posicionamiento de un ArrayList de Ejercitos los cuales los datos de ellos ejercitos seran complementados mediante metodos que haran el ingreso de datos
ArrayList<Ejercito> actual = new ArrayList<Ejercito>();
ArrayList<Ejercito> enemigo = new ArrayList<Ejercito>();
boolean validez = true;
String map = aleatorioMapa();
Mapa tablero = new Mapa(map);
datosEjercitos(actual, m: '*');
System.out.println(x: "_____");
datosEjercitos(enemigo, m: '%');
```

Este método aportara en el ingreso de adicionar objetos de tipo Ejercito que vayan conformando a los ejércitos correspondientes por cada Reino.

```
public static void datosEjercitos(ArrayList<Ejercito> ejer, char m){
    int rand = (int)(Math.random() * 10 + 1);
    System.out.println("El nro de ejercitos es " + rand);
    System.out.println(x: "Ingrese el nombre del territorio: ");
    String ingresar = ingresarNombre();
    for(int i = 0; i < rand; i++){
        ejer.add(new Ejercito(ingresar, m));
    }
}

public static String ingresarNombre(){
```

Asimismo, dentro de este método habrá otro método que hará validar el nombre del reino correspondiente a un cierto conjunto de nombres validándose mediante while.

```
//Metodos que hacen validar el nombre correspondiente al arreglo en un cierto nro de Nombres correspondientes.
public static String ingresarNombre(){
    Scanner sc = new Scanner(System.in);
    System.out.println(x: "Ingrese el nombre del ejercito solo validos :");
    System.out.println(x: "Inglaterra - Francia - Castilla o Aragon - Sacro - Moros");
    String nombre = sc.next().toUpperCase();
    while(!validarNombre(nombre)){
        System.out.println(x: "Reino incorrecto ingrese otro");
        nombre = sc.next().toUpperCase();
    }
    return nombre;
}

public static boolean validarNombre(String nombre){
    return nombre.equals(anObject: "INGLATERRA") || nombre.equals(anObject: "FRANCIA") || nombre.equals(anObject: "CASTILLA") || nombre.equals(anObject: "ARAGON") || nombre.equals(anObject: "SACRO MOROS");
}

//Metodo para definir el entorno del territorio
public static String aleatorioMapa(){
    int rand = (int)(Math.random() * 5 + 1);
    switch(rand){
        case 1:
            return "BOSQUE";
        case 2:
            return "CAMPO ABIERTO";
        case 3:
            return "MONTANIA";
        case 4:
            return "DESIERTO";
        case 5:
            return "PLAYA";
        default:
            return " ";
    }
}

private static final int COLS = 10;

public Ejercito(String nombre, char fig){
    posFEjer = (int)(Math.random() * 10 + 1);
    posCEjer = numCol();
    reino = nombre;
    this.fig = fig;
    misSoldados = new ArrayList<Soldado>();
    datosSoldados(nombre, fig);
    comprobarValores();
}
```

Mediante un mismo método de la Clase Ejercito hacemos el añadido correspondiente de los Soldados que corresponderán al ejercito correspondiente.

```
//Datos de los soldados
public void datosSoldados(String nombre, char fig){
    int nroSoldados = (int)(Math.random() * 10 + 1);
    System.out.println("El ejercito tiene un total de " + nroSoldados + " soldados");
    for(int i = 0; i < nroSoldados; i++){
        misSoldados.add(new Soldado(nombre, fig));
    }
}
```

Una vez tenemos los datos correspondientes con respecto a los Ejércitos de ambos reinos pues tendremos un método el cual favorecerá el reino correspondiente si el ámbito del territorio es favorable para cierto país.

```
tablero.vidaAdicional(actual);
tablero.vidaAdicional(enemigo);
```

Esto se dará mediante métodos de clase Mapa los cuales añadirán vida adicional a cada uno de sus soldados correspondiente.

```
//Metodo encargado de hacer validar si tanto el territorio que fue formado el mapa guarda vinculo con el reino correspondiente.
public void vidaAdicional(ArrayList<Ejercito> ejer){
    if(territorio.equals(anObject:"BOSQUE") && ejer.get(index:0).getReino().equals(anObject:"INGLATERRA")){
        agregarVida(ejer);
    } else if(territorio.equals(anObject:"CAMPO ABIERTO") && ejer.get(index:0).getReino().equals(anObject:"FRANCIA")){
        agregarVida(ejer);
    } else if(territorio.equals(anObject:"MONTANIA") && (ejer.get(index:0).getReino().equals(anObject:"CASTILLA") || ejer.get(index:0).getReino().equals(anObject:"ARAGON"))){
        agregarVida(ejer);
    } else if(territorio.equals(anObject:"PLAYA") && ejer.get(index:0).getReino().equals(anObject:"MOROS")){
        agregarVida(ejer);
    } else if(territorio.equals(anObject:"DESIERTO") && ejer.get(index:0).getReino().equals(anObject:"SACRO")){
        agregarVida(ejer);
    }
}

//Metodo acoplado para agregar vida si tienen ventaja de territorio
public static void agregarVida(ArrayList<Ejercito> ejer){
    for(int i = 0; i < ejer.size(); i++){
        for(int j = 0; j < ejer.get(i).getSoldados().size(); j++){
            if(ejer.get(i).getSoldados().get(j).getNivVidaAct() < 5){
                ejer.get(i).getSoldados().get(j).addVida();
            }
        }
    }
}
```

Una vez realizado esto hacemos una comprobación de valores correspondiente tanto como el posicionamiento de lo Ejércitos al tener una posición en el mapa así también como la de los Soldados.

```
tablero.comprobarValores(actual);
tablero.comprobarValores(enemigo);

//El metodo encargado de verificar que las posiciones de los ejercitos no ocupen dos o mas un mismo recuadro
public void comprobarValores(ArrayList<Ejercito>actual){
    int fila;
    char columna;
    for(int i = 0; i < actual.size(); i++){
        fila = actual.get(i).getFila();
        columna = actual.get(i).getColumna();
        while(mapa[fila - 1][nroColumna(columna)] != '-'){
            fila = (int)(Math.random() * 10 + 1);
            columna = Soldado.numCol();
        }
        actual.get(i).setFila(fila);
        actual.get(i).setCol(columna);
        mapa[fila - 1][nroColumna(columna)] = actual.get(i).getFigura();
    }
}
```

Este código dado para la interacción mediante un juego se da por el mismo código trabajado en laboratorios anteriores.

```
//Inicia el juego entre ambos ejércitos
public static void juego(ArrayList<Ejercito> ejercito1, ArrayList<Ejercito> ejercito2, char[][]tablero){
    Scanner sc = new Scanner(System.in);
    boolean validez = true;
    String coordenada, jugar = "";
    int movimiento, fila, columna;
    int i = 0;
    while(validez){
        if(ejercito1.size()== 0 || ejercito2.size() == 0 || jugar.toUpperCase().equals(anObject:"NO")){
            validez = false;
            break;
        } else if(i % 2 == 0){
            System.out.println(x:"Turno del Ejercito 1");
            coordenada = ingresar(ejercito1.get(index:0).getFigura(), tablero);
            movimiento = ingresarMovimiento(coordenada);
            movimientoJugado(coordenada, movimiento, ejercito1, ejercito2, tablero);
            imprimirTablero(tablero);
        } else {
            System.out.println(x:"Turno del Ejercito 2");
            coordenada = ingresar(ejercito2.get(index:0).getFigura(), tablero);
            movimiento = ingresarMovimiento(coordenada);
            movimientoJugado(coordenada, movimiento, ejercito2, ejercito1, tablero);
            imprimirTablero(tablero);
        }
        System.out.println(x:"Desea seguir jugando: Ingrese si o no");
        jugar = sc.next();
        i++;
    }
    if(ejercito1.size() == 0){
        System.out.println(x:"Salio victorio el ejercito 2");
        for(Ejercito n: ejercito1){
            System.out.println(n);
        }
    } else{
        System.out.println(x:"Salio victorioso el ejercito 1");
        for(Ejercito m: ejercito2){
            System.out.println(m);
        }
    }
}
```

```

public static void movimientoJugado(String coordenada, int movimiento, ArrayList<Ejercito> usuario, ArrayList<Ejercito> contrincante, char[][] tablero){
    int fila_act, columna_act, fila_mov, columna_mov, posAct;
    fila_act = conversionFila(coordenada.substring(beginIndex:1, coordenada.length())) - 1;
    columna_act = nroColumna(coordenada.charAt(index:0));
    fila_mov = Movimiento.movFila(fila_act, movimiento);
    columna_mov = Movimiento.movColumna(columna_act, movimiento);
    if(tablero[fila_mov][columna_mov] == usuario.get(index:0).getFigura()){
        System.out.println(x:"Posicion ocupada por una figura de tu mismo ejercito");
    } else if(tablero[fila_mov][columna_mov] == '-'){
        posAct = buscadorPosicion(usuario, coordenada);
        usuario.get(posAct).setFila(fila_mov + 1);
        usuario.get(posAct).setCol(conversionBusqueda(columna_mov));
        tablero[fila_act][columna_act] = '-';
        tablero[fila_mov][columna_mov] = usuario.get(index:0).getFigura();
    } else {
        System.out.println(x:"Enemigo hallado");
        batalla(coordenada, movimiento, usuario, contrincante, tablero);
    }
}

//La metrica cuando se encuentran 2 ejercitos es por medio de porcentaje asimismo el valor total de vida por cada soldado por cada ejercito
public static void batalla(String coordenada, int movimiento, ArrayList<Ejercito> usuario, ArrayList<Ejercito> contrincante, char[][] tablero){
    int fila_act, columna_act, fila_mov, columna_mov, posAct, us, cont;
    fila_act = conversionFila(coordenada.substring(beginIndex:1, coordenada.length())) - 1;
    columna_act = nroColumna(coordenada.charAt(index:0));
    fila_mov = Movimiento.movFila(fila_act, movimiento);
    columna_mov = Movimiento.movColumna(columna_act, movimiento);
    String movRealizado = conversionBusqueda(columna_mov) + String.valueOf(fila_mov + 1);
    us = buscadorPosicion(usuario, coordenada);
    cont = buscadorPosicion(contrincante, movRealizado);
    double vidaEjercitoActual = usuario.get(us).cantidadVida();
    double vidaEjercitoEnemigo = contrincante.get(cont).cantidadVida();
    double nroRandom = (int)(Math.random() * 100 + 1);
    double probabilidad1 = (vidaEjercitoActual * 100) / (vidaEjercitoActual + vidaEjercitoEnemigo);
    System.out.println("Posibilidad de porcentaje del Ejercito actual: " + probabilidad1 + "%");
    System.out.println("Posibilidad de porcentaje del Ejercito enemigo: " + (100 - probabilidad1) + "%");
    System.out.println("El numero aleatorio fue " + nroRandom);
    if(0 <= nroRandom && nroRandom <= probabilidad1){
        contrincante.remove(cont);
        usuario.get(us).setFila(fila_mov + 1);
        usuario.get(us).setCol(conversionBusqueda(columna_mov));
        tablero[fila_mov][columna_mov] = usuario.get(us).getFigura();
        tablero[fila_act][columna_act] = '-';
        usuario.get(us).agregarVida();
    } else {
        usuario.remove(us);
        tablero[fila_act][columna_act] = '-';
        contrincante.get(cont).agregarVida();
    }
}



```

```
public static String ingresar(char n, char[][] tablero){
    String coordenada = "";
    int fila, columna;
    boolean validez = true;
    Scanner sc = new Scanner(System.in);
    while(validez){
        System.out.println(x:"1 2 3\n8 x 4\n7 6 5");
        System.out.println(x:"Ingrese otra coordenada");
        coordenada = sc.next();
        fila = conversionFila(coordenada.substring(beginIndex:1, coordenada.length()));
        columna = nroColumna(coordenada.charAt(index:0));
        System.out.println(fila + " " + columna);
        if(tablero[fila - 1][columna] == n){
            break;
        } else {
            System.out.println(x:"Coordenada incorrecta");
        }
    }
    return coordenada;
}

public static int ingresarMovimiento(String coordenada){
    int movimiento = 0;
    boolean validez = true;
    int fila, columna;
    Scanner sc = new Scanner(System.in);
    while(validez){
        System.out.println(x:"Ingresar Movimiento: ");
        movimiento = sc.nextInt();
        fila = conversionFila(coordenada.substring(beginIndex:1, coordenada.length())) - 1;
        columna = nroColumna(coordenada.charAt(index:0));
        fila = Movimiento.movFila(fila, movimiento);
        columna = Movimiento.movColumna(columna, movimiento);
        System.out.println(fila + " Movimiento aplicado " + columna);
        if(validezMovimiento(fila, columna)){
            break;
        } else {
            System.out.println(x:"Movimiento invalido");
            fila = Movimiento.restFila(fila, movimiento);
            columna = Movimiento.restColumna(columna, movimiento);
        }
    }
    return movimiento;
}

public static boolean validezMovimiento(int fila, int columna){
    return fila <= 9 && fila >= 0 && columna <= 9 && columna >= 0;
}
```

Ejecutado el código :

	<p style="text-align: center;">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
<p style="text-align: center;">Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación</p>		
<p>Aprobación: 2022/03/01</p>	<p>Código: GUIA-PRLE-001</p>	<p>Página: 8</p>



```

El nro de ejercitos es 9
Ingrese el nombre del territorio:
Ingrese el nombre del ejercito solo validos :
Inglaterra - Francia - Castilla o Aragon - Sacro - Moros
francia
El ejercito tiene un total de 6 soldados
El ejercito tiene un total de 4 soldados
El ejercito tiene un total de 10 soldados
El ejercito tiene un total de 2 soldados
El ejercito tiene un total de 2 soldados
El ejercito tiene un total de 5 soldados
El ejercito tiene un total de 2 soldados
El ejercito tiene un total de 2 soldados
El ejercito tiene un total de 2 soldados

-----
El nro de ejercitos es 1
Ingrese el nombre del territorio:
Ingrese el nombre del ejercito solo validos :
Inglaterra - Francia - Castilla o Aragon - Sacro - Moros
inglaterra
El ejercito tiene un total de 7 soldados
El mapa se encuentra en CAMPO ABIERTO
- % * - * - * - - -
- - - - - * -
- - * - - * - - -
- - - - - - -
- - * - - - -
- * - - - - -
- - - - - - -
- - - - - - -
- - * - - - -

Reino FRANCIA
Ejercito 1 Total de vida : 16 Cantidad de Soldados : 6
Ejercito 2 Total de vida : 16 Cantidad de Soldados : 4
Ejercito 3 Total de vida : 24 Cantidad de Soldados : 10
Ejercito 4 Total de vida : 5 Cantidad de Soldados : 2
Ejercito 5 Total de vida : 3 Cantidad de Soldados : 2
Ejercito 6 Total de vida : 19 Cantidad de Soldados : 5
Ejercito 7 Total de vida : 6 Cantidad de Soldados : 2
Ejercito 8 Total de vida : 5 Cantidad de Soldados : 2
Ejercito 9 Total de vida : 8 Cantidad de Soldados : 2
Reino INGLATERRA
Ejercito 1 Total de vida : 23 Cantidad de Soldados : 7
Tienen vida adicional los que tienen ventaja en entorno CAMPO ABIERTO
Turno del Ejercito 1
1 2 3
8 x 4
7 6 5
Ingrese otra coordenada
C1
1      2

```


	<p align="center">UNIVERSIDAD NACIONAL DE SAN AGUSTIN FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMA</p>	
Formato: Guía de Práctica de Laboratorio / Talleres / Centros de Simulación		
Aprobación: 2022/03/01	Código: GUIA-PRLE-001	Página: 9

<pre> Enemigo Hallado Posibilidad de porcentaje del Ejercito actual: 28.13% Posibilidad de porcentaje del Ejercito enemigo: 71.87% El numero aleatorio fue 1.72 - * - - * - * - - - - - - - - * - - - * - - * - - - - - - - - - - - - - - * - - - - - - - * - * - * - - - - - - Desea seguir jugando: Ingrese si o no no </pre>	
---	--

II. CONCLUSIONES

La implementación de la Clase Ejercito debe ser mas sofisticada tanto en métodos como atributos ya que ofrece variedad al ajustar el posicionamiento de sus soldados correspondientes.

RETROALIMENTACIÓN GENERAL

Mejorar la síntesis del código en java asimismo la implementación de bibliotecas que nos ofrece este lenguaje de programación.

REFERENCIAS Y BIBLIOGRAFÍA

Fundamentos de la Programacion 2.Topicos de Programacion orientada a objetos – Marcos Aedo & Eveling Castro