MANUAL DE TÉCNICO

PRÁCTICA 2, IPC1
PRIMER SEMESTRE 2021

DANNY HUGO BRYAN TEJAXÚN PICHIYÁ - 201908355

INTRODUCCIÓN

EL DOCUMENTO DA A CONOCER CADA UNA DE LAS FUNCIONES DE LOS MÉTODOS Y CLASES QUE CONFORMAN UN SOFTWARE ENFOCADO A REALIZAR OPERACIONES O PROCESOS DE GESTIÓN DE UN JUEGO POR CONSOLA, BASADO EN POKÉMON.

OBJETIVO

DAR A CONOCER CADA PROCEDIMIENTO IMPORTANTE QUE SOSTIENE EL BUEN FUNCIONAMIENTO DEL SOFTWARE.

BRINDAR SOPORTE A NUEVOS TÉCNICOS SOBRE DAR UN MANTENIMIENTO ÓPTIMO DEL SOFTWARE O PARA PODER MEJORARLO.

CLASE ENFRENTAMIENTOS

MÉTODO ENFRENTAR

EL MÉTODO RECIBE LOS PARÁMETROS ENVIADOS POR EL MÉTODO DE LA CLASE CONTROLADORA, QUE SON ID DE GIMNASIO Y DE LOS POKÉMONS. BÚSCA CADA PARÁMETRO EN LOS RESPECTIVOS ARREGLOS PARA EMPAREJARLOS EN EL ENFRENTAMIENTO, LAS VARIABLES SON DE TIPO ESTÁTICA PARA MANTENER EL VALOR OBTENIDO DESPUÉS DE EJECUTARSE EL MÉTODO Y PASAR AL SIGUIENTE. LOS MÉTODOS ANUNCIAR Y CALIFICAR SON ÚNICAMENTE PARA IMPRIMIR EN PANTALLA LOS NOMBRES DE LOS PELEADORES Y EL GANADOR.

```
public static void enfrentar(int gym,int pokemon1,int pokemon2) {
    for(int i=0; i<gimnasios.length; i++) {
        if(gimnasios[i]!=null) {
            if(gimnasios[i].id==gym) {
                gimnasio = gimnasios[i].lugar;
            }
        }
    }
    for(int i=0; i<pokemons.length; i++) {
        if(pokemons[i]!=null) {
            if(pokemons[i].id==pokemon1) {
                tipo1 = pokemons[i].tipo;
                nombre1 = pokemons[i].nombre;
                vida1 = pokemons[i].vida;
                ataque1 = pokemons[i].puntosAtaque;
                capturado1 = pokemons[i].capturado;
                estado1 = pokemons[i].estado;
            }
            if(pokemons[i].id==pokemon2) {
                tipo2 = pokemons[i].tipo;
                nombre2 = pokemons[i].nombre;
                vida2 = pokemons[i].vida;
                ataque2 = pokemons[i].puntosAtaque;
                capturado2 = pokemons[i].capturado;
                estado2 = pokemons[i].estado;
            }
        }
    }
    VerificacionPeleas verificar = new VerificacionPeleas();
    verificar.Anunciar();
    AtaqueP1();
    verificar.Calificar(pokemon1,pokemon2);
}
```

MÉTODO ATAQUEP1

ESTE MÉTODO EJECUTA CADA RONDA DE ATAQUE DEL PRIMER POKÉMON. SI LA VIDA DEL POKÉMON ES MAYOR A CERO, ATACA QUITÁNDOLE LA CANTIDAD DE SU ATAQUE A LA VIDA DEL SEGUNDO POKÉMON POR CADA REPETICIÓN DEL BUCLE. SI SE CUMPLE LA CONDICIÓN DE LA VIDA DEL SEGUNDO POKÉMON, EJECUTA EL MÉTODO ATAQUEP2, DE LO CONTRARIO ROMPE EL BUCLE. Y VUELVE A EJECUTARSE EL MÉTODO ATACAR PARA RECIBIR NUEVOS PARÁMETROS. EL BUCLE PRINCIPAL SE REPITE 3 VECES COMO MÁXIMO.

```
public static void AtaqueP2(int i1) {
    for(int i=i1; i<(i1+1); i++) {
        vida1 -= ataque2;
        if(vida1>0) {
            System.out.print("    - Ataque 2: "+nombre2+
                             " deja con "+vida1+" de vida a "+nombre1);
        }else {
            System.out.print("    - Ataque 2: "+nombre2+
                    " deja con "+0+" de vida a "+nombre1);
        }
    }
}
```

MÉTODO ATAQUEP2

EL MÉTODO SE EJECUTA EN FUNCIÓN DE ATAQUEP1. SI SE CUMPLE LA CONDICIÓN DE LA VIDA DEL PRIMER POKÉMON O NO SE CUMPLA LO ÚNICO QUE CAMBIA ES EL VALOR QUE TENDRÁ LA VIDA DEL QUE PIERDA LA PELEA.

```
public static void AtaqueP1() {
    for(int i=1; i<4; i++) {
        if(vida1>0) {
            vida2 -= ataque1;
            if(vida2>0) {
                System.out.println("\n   Ronda "+i);
                System.out.println("    - Ataque 1: "+nombre1+
                                   " deja con "+vida2+" de vida a "+nombre2);
                AtaqueP2(i);
            }else {
                System.out.println("\n   Ronda "+i);
                System.out.println("    - Ataque 1: "+nombre1+
                                   " deja con "+0+" de vida a "+nombre2);
                break;
            }
        }
    }
}
```

CLASE VERIFICACIÓN PELEAS

MÉTODO CALIFICAR

ESTE MÉTODO DEFINE QUIÉN ES EL GANADOR. LA PRIMERA CONDICIÓN DECIDE QUE SI AMBOS TIENEN LA MISMA VIDA ES PORQUE AMBOS POKÉMONS YA ESTABAN MUERTOS ANTES DE LA PELEA, LA SEGUNDA CONDICIÓN COMPARA LAS VIDAS DE AMBOS LUEGO DE FINALIZAR LAS 3 RONDAS DE LA PELEA, SI LA VIDA DEL PRIMER POKÉMON ES MAYOR QUE LA DEL SEGUNDO, ENTONCES EL PRIMER POKÉMON ES EL GANADOR, DE LO CONTRARIO EL SEGUNDO POKÉMON ES EL GANADOR. EN CADA UNA SE EJECUTA EL MÉTODO ACTUALIZAR AL QUE SE LE ENVÍAN LOS ID DE LOS POKÉMONS COMO PARÁMETROS.

```
public static void Calificar(int pokemon1,int pokemon2) {
    if(vida1==vida2) {
        Actualizar(pokemon1,pokemon2);
        campeon = "Los 2 Pokémons estaban muertos";
        System.out.printf("\n%-10s%-5s\n\n\n","",campeon);
    }else if(vida1>vida2) {
        Actualizar(pokemon1,pokemon2);
        campeon = nombre1;
        System.out.printf("\n%-10s%-5s\n\n\n","","!!!Ha Ganado "+campeon+"!!!");
    }else {
        Actualizar(pokemon1,pokemon2);
        campeon = nombre2;
        System.out.printf("\n%-10s%-5s\n\n\n","","!!!Ha Ganado "+campeon+"!!!");
    }
}
```

MÉTODO ACTUALIZAR

ESTE MÉTODO RECIBE DOS COMO PARÁMETROS LOS ID DE LOS DOS POKÉMONS, LAS CONDICIONES SON LAS MISMAS PARA AMBOS POKÉMONS, PERO SE EJECUTAN POR SEPARADO. SI VIDA ES MAYOR QUE CERO EN UN OBJETO LLAMADO RESPALDO SE ALMACENAN LOS NUEVOS DATOS DE AMBOS POKÉMONS LUEGO DE LA PELEA, ES DECIR SU VIDA Y ESTADO, LUEGO SE RECORRE EL ARREGLO CORRESPONDIENTE A LOS POKÉMONS HASTA ENCONTRAR EL ID BUSCADO PARA PODER INSERTAR LOS NUEVOS DATOS EN EL ARREGLO. DE LO CONTRARIO, SE GUARDARÁN LOS MISMOS DATOS, PERO CON VIDA CERO Y ESTADO MUERTO. SE SERIALIZA PARA CADA CONDICIÓN.

```
public static void Actualizar(int pokemon1,int pokemon2) {
    if(vida1>0) {
        respaldo = new Pokemon(pokemon1,tipo1,nombre1,vida1,ataque1,capturado1,true);
        for(int i=0; i<pokemons.length; i++) {
            if(pokemons[i]!=null) {
                if(pokemons[i].id==pokemon1) {
                    pokemons[i]=respaldo;
                }
            }
        }
        ControladorPokemon serPok = new ControladorPokemon();
        serPok.Serializar();
    }else {
        respaldo = new Pokemon(pokemon1,tipo1,nombre1,0,ataque1,capturado1,false);
        for(int i=0; i<pokemons.length; i++) {
            if(pokemons[i]!=null) {
                if(pokemons[i].id==pokemon1) {
                    pokemons[i]=respaldo;
                }
            }
        }
        ControladorPokemon serPok = new ControladorPokemon();
        serPok.Serializar();
    }
    if(vida2>0) {
        respaldo = new Pokemon(pokemon2,tipo2,nombre2,vida2,ataque2,capturado2,true);
        for(int i=0; i<pokemons.length; i++) {
            if(pokemons[i]!=null) {
                if(pokemons[i].id==pokemon2) {
                    pokemons[i]=respaldo;
                }
            }
        }
        ControladorPokemon serPok = new ControladorPokemon();
        serPok.Serializar();
    }else {
        respaldo = new Pokemon(pokemon2,tipo2,nombre2,0,ataque2,capturado2,false);
        for(int i=0; i<pokemons.length; i++) {
            if(pokemons[i]!=null) {
                if(pokemons[i].id==pokemon2) {
                    pokemons[i]=respaldo;
                }
            }
        }
        ControladorPokemon serPok = new ControladorPokemon();
        serPok.Serializar();
    }
}
```

CLASE ALIMENTAR

MÉTODO ALIMENTAR

RECIBE COMO PARÁMETROS EL ID DEL ALIMENTO Y DEL POKÉMON QUE LO COMERÁ. RECORRE LOS ARREGLOS HASTA ENCONTRAR EL ID IGUAL AL PARÁMETRO. SE EJECUTA EL MÉTODO COMER Y SE LE ENVÍA COMO PARÁMETRO EL ID DEL POKÉMON.

```
public static void alimentar(int alimento,int pokemon) {
    for(int i=0; i<alimentos.length; i++) {
        if(alimentos[i]!=null) {
            if(alimentos[i].id==alimento) {
                comida = alimentos[i].nombre;
                salud = alimentos[i].vida;
            }
        }
    }
    for(int i=0; i<pokemons.length; i++) {
        if(pokemons[i]!=null) {
            if(pokemons[i].id==pokemon) {
                tipo = pokemons[i].tipo;
                nombre = pokemons[i].nombre;
                vida = pokemons[i].vida;
                ataque = pokemons[i].puntosAtaque;
                capturado = pokemons[i].capturado;
                estado = pokemons[i].estado;
            }
        }
    }
    comer(pokemon);
    System.out.println(nombre+" ha comido "+comida+" y ha recuperado "+salud+" de vida");
}
```

MÉTODO COMER

RECIBE COMO PARÁMETROS EL ID DEL POKÉMON. SUMA LA SALUD DEL ALIMENTO A LA VIDA DEL POKÉMON. SI LA VIDA DEL POKÉMON ES MAYOR QUE CERO SE GUARDARÁ EN UN OBJETO LLAMADO RESPALDO LOS NUEVOS DATOS LUEGO DE ALIMENTARSE SI ESTABA MUERTO AHORA PASARÁ A ESTADO VIVO. DE LO CONTRARIO SEGUIRÁ ESTANDO MUERTO Y CON VIDA CERO. SE SERIALIZA EN CADA CONDICIÓN

```
public static void comer(int pokemon) {
    vida+=salud;
    if(vida>0) {
        respaldo = new Pokemon(pokemon,tipo,nombre,vida,ataque,capturado,true);
        for(int i=0; i<pokemons.length; i++) {
            if(pokemons[i]!=null) {
                if(pokemons[i].id==pokemon) {
                    pokemons[i]=respaldo;
                }
            }
        }
        ControladorPokemon serPok = new ControladorPokemon();
        serPok.Serializar();
    }else {
        respaldo = new Pokemon(pokemon,tipo,nombre,0,ataque,capturado,false);
        for(int i=0; i<pokemons.length; i++) {
            if(pokemons[i]!=null) {
                if(pokemons[i].id==pokemon) {
                    pokemons[i]=respaldo;
                }
            }
        }
        ControladorPokemon serPok = new ControladorPokemon();
        serPok.Serializar();
    }
}
```

CLASE REPORTE TOP

ORDENAMIENTO BURBUJA

SE USÓ PARA EL REPORTE DE TOP 5 TANTO PARA ALIMENTOS COMO PARA POKÉMONS. SE CREÓ UN ARREGLO AUXILIAR PARA NO MODIFICAR EL ARREGLO EN DONDE SE MATIENEN LOS DATOS ORIGINALES INTACTOS. SE RECORRE EL NUEVO ARREGLO CON LOS DATOS GUARDADOS. SE CREA UN OBJETO TEMPORAL LA CONDICIÓN ES: SI LA POSICIÓN ACTUAL ES MENOR A LA POSICIÓN SIGUIENTE, EL OBJETO TEMPORAL ES IGUAL A LOS DATOS DE LA POSICIÓN SIGUIENTE, LA POSICIÓN SIGUIENTE ES IGUAL A LA POSICIÓN ACTUAL Y LA POSICIÓN ACTUAL ES IGUAL AL OBJETO TEMPORAL Y SE ROMPE EL BUCLE PARA NO LEER DATOS INNECESARIOS.

```
for(int i=0; i<pokemons.length; i++) {
    pok[i]=pokemons[i];
}
try {
    for(int i=0; i<pok.length; i++) {
        for(int j=0; j<pok.length;j++) {
            if(pok[j].getPuntosAtaque()<pok[j+1].getPuntosAtaque()) {
                Pokemon temp = pok[j+1];
                pok[j+1] = pok[j];
                pok[j] = temp;
                break;
            }
        }
    }
}catch(Exception e) {}
```