

## Manual técnico

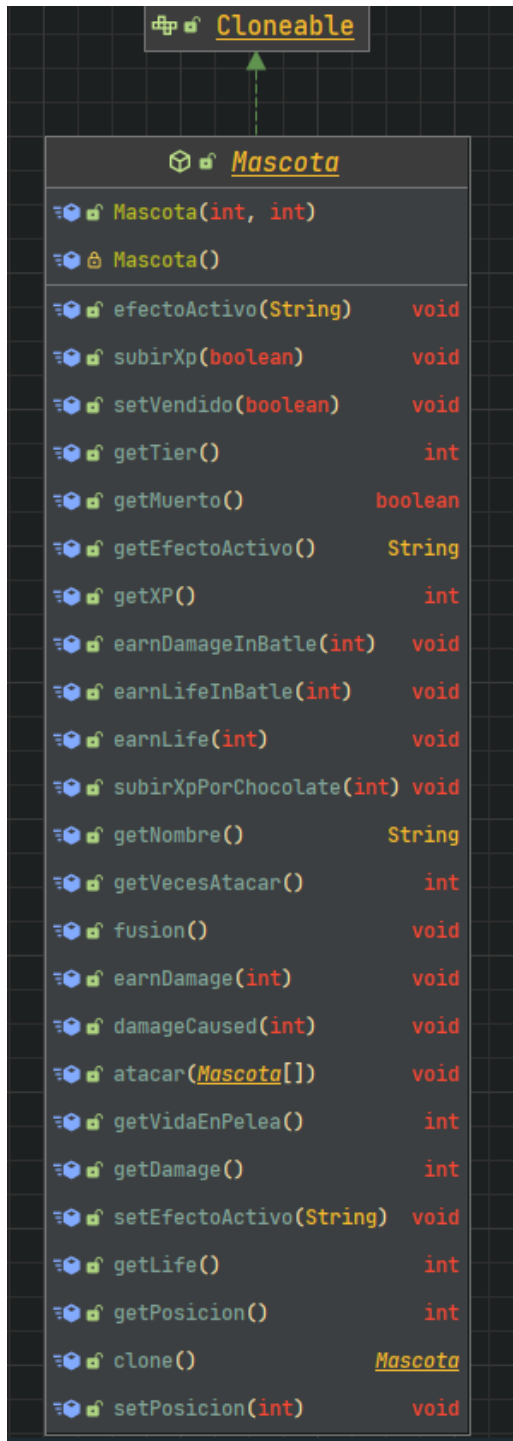
Sistema operativo: Windows 10

IDE: IntelliJ IDEA Versión:ultimate 2021.3.3

Lenguaje: Java

JDK: 15

Diagrama UML de clase mascota(abstracta):



Código:

```
public abstract class Mascota implements Cloneable {
```

```
protected int life;  
protected int damage;  
protected int xp = 0;  
protected int level = 1;  
protected int posicion;  
protected int tier;  
protected int vidaEnPelea = life;  
protected int danioEnPelea = damage;  
protected int vecesAtacar = 0;  
protected boolean muerto = false;  
protected boolean vendido = false;  
protected String efectoActivo = "Ninguno";  
protected String nombre;
```

```
protected Mascota() {  
}
```

```
public String getNombre() {  
    return nombre;  
}
```

```
public int getXP() {  
    return this.xp;  
}
```

```
public boolean getMuerto() {  
    return this.muerto;  
}
```

```
public int getVecesAtacar() {  
    return this.vecasAtacar;  
}
```

```
public void setVendido(boolean vendido) {  
    this.vendido = vendido;  
}
```

```
public void setEfectoActivo(String efectoActivo) {  
    this.efectoActivo = efectoActivo;  
}
```

```
public String getEfectoActivo() {  
    return efectoActivo;  
}
```

```
public int getPosicion() {
```

```

        return posicion;
    }

    public void setPosicion(int posicion) {
        this.posicion = posicion;
    }

    public int getDamage() {
        return this.damage;
    }

    public int getLife() {

        return life;
    }

    public int getVidaEnPelea() {
        return this.vidaEnPelea;
    }

    public int getTier() {

        return tier;
    }

    public void efectoActivo(String fruta) {
        if (fruta.equals("Cebolla")) {
            this.efectoActivo = "Cebolla";
        } else if (fruta.equals("Melon")) {
            this.efectoActivo = "Melon";
        } else if (fruta.equals("Miel")) {
            this.efectoActivo = "Miel";
        } else if (fruta.equals("Hueso de carne")) {
            this.efectoActivo = "Hueso de carne";
        } else if (fruta.equals("Chile")) {
            this.efectoActivo = "Chile";
        }
    }

    public Mascota(int life, int damage) {
        this.life = life;
        this.damage = damage;
    }

    public void fusion() {

    }

```

```

    public void subirXp(boolean fusion) {
        if (!fusion) {
            System.out.println("No se puede fusionar");
        } else {
            xp += 1;
            this.life += 1;
            this.damage += 1;
        }
        //comprobando si subio de nivel
        if (xp == 2) {
            System.out.println("Mascota subio a nivel 2");
            level = 2;
        } else if (xp == 5) {
            System.out.println("Mascota subio a nivel 3");
            level = 3;
        }
    }
}

```

```

    public void subirXpPorChocolate(int extraXp) {
        this.xp += extraXp;
    }
}

```

```

    public void damageCaused(int damageCaused) {
        life -= damageCaused;
        if (life <= 0) {
            this.muerto = true;
        }
    }
}

```

```

}

```

```

    public void atacar(Mascota[] mascotasEnemigas) {
        if (mascotasEnemigas[0].getEfectoActivo().equals("Ajo"))
        {
            this.damage = this.damage - 2;
            mascotasEnemigas[0].damageCaused(this.damage);
            vecesAtacar += 1;
        } else if
(mascotasEnemigas[0].getEfectoActivo().equals("Melon")) {
            this.damage = 0;
            mascotasEnemigas[0].damageCaused(this.damage);
            vecesAtacar += 1;
            mascotasEnemigas[0].setEfectoActivo("Ninguno");
        } else if
(mascotasEnemigas[0].getEfectoActivo().equals("Ninguno")) {
            mascotasEnemigas[0].damageCaused(this.damage);
        }
    }
}

```

```

        vecesAtacar += 1;
    } else if
(mascotasEnemigas[0].getEfectoActivo().equals("Miel")) {
        mascotasEnemigas[0].damageCaused(this.damage);
        if (mascotasEnemigas[0].getMuerto()) {
            mascotasEnemigas[0] = new Abeja(1, 1, 1);
        }
    }
}

public void earnDamage(int extraDamage) {
    this.damage += extraDamage;
}

public void earnDamageInBatle(int extraDamage) {
    danioEnPelea = +extraDamage;
}

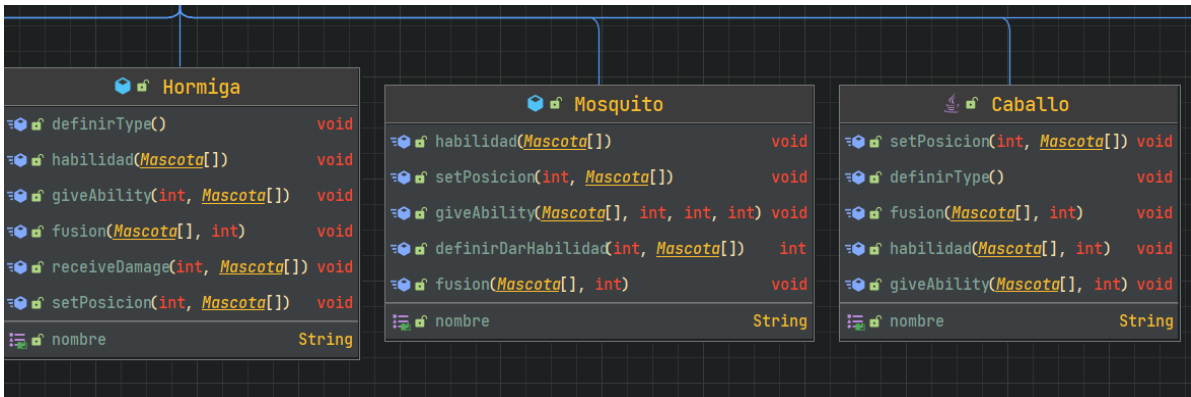
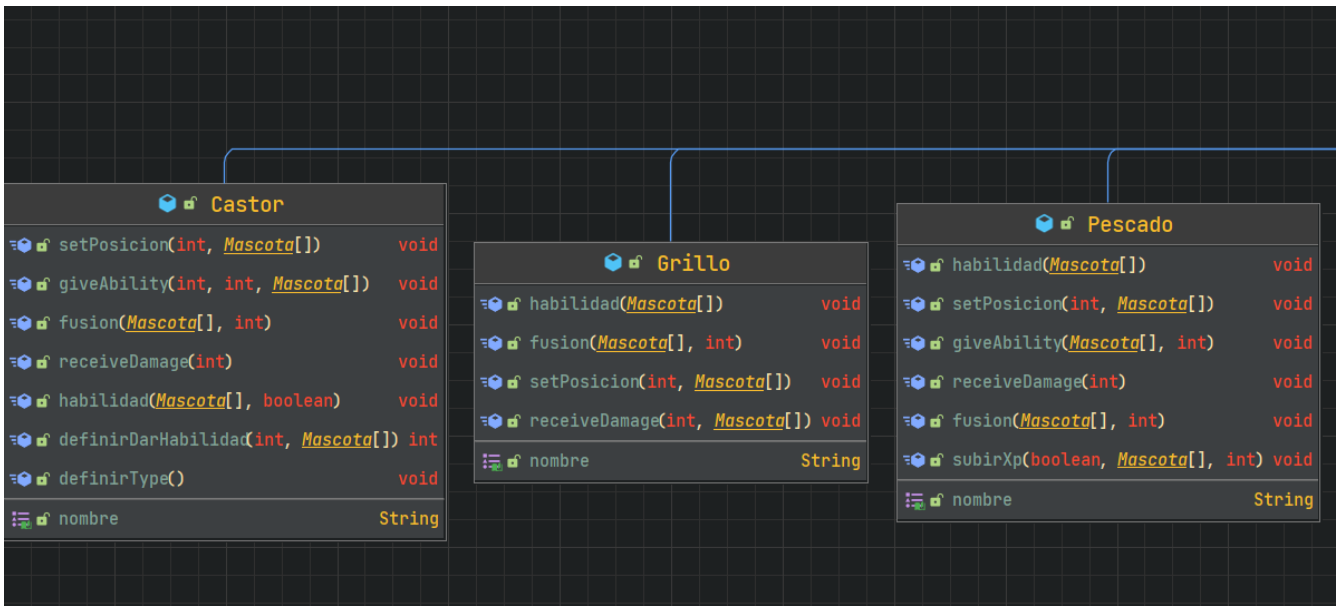
public void earnLifeInBatle(int extraLife) {
    vidaEnPelea += extraLife;
}

public void earnLife(int extraLife) {
    this.life += extraLife;
}













@Override
public Mascota clone() {
    try {
        return (Mascota) super.clone();
    } catch (CloneNotSupportedException e) {
        throw new AssertionError();
    }
}

```

Diagrama UML de mascotas tier1



## Nutria

  setPosition(int, <u>Mascota</u> [])	void
  habilidad( <u>Mascota</u> [])	void
  definirDarHabilidad(int, <u>Mascota</u> [])	int
  fusion( <u>Mascota</u> [], int)	void
  giveAbility( <u>Mascota</u> [], int)	void
  nombre	String

Castor:

```
public class Castor extends Mascota {

    private String[] type = new String[2];
    protected String nombre="Castor";

    public String getNombre() {
        return nombre;
    }

    public Castor(int damage, int life, int tier) {
        this.life = life;
        this.damage = damage;
        this.tier = tier;
    }

    //definir el tipo de la mascota
    public void definirType() {
        this.type[0] = "Acuatico";
        this.type[1] = "Terrestre";
    }

    //definir posicion
    public void setPosicion(int posicion, Mascota misMascotas[]) {
        this.posicion = posicion;
        Castor castor = new Castor(2, 2, 1);
        misMascotas[this.posicion] = castor;
    }

    public void habilidad(Mascota misMascotas[], boolean vendido) {
        int buffAnimal1 =0;
        int buffAnimal2 =0;
        if (vendido) {
            giveAbility(definirDarHabilidad(buffAnimal1 misMascotas),
definirDarHabilidad(buffAnimal2,misMascotas), misMascotas);
        }
    }

    //definir lo que hara su habilidad
    public void giveAbility(int buffAnimal1, int buffAnimal2, Mascota
misMascotas[]) {
        if (this.level == 1) {
            int extraLife = 1;
            misMascotas[buffAnimal1].earnLife(extraLife);
            misMascotas[buffAnimal2].earnLife(extraLife);
        } else if (this.level == 2) {
```



```

        int extraLife = 2;
        misMascotas[buffAnimal1].earnLife(extraLife);
        misMascotas[buffAnimal2].earnLife(extraLife);
    } else if (this.level == 3) {
        int extraLife = 3;
        misMascotas[buffAnimal1].earnLife(extraLife);
        misMascotas[buffAnimal2].earnLife(extraLife);
    }
}

```

*//Establecer que la habilidad le caiga a 2 animales aleatorios que no sean el*

```

public int definirDarHabilidad(int buffAnimal, Mascota[]
misMascotas) {

```

```

    while (buffAnimal == this.posicion &&
misMascotas[buffAnimal]==null) {
        buffAnimal = (int) (Math.random() * (5 - 0) + 0);
    }
    return buffAnimal;
}

```

```

//fusionar animal
public void fusion(Mascota misMascotas[], int animalAFusionar) {
    boolean fusion;
    if (misMascotas[this.posicion] ==
misMascotas[animalAFusionar]) {
        fusion = true;
    } else {
        fusion = false;
    }
    Castor.super.subirXp(fusion);
}

```

```

}
public void receiveDamage(int damageCaused) {
    vidaEnPelea -= damageCaused;
}

```

Grillo:

```
public class Grillo extends Mascota {

    private String type = "Insecto";
    protected String nombre="Grillo";

    public String getNombre() {
        return nombre;
    }

    public Grillo(int damage, int life, int tier) {
        this.life = life;
        this.damage = damage;
        this.tier = tier;
    }

    //definir posicion
    public void setPosicion(int posicion, Mascota misMascotas[])
    {
        this.posicion = posicion;
        Grillo grillo = new Grillo(1, 2, 1);
        misMascotas[this.posicion] = grillo;
    }

    //definiendo que hara su habilidad
    public void habilidad(Mascota misMascotas[]) {

        if (level == 1) {
            misMascotas[0] = new GrilloZombie(1, 1, 1);
        } else if (level == 2) {
            misMascotas[0] = new GrilloZombie(2, 2, 1);
        } else if (level == 3) {
            misMascotas[0] = new GrilloZombie(3, 3, 1);
        }
    }

    //fusionar animal
    public void fusion(Mascota misMascotas[], int
    animalAFusionar) {
        if (misMascotas[this.posicion] ==
        misMascotas[animalAFusionar]) {
            boolean fusion = true;
```

```

        Grillo.super.subirXp(fusion);
    } else {
        boolean fusion = false;
        Grillo.super.subirXp(fusion);
    }

}

public void receiveDamage(int damageCaused, Mascota
misMascotas[]) {
    life -= damageCaused;
    if (life <= 0) {
        habilidad(misMascotas);
    }
}
}

```

Pescado:

```

public class Pescado extends Mascota {

    private String type = "Acuatico";
    protected String nombre="Pescado";

    public String getNombre() {
        return nombre;
    }

    public Pescado(int damage, int life, int tier) {
        this.life = life;
        this.damage = damage;
        this.tier = tier;
    }

    //definir poscion
    public void setPosicion(int posicion, @NotNull Mascota
misMascotas[]) {
        this.posicion = posicion;
        Pescado pescado = new Pescado(2, 3, 1);
        misMascotas[posicion] = pescado;
    }

    //definir Habilidad
    public void habilidad(Mascota misMascotas[]) {
        for (int i = 0; i < misMascotas.length; i++) {

```

```

        if (misMascotas[i] == misMascotas[this.posicion]) {

        } else {
            giveAbility(misMascotas, i);
        }
    }

}

//definir lo que hara su habilidad
public void giveAbility(Mascota misMascotas[], int i) {
    if (this.level == 2) {
        int extraDamage = 1;
        int extraLife = 1;
        misMascotas[i].earnDamage(extraDamage);
        misMascotas[i].earnLife(extraLife);
    } else if (this.level == 3) {
        int extraDamage = 2;
        int extraLife = 2;
        misMascotas[i].earnDamage(extraDamage);
        misMascotas[i].earnLife(extraLife);
    }
}

//fusionar animal
public void fusion(Mascota misMascotas[], int
animalAFusionar) {
    boolean fusion;
    if (misMascotas[this.posicion] ==
misMascotas[animalAFusionar]) {
        fusion = true;
    } else {
        fusion = false;
    }
    subirXp(fusion, misMascotas, animalAFusionar);
}

public void subirXp(boolean fusion, Mascota[] misMascotas,
int animalAFusionar) {
    if (!fusion) {
        System.out.println("No se puede fusionar");
    } else {
        xp += 1;
    }
    //comprobando si subio de nivel
    if (xp == 2) {

```

```

        System.out.println("Mascota subio a nivel 2");
        level = 2;
        habilidad(misMascotas);
    } else if (xp == 5) {
        System.out.println("Mascota subio a nivel 3");
        level = 3;
        habilidad(misMascotas);
    }
}

```

## Hormiga

```

public class Hormiga extends Mascota {

    private String[] type = new String[2];
    protected String nombre="Hormiga";

    public String getNombre() {
        return nombre;
    }

    public Hormiga(int damage, int life, int tier) {
        this.life = life;
        this.damage = damage;
        this.tier = tier;
    }

    //definir el tipo de la mascota
    public void definirType() {
        this.type[0] = "Insecto";
        this.type[1] = "Terrestre";
    }

    //definir posicion
    public void setPosicion(int posicion, Mascota misMascotas[])
    {
        this.posicion = posicion;
        Hormiga hormiga = new Hormiga(2, 1, 1);
        misMascotas[posicion] = hormiga;
    }

    //definiendo a quien le dara su habilidad
    public void habilidad(Mascota misMascotas[]) {

        int animalesAtras = 4 - this.posicion;

        if (animalesAtras == 4) {

```

```

        int noAnimal = (int) (Math.random() * (4 - 1) + 1);
        giveAbility(noAnimal, misMascotas);
    } else if (animalesAtras == 3) {
        int noAnimal = (int) (Math.random() * (4 - 2) + 2);
        giveAbility(noAnimal, misMascotas);
    } else if (animalesAtras == 2) {
        int noAnimal = (int) (Math.random() * (4 - 3) + 3);
        giveAbility(noAnimal, misMascotas);
    } else if (animalesAtras == 1) {
        int noAnimal = 4;
        giveAbility(noAnimal, misMascotas);
    }
}

```

```

}

```

```

//definir lo que hara su habilidad
public void giveAbility(int noAnimal, Mascota misMascotas[])
{
    if (this level == 1) {
        int extraDamage = 2;
        int extraLife = 1;
        misMascotas[noAnimal].earnDamage(extraDamage);
        misMascotas[noAnimal].earnLife(extraLife);
    } else if (this level == 2) {
        int extraDamage = 4;
        int extraLife = 2;
        misMascotas[noAnimal].earnDamage(extraDamage);
        misMascotas[noAnimal].earnLife(extraLife);
    } else if (this level == 3) {
        int extraDamage = 6;
        int extraLife = 3;
        misMascotas[noAnimal].earnDamage(extraDamage);
        misMascotas[noAnimal].earnLife(extraLife);
    }
}
}

```

```

//fusionar animal
public void fusion(Mascota misMascotas[], int
animalAFusionar) {
    if (misMascotas[this posicion] ==
misMascotas[animalAFusionar]) {
        boolean fusion = true;
        Hormiga.super.subirXp(fusion);
    } else {
        boolean fusion = false;
        Hormiga.super.subirXp(fusion);
    }
}

```

```
}
```

```
public void receiveDamage(int damageCaused, Mascota[]  
misMascotas) {  
    vidaEnPelea -= damageCaused;  
    if (vidaEnPelea <= 0) {  
        habilidad(misMascotas);  
    }  
}
```

Mosquito:

```
public class Mosquito extends Mascota{  
    private String type="Volador";  
    protected String nombre="Mosquito";
```

```
public String getNombre() {  
    return nombre;  
}
```

```
public Mosquito (int damage, int life, int tier) {  
    this.life = life;  
    this.damage = damage;  
    this.tier = tier;  
}
```

```
//definir posicion
```

```
public void setPosicion(int posicion, Mascota misMascotas[])  
{  
    this.posicion = posicion;  
    Mosquito mosquito = new Mosquito(2, 2, 1);  
    misMascotas[posicion] = mosquito;  
}
```

```
//definir Habilidad
```

```
public void habilidad(Mascota mascotasEnemigas[]){  
    int nerfAnimal = 0;  
    int nerfAnimal2 = 0;  
    int nerfAnimal3 = 0;
```

```
giveAbility(mascotasEnemigas,definirDarHabilidad(nerfAnimal,masc  
otasEnemigas)).
```

```
definirDarHabilidad(nerfAnimal2,mascotasEnemigas),definirDarHabi  
lidad(nerfAnimal3,mascotasEnemigas));
```

```
}
```

```
//definir lo que hara su habilidad
```

```
public void giveAbility(Mascota[] mascotasEnemigas int  
nerfAnimal, int nerfAnimal2, int nerfAnimal3){
```

```

        if (this.level == 1) {
            int Damage = 1;
            mascotasEnemigas[nerfAnimal].damageCaused(Damage);
        } else if (this.level == 2) {
            int Damage = 1;
            mascotasEnemigas[nerfAnimal].damageCaused(Damage);
            mascotasEnemigas[nerfAnimal2].damageCaused(Damage);
        } else if (this.level == 3) {
            int Damage = 1;
            mascotasEnemigas[nerfAnimal].damageCaused(Damage);
            mascotasEnemigas[nerfAnimal2].damageCaused(Damage);
            mascotasEnemigas[nerfAnimal3].damageCaused(Damage);
        }
    }
}

```

```

    public int definirDarHabilidad(int nerfAnimal, Mascota[]
mascotasEnemigas) {

```

```

        while (nerfAnimal == this.posicion &&
mascotasEnemigas[nerfAnimal] == null) {
            nerfAnimal = (int) (Math.random() * (5 - 0) + 0);
        }
        return nerfAnimal;
    }
}

```

```

//fusionar animal
    public void fusion(Mascota misMascotas[], int
animalAFusionar) {
        if (misMascotas[this.posicion] ==
misMascotas[animalAFusionar]) {
            boolean fusion = true;
            Mosquito.super.subirXp(fusion);
        } else {
            boolean fusion = false;
            Mosquito.super.subirXp(fusion);
        }
    }

}

}

```



Caballo:

```
public class Caballo extends Mascota {

    private String[] type = new String[2];
    protected String nombre="Caballo";

    public String getNombre() {
        return nombre;
    }

    //definir el tipo de la mascota
    public void definirType() {
        this.type[0] = "Domestico";
        this.type[1] = "Mamifero";
    }

    public Caballo(int damage, int life, int tier) {
        this.life = life;
        this.damage = damage;
        this.tier = tier;
    }

    //definir posicion
    public void setPosicion(int posicion, Mascota misMascotas[])
    {
        this.posicion = posicion;
        Caballo caballo = new Caballo(2, 1, 1);
        misMascotas[this.posicion] = caballo;
    }

    //definir su habilidad
    public void habilidad(Mascota misMascotas[], int
posicionDeAnimalInvocado) {
        giveAbility(misMascotas, posicionDeAnimalInvocado);
    }

    //definir lo que hara su habilidad
    public void giveAbility(Mascota misMascotas[], int posicion)
    {
        if (level == 1) {
            int extraDamage = 1;
            misMascotas[posicion].earnDamage(extraDamage);
        } else if (level == 2) {
            int extraDamage = 2;
            misMascotas[posicion].earnDamage(extraDamage);
        }
    }
}
```

```

        } else if (level == 3) {
            int extraDamage = 3;
            misMascotas[posicion].earnDamage(extraDamage);
        }
    }

    //fusionar animal
    public void fusion(@NotNull Mascota misMascotas[], int
    animalAFusionar) {
        boolean fusion;
        if (misMascotas[this posicion] ==
        misMascotas[animalAFusionar]) {
            fusion = true;
        } else {
            fusion = false;
        }
        Caballo.super.subirXp(fusion);
    }

```

Nutria:

```

package proyecto1;

public class Nutria extends Mascota {

    private String type = "Mamifero";

    protected String nombre="Nutria";

    public String getNombre() {
        return nombre;
    }

    public Nutria(int damage, int life, int tier) {
        this.life = life;
        this.damage = damage;
        this.tier = tier;
    }

    //definir posicion
    public void setPosicion(int posicion, Mascota misMascotas[])
    {
        this.posicion = posicion;
        Nutria nutria = new Nutria(1, 2, 1);
        misMascotas[this posicion] = nutria;
        habilidad(misMascotas);
    }

    //defini habilidad

```

```

    public void habilidad(Mascota misMascotas[]) {
        int buffAnimal = 0;
        giveAbility(misMascotas, definirDarHabilidad(buffAnimal,
misMascotas));
    }

```

```

    //definir lo que hara la habilidad
    public void giveAbility(Mascota misMascotas[], int
buffAnimal) {
        if (this.level == 1) {
            int extraDamage = 1;
            int extraLife = 1;
            misMascotas[buffAnimal].earnDamage(extraDamage);
            misMascotas[buffAnimal].earnLife(extraLife);
        } else if (this.level == 2) {
            int extraDamage = 2;
            int extraLife = 2;
            misMascotas[buffAnimal].earnDamage(extraDamage);
            misMascotas[buffAnimal].earnLife(extraLife);
        } else if (this.level == 3) {
            int extraDamage = 3;
            int extraLife = 3;
            misMascotas[buffAnimal].earnDamage(extraDamage);
            misMascotas[buffAnimal].earnLife(extraLife);
        }
    }
}

```

```

    public int definirDarHabilidad(int buffAnimal, Mascota[]
misMascotas) {
        while (buffAnimal == this.posicion &&
misMascotas[buffAnimal] == null) {
            buffAnimal = (int) (Math.random() * (5 - 0) + 0);
        }
        return buffAnimal;
    }
}

```

```

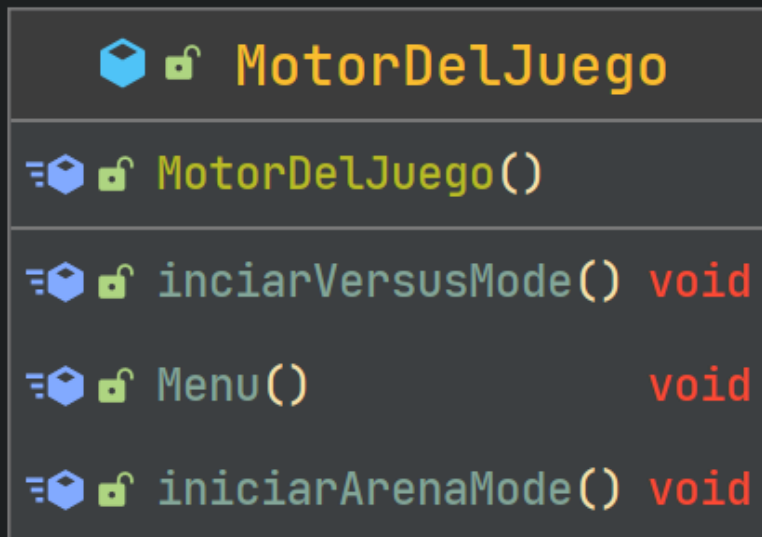
    //fusionar animal
    public void fusion(Mascota misMascotas[], int
animalAFusionar) {
        boolean fusion;
        if (misMascotas[this.posicion] ==
misMascotas[animalAFusionar]) {
            fusion = true;
        } else {
            fusion = false;
        }
    }
}

```

```
Nutria.super.subirXp(fusion);
```

```
}
```

Diagrama UML del motor del juego:



Codigo:

```
public class MotorDelJuego {
    Scanner scanner = new Scanner(System.in);
    protected int opcion = 0;

    public void Menu() {
        while (opcion != 5) {
            System.out.println("\nBienvenido a SuperAutoPets\n");
            System.out.println("Que desea hacer o jugar\n");
            System.out.println("1. Modo Arena:");
            System.out.println("2. Salir");
            opcion=scanner.nextInt();
            if(opcion==1){
                iniciarArenaMode();
            }else if(opcion==2){
                System.out.println("Adios :)");
                break;
            }
        }
    }
}
```

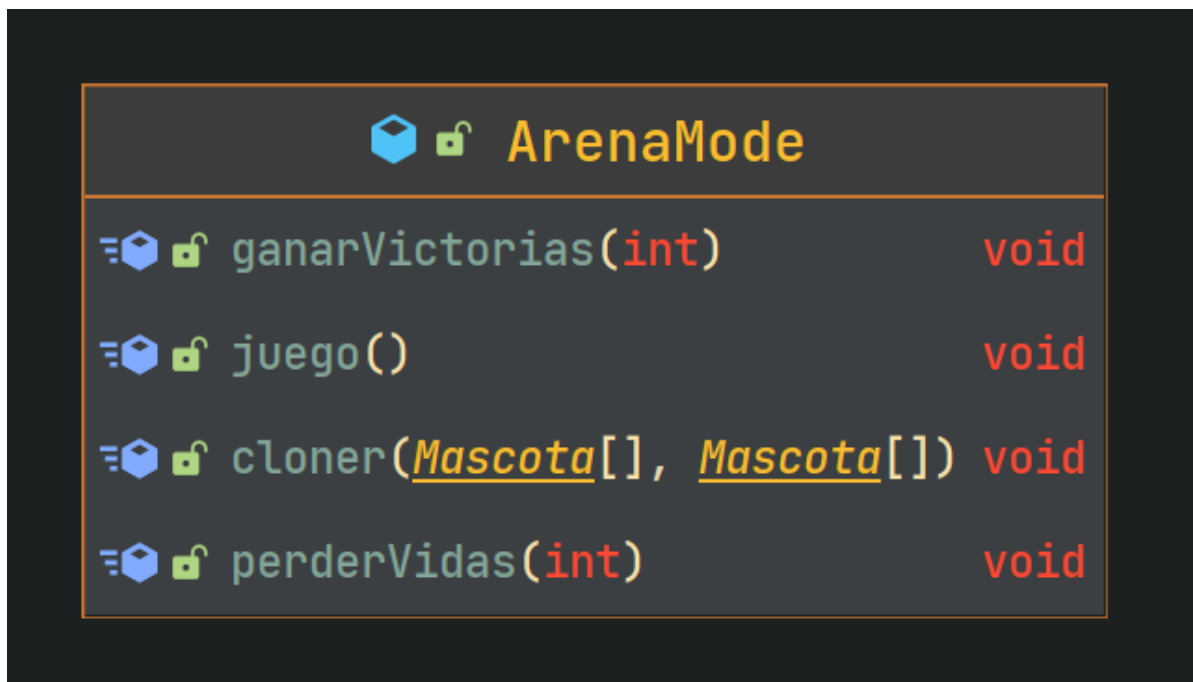
```

    public void iniciarArenaMode() {
        ArenaMode arenaMode = new ArenaMode();
        arenaMode.juego();
    }

    public void inciarVersusMode() {
        VersusMode versusMode = new VersusMode();
    }
}

```

Diagrama UML del modo Arena



Codigo:

```

protected int victorias = 0;
protected int vidas = 10;

public void perderVidas(int cantidadVidas) {
    this.vidas-=cantidadVidas;
}

public void ganarVictorias(int cantidadVictorias) {
    this.victorias+=cantidadVictorias;
}

public void juego() {

```

```

        boolean gano = true;
        int rondas = 1;
        IA ia = new IA();
        Mascota[] misMascotas = new Mascota[5];
        Mascota[] mascotasEnemigas = new Mascota[5];
        EntreBatallas entreBatallas = new EntreBatallas();
        EspacioDeBatalla espacioDeBatallas = new EspacioDeBatalla();
        Mascota[] copia = new Mascota[5];
        while (victorias <= 10 || vidas > 0) {
            ia.generadorDeAnimales(rondas, mascotasEnemigas);

entreBatallas.menuDeCompra(misMascotas, rondas, gano, this.vidas,
this.victorias);
            cloner(copia, misMascotas);
            espacioDeBatallas.pelea(copia, mascotasEnemigas, rondas);
            if(copia[0]==null) {
                perderVidas(espacioDeBatallas.vidasPerdidas(copia,
rondas, gano, mascotasEnemigas));
            }else {

ganarVictorias(espacioDeBatallas.victoriasGanadas(mascotasEnemig
as, gano, copia));
            }
            rondas+=1;
        }
    }

public void cloner(Mascota[] misMascotasCopia, Mascota[]
misMascotas) {
    for (int i = 0; i <= 4; i++) {
        if(misMascotas[i]!=null) {
            misMascotasCopia[i] = misMascotas[i].clone();
        }
    }
}
}

```

## Diagrama UML del la IA



## Codigo

```
public void generadorDeAnimales(int ronda, Mascota[]  
mascotasEnemigas) {  
    if (ronda == 1) {  
        for (int i = 0; i <= 2; i++) {  
            generarAnimales1(animalesAleatorios1(),  
mascotasEnemigas, i);  
        }  
  
    } else if (ronda == 2 || ronda == 3) {  
        for (int i = 0; i <= 4; i++) {  
            int tierAleatorio;  
            tierAleatorio = (int) (Math.random() * (2 - 1) + 1);  
            if (tierAleatorio == 1) {  
                generarAnimales1(animalesAleatorios1(),  
mascotasEnemigas, i);  
            } else {  
                generarAnimales2(animalesAleatorios2(),  
mascotasEnemigas, i);  
            }  
        }  
    }  
}
```

```

    }
}

    } else if (ronda >= 4 && ronda < 6) {
        for (int i = 0; i <= 4; i++) {
            int tierAleatorio;
            tierAleatorio = (int) (Math.random() * (3 - 1) + 1);
            if (tierAleatorio == 1) {
                generarAnimales1(animalesAleatorios1(),
mascotasEnemigas, i);
            } else if (tierAleatorio == 2) {
                generarAnimales2(animalesAleatorios2(),
mascotasEnemigas, i);
            } else {
                generarAnimales3(animalesAleatorios3(),
mascotasEnemigas, i);
            }
        }
    } else {
        for (int i = 0; i <= 4; i++) {
            int tierAleatorio;
            tierAleatorio = (int) (Math.random() * (4 - 1) + 1);
            if (tierAleatorio == 1) {
                generarAnimales1(animalesAleatorios1(),
mascotasEnemigas, i);
            } else if (tierAleatorio == 2) {
                generarAnimales2(animalesAleatorios2(),
mascotasEnemigas, i);
            } else if (tierAleatorio == 3) {
                generarAnimales3(animalesAleatorios3(),
mascotasEnemigas, i);
            } else {
                generarAnimales4(animalesAleatorios4(),
mascotasEnemigas, i);
            }
        }
    }
}

public int animalesAleatorios1() {
    int noAnimal;
    noAnimal = (int) (Math.random() * (7 - 1) + 1);
    return noAnimal;
}

public void generarAnimales1(int noAnimal, Mascota[]
mascotasEnemigas, int posicion) {

```



```

        if (noAnimal == 1) {
            mascotasEnemigas[posicion] = new Hormiga(2, 1, 1);
        } else if (noAnimal == 2) {
            mascotasEnemigas[posicion] = new Pescado(2, 3, 1);
        } else if (noAnimal == 3) {
            mascotasEnemigas[posicion] = new Mosquito(2, 2, 1);
        } else if (noAnimal == 4) {
            mascotasEnemigas[posicion] = new Grillo(1, 2, 1);
        } else if (noAnimal == 5) {
            mascotasEnemigas[posicion] = new Castor(2, 2, 1);
        } else if (noAnimal == 6) {
            mascotasEnemigas[posicion] = new Caballo(2, 1, 1);
        } else if (noAnimal == 7) {
            mascotasEnemigas[posicion] = new Nutria(1, 2, 1);
            ((Nutria)
mascotasEnemigas[posicion]).habilidad(mascotasEnemigas);
        }
    }
}

```

```

public int animalesAleatorios2() {
    int noAnimal;
    noAnimal = (int) (Math.random() * (16 - 9) + 9);
    return noAnimal;
}

```

```

public void generarAnimales2(int noAnimal, Mascota[]
mascotasEnemigas, int posicion) {
    if (noAnimal == 9) {
        mascotasEnemigas[posicion] = new Sapo(3, 3, 2);
        mascotasEnemigas[posicion].setPosicion(posicion);
        ((Sapo)
mascotasEnemigas[posicion]).habilidad(mascotasEnemigas);
    } else if (noAnimal == 10) {
        mascotasEnemigas[posicion] = new Dodo(2, 3, 2);
        mascotasEnemigas[posicion].setPosicion(posicion);
    } else if (noAnimal == 11) {
        mascotasEnemigas[posicion] = new Elefante(3, 5, 2);
        mascotasEnemigas[posicion].setPosicion(posicion);
    } else if (noAnimal == 12) {
        mascotasEnemigas[posicion] = new PuercoEspin(3, 2, 2);
        mascotasEnemigas[posicion].setPosicion(posicion);
    } else if (noAnimal == 13) {
        mascotasEnemigas[posicion] = new Pavoreal(2, 5, 2);
        mascotasEnemigas[posicion].setPosicion(posicion);
    } else if (noAnimal == 14) {
        mascotasEnemigas[posicion] = new Rata(4, 5, 2);
        mascotasEnemigas[posicion].setPosicion(posicion);
    } else if (noAnimal == 15) {

```

```

        mascotasEnemigas[posicion] = new Zorro(5, 2, 2);
        mascotasEnemigas[posicion].setPosicion(posicion);
    } else if (noAnimal == 16) {
        mascotasEnemigas[posicion] = new Arania(2, 2, 2);
        mascotasEnemigas[posicion].setPosicion(posicion);
    }
}

```

```

public int animalesAleatorios3() {
    int noAnimal;
    noAnimal = (int) (Math.random() * (26 - 17) + 17);
    return noAnimal;
}

```

```

public void generarAnimales3(int noAnimal, Mascota[]
mascotasEnemigas, int posicion) {
    if (noAnimal == 17) {
        mascotasEnemigas[posicion] = new Camello(2, 5, 3);
        mascotasEnemigas[posicion].setPosicion(posicion);
    } else if (noAnimal == 18) {
        mascotasEnemigas[posicion] = new Mapache(5, 4, 3);
        mascotasEnemigas[posicion].setPosicion(posicion);
    } else if (noAnimal == 19) {
        mascotasEnemigas[posicion] = new Jirafa(2, 5, 3);
        mascotasEnemigas[posicion].setPosicion(posicion);
    } else if (noAnimal == 20) {
        mascotasEnemigas[posicion] = new Tortuga(1, 2, 3);
        mascotasEnemigas[posicion].setPosicion(posicion);
    } else if (noAnimal == 21) {
        mascotasEnemigas[posicion] = new Caracol(2, 2, 3);
        mascotasEnemigas[posicion].setPosicion(posicion);
        boolean gano = false;
        ((Caracol)
mascotasEnemigas[posicion]).habilidad(mascotasEnemigas, gano);
    } else if (noAnimal == 22) {
        mascotasEnemigas[posicion] = new Oveja(2, 2, 3);
        mascotasEnemigas[posicion].setPosicion(posicion);
    } else if (noAnimal == 23) {
        mascotasEnemigas[posicion] = new Conejo(3, 2, 3);
        mascotasEnemigas[posicion].setPosicion(posicion);
    } else if (noAnimal == 24) {
        mascotasEnemigas[posicion] = new Buey(1, 4, 3);
        mascotasEnemigas[posicion].setPosicion(posicion);
    } else if (noAnimal == 25) {
        mascotasEnemigas[posicion] = new Canguro(1, 2, 3);
        mascotasEnemigas[posicion].setPosicion(posicion);
    } else if (noAnimal == 26) {
        mascotasEnemigas[posicion] = new Buho(5, 3, 3);
    }
}

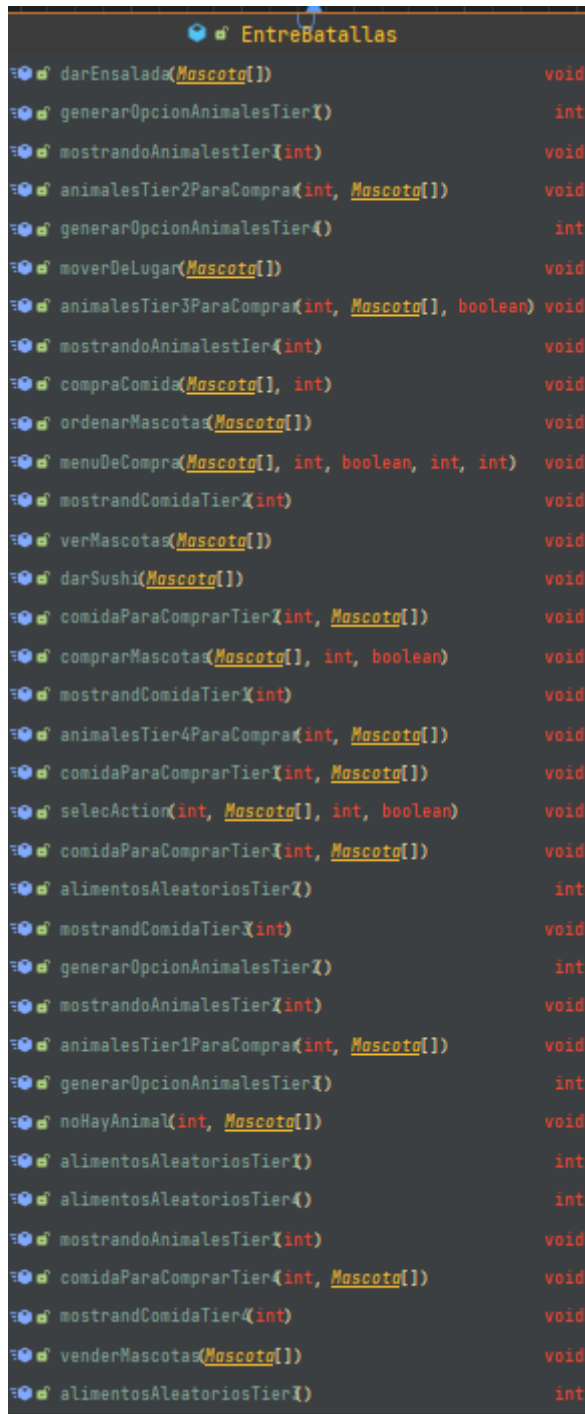
```

```
        mascotasEnemigas[posicion].setPosicion(posicion);  
    }  
}
```

```
public int animalesAleatorios4() {  
    int noAnimal;  
    noAnimal = (int) (Math.random() * (31 - 27) + 27);  
    return noAnimal;  
}
```

```
public void generarAnimales4(int noAnimal, Mascota[]  
mascotasEnemigas, int posicion) {  
    if (noAnimal == 27) {  
        mascotasEnemigas[posicion] = new Venado(1, 1, 4);  
        mascotasEnemigas[posicion].setPosicion(posicion);  
    } else if (noAnimal == 28) {  
        mascotasEnemigas[posicion] = new Hipopotamo(4, 7, 4);  
        mascotasEnemigas[posicion].setPosicion(posicion);  
    } else if (noAnimal == 29) {  
        mascotasEnemigas[posicion] = new Delfin(4, 6, 4);  
        mascotasEnemigas[posicion].setPosicion(posicion);  
    } else if (noAnimal == 30) {  
        mascotasEnemigas[posicion] = new Puma(3, 7, 4);  
        mascotasEnemigas[posicion].setPosicion(posicion);  
    } else if (noAnimal == 31) {  
        mascotasEnemigas[posicion] = new Llama(3, 6, 4);  
        mascotasEnemigas[posicion].setPosicion(posicion);  
        ((Llama)  
mascotasEnemigas[posicion]).habilidad(mascotasEnemigas);  
    }  
}
```

## Diagrama UML del espacio entre batallas



Codigo:

```
public class EntreBatallas extends ArenaMode {
    protected int tier;
    protected int oro;

    public void menuDeCompra(Mascota[] misMascotas, int rondas,
boolean gano, int vidas, int victorias) {
        int opcion = 0;
        oro = 10;
        while (opcion!=5) {
            System.out.println("\nBienvenido\n");
            System.out.println("Tienes " + oro + " de oro");
            System.out.println("Tienes " + vidas + " vidas");
            System.out.println("Tienes " + victorias + "
victorias\n");
            System.out.println("Que deseas hacer?");
            System.out.println("0. Ver a mis mascotas");
            System.out.println("1. Comprar Mascotas");
            System.out.println("2. Comprar Comida");
            System.out.println("3. Ordenar Mascotas");
            System.out.println("4. Vender Mascotas");
            System.out.println("5. Finalizar Compra y entrar en
batalla");
            Scanner scanner = new Scanner(System.in);
            opcion = scanner.nextInt();
            if(opcion==5){
                break;
            }
            selecAction(opcion, misMascotas, rondas, gano);
        }
    }

    // encargado de ver que eleccion se eligio y llamarlas
    public void selecAction(int opcion, Mascota[] misMascotas,
int rondas, boolean gano) {
        if (opcion == 0) {
            verMascotas(misMascotas);
        } else if (opcion == 1) {
            comprarMascotas(misMascotas, rondas, gano);
        } else if (opcion == 2) {
            compraComida(misMascotas, rondas);
        } else if (opcion == 3) {
            ordenarMascotas(misMascotas);
        } else if (opcion == 4) {
            venderMascotas(misMascotas);
        }
    }
}
```

```
    }
```

```
    }
```

```
    //este sera el encargado de comprar las mascotas  
    //En el generara cierta cantidad de mascotas dependiendo la  
    ronda
```

```
    //Y te dara la opcion de comprar o salir del menu de compra
```

```
    public void comprarMascotas(Mascota[] misMascotas, int  
    rondas, boolean gano) {
```

```
        if (rondas == 1) {
```

```
            for (int i = 0; i <= 2; i++) {
```

```
mostrandoAnimalesTier1(generarOpcionAnimalesTier1());
```

```
            }
```

```
            Scanner scanner = new Scanner(System.in);
```

```
            boolean seguirComprando = true;
```

```
            while (seguirComprando) {
```

```
                System.out.println("Digite el numero del animal  
que desea comprar");
```

```
                System.out.println("Si quiere salir presione 0  
");
```

```
                int noAnimal = scanner.nextInt();
```

```
                if (noAnimal == 0) {
```

```
                    break;
```

```
                }
```

```
                animalesTier1ParaComprar(noAnimal, misMascotas);
```

```
            }
```

```
        } else if (rondas == 2 || rondas == 3) {
```

```
            for (int i = 0; i <= 2; i++) {
```

```
                int tierAleatorio = (int) (Math.random() * (2 -  
1) + 1);
```

```
                if (tierAleatorio == 1) {
```

```
mostrandoAnimalesTier1(generarOpcionAnimalesTier1());
```

```
                } else if (tierAleatorio == 2) {
```

```
mostrandoAnimalesTier2(generarOpcionAnimalesTier2());
```

```
            }
```

```
        }
```

```
        boolean seguirComprando = true;
```

```
        while (seguirComprando) {
```

```
            Scanner scanner = new Scanner(System.in);
```

```
            System.out.println("Digite el numero del animal  
que desea comprar");
```

```

        System.out.println("Si quiere salir presione 0
");
        int noAnimal = scanner.nextInt();
        if (noAnimal == 0) {
            break;
        }
        if (noAnimal >= 1 && noAnimal <= 8) {
            animalesTier1ParaComprar(noAnimal,
misMascotas);
        } else if (noAnimal >= 9 && noAnimal <= 16) {
            animalesTier2ParaComprar(noAnimal,
misMascotas);
        }
    }
} else if (rondas >= 4 && rondas < 6) {
    for (int i = 0; i <= 3; i++) {
        int tierAleatorio = (int) (Math.random() * (3 -
1) + 1);
        if (tierAleatorio == 1) {

mostrandoAnimalesTier1(generarOpcionAnimalesTier1());
        } else if (tierAleatorio == 2) {

mostrandoAnimalesTier2(generarOpcionAnimalesTier2());
        } else if (tierAleatorio == 3) {

mostrandoAnimalesTier3(generarOpcionAnimalesTier3());
        }

    }

}

boolean seguirComprando = true;
while (seguirComprando) {
    Scanner scanner = new Scanner(System.in);
    System.out.println("Digite el numero del animal
que desea comprar");
    System.out.println("Si quiere salir presione 0
");
    int noAnimal = scanner.nextInt();
    if (noAnimal == 0) {
        break;
    }
    if (noAnimal >= 1 && noAnimal <= 8) {
        animalesTier1ParaComprar(noAnimal,
misMascotas);
    } else if (noAnimal >= 9 && noAnimal <= 16) {
        animalesTier2ParaComprar(noAnimal,
misMascotas);
    }
}

```

```

        } else if (noAnimal >= 17 && noAnimal <= 26) {
            animalesTier3ParaComprar(noAnimal,
misMascotas, gano);
        }
    }

    } else if (rondas >= 6) {
        for (int i = 0; i <= 3; i++) {
            int tierAleatorio = (int) (Math.random() * (4 -
1) + 1);
            if (tierAleatorio == 1) {

mostrandoAnimalesTier1(generarOpcionAnimalesTier1());
            } else if (tierAleatorio == 2) {

mostrandoAnimalesTier2(generarOpcionAnimalesTier2());
            } else if (tierAleatorio == 3) {

mostrandoAnimalesTier3(generarOpcionAnimalesTier3());
            } else if (tierAleatorio == 4) {

mostrandoAnimalesTier4(generarOpcionAnimalesTier4());
            }

        }
        boolean seguirComprando = true;
        while (seguirComprando) {
            Scanner scanner = new Scanner(System.in);
            System.out.println("Digite el numero del animal
que desea comprar");
            System.out.println("Si quiere salir presione 0
");
            int noAnimal = scanner.nextInt();
            if (noAnimal == 0) {
                break;
            }
            if (noAnimal >= 1 && noAnimal <= 8) {
                animalesTier1ParaComprar(noAnimal,
misMascotas);
            } else if (noAnimal >= 9 && noAnimal <= 16) {
                animalesTier2ParaComprar(noAnimal,
misMascotas);
            } else if (noAnimal >= 17 && noAnimal <= 26) {
                animalesTier3ParaComprar(noAnimal,
misMascotas, gano);
            } else if (noAnimal >= 27 && noAnimal <= 31) {
                animalesTier4ParaComprar(noAnimal,
misMascotas);
            }
        }
    }
}

```



}

```
//encargado de generar aleatoriamente numeros para usar de
referencia a que animal comprar de tier 1
public int generarOpcionAnimalesTier1() {
    int animalGenerado;
    animalGenerado = (int) (Math.random() * (7 - 1) + 1);
    return animalGenerado;
}
```

```
//Este sera el encargado de mostrar las opciones(dependiendo
generarOpcionAnimales()) de animales de tier 1
public void mostrandoAnimalesTier1(int noAnimal) {
    if (noAnimal == 1) {
        System.out.println("1. Hormiga");
    } else if (noAnimal == 2) {
        System.out.println("2. Pescado");
    } else if (noAnimal == 3) {
        System.out.println("3. Mosquito");
    } else if (noAnimal == 4) {
        System.out.println("4. Grillo");
    } else if (noAnimal == 5) {
        System.out.println("5. Castor");
    } else if (noAnimal == 6) {
        System.out.println("6. Caballo");
    } else if (noAnimal == 7) {
        System.out.println("7. Nutria");
    }
}
```

```
//este sera el encargado de crear al animal y posicionarlo en
la posicion que el jugador decida
public void animalesTier1ParaComprar(int noAnimal, Mascota[]
misMascotas) {
    if (noAnimal == 1) {
        if (oro >= 3) {
            Scanner scan = new Scanner(System.in);
            int posicion;
            System.out.println("En que posicion lo quiere
poner");
            posicion = scan.nextInt();
            if (posicion <= 4 && posicion >= 0) {
                if (misMascotas[posicion] == null) {
```

```

        misMascotas[posicion] = new Hormiga(2, 1,
1);

misMascotas[posicion].setPosicion(posicion);
        oro -= 3;
        for(int i =0;i<=4;i++){
            if(misMascotas[i] instanceof Caballo){
                ((Caballo)
misMascotas[i]).habilidad(misMascotas,posicion);
            }
        }
    } else {
        System.out.println("Ya hay un animal en
ese lugar ");
    }
    } else {
        System.out.println("La posicion tiene que
estar entre 0 y 4");
    }
    } else {
        System.out.println("No tienes el suficiente
dinero ");
    }

    } else if (noAnimal == 2) {
        if (oro >= 3) {
            Scanner scan = new Scanner(System.in);
            int posicion;
            System.out.println("En que posicion lo quiere
poner");
            posicion = scan.nextInt();
            if (posicion <= 4 && posicion >= 0) {
                if (misMascotas[posicion] == null) {
                    misMascotas[posicion] = new Pescado(2, 3,
1);

misMascotas[posicion].setPosicion(posicion);
                oro -= 3;
                for(int i =0;i<=4;i++){
                    if(misMascotas[i] instanceof Caballo){
                        ((Caballo)
misMascotas[i]).habilidad(misMascotas,posicion);
                    }
                }
            } else {
                System.out.println("Ya hay un animal en
ese lugar ");
            }
        }
    }
}

```

```

        } else {
            System.out.println("La posicion tiene que
estar entre 0 y 4");
        }
    } else {
        System.out.println("No tienes el suficiente
dinero ");
    }

    } else if (noAnimal == 3) {
        if (oro >= 3) {
            Scanner scan = new Scanner(System.in);
            int posicion;
            System.out.println("En que posicion lo quiere
poner");
            posicion = scan.nextInt();
            if (posicion <= 4 && posicion >= 0) {
                if (misMascotas[posicion] == null) {
                    misMascotas[posicion] = new Mosquito(2,
2, 1);

misMascotas[posicion].setPosicion(posicion);
                    oro -= 3;
                    for(int i =0;i<=4;i++){
                        if(misMascotas[i] instanceof Caballo){
                            ((Caballo)
misMascotas[i]).habilidad(misMascotas,posicion);
                        }
                    }
                } else {
                    System.out.println("Ya hay un animal en
ese lugar ");
                }
            } else {
                System.out.println("La posicion tiene que
estar entre 0 y 4");
            }
        } else {
            System.out.println("No tienes el suficiente
dinero ");
        }

    }

    } else if (noAnimal == 4) {
        if (oro >= 3) {
            Scanner scan = new Scanner(System.in);
            int posicion;
            System.out.println("En que posicion lo quiere
poner");

```

```

        posicion = scan.nextInt();
        if (posicion <= 4 && posicion >= 0) {
            if (misMascotas[posicion] == null) {
                misMascotas[posicion] = new Grillo(1, 2,
1);

misMascotas[posicion].setPosicion(posicion);
                oro -= 3;
                for(int i =0;i<=4;i++){
                    if(misMascotas[i]instanceof Caballo){
                        ((Caballo)
misMascotas[i]).habilidad(misMascotas,posicion);
                    }
                }
            } else {
                System.out.println("Ya hay un animal en
ese lugar ");
            }
        } else {
            System.out.println("La posicion tiene que
estar entre 0 y 4");
        }
    } else {
        System.out.println("No tienes el suficiente
dinero ");
    }

} else if (noAnimal == 5) {
    if (oro >= 3) {
        Scanner scan = new Scanner(System.in);
        int posicion;
        System.out.println("En que posicion lo quiere
poner");
        posicion = scan.nextInt();
        if (posicion <= 4 && posicion >= 0) {
            if (misMascotas[posicion] == null) {
                misMascotas[posicion] = new Castor(2, 2,
1);

misMascotas[posicion].setPosicion(posicion);
                oro -= 3;
                for(int i =0;i<=4;i++){
                    if(misMascotas[i]instanceof Caballo){
                        ((Caballo)
misMascotas[i]).habilidad(misMascotas,posicion);
                    }
                }
            } else {

```

```

        System.out.println("Ya hay un animal en
ese lugar ");
    }
    } else {
        System.out.println("La posicion tiene que
estar entre 0 y 4");
    }
    } else {
        System.out.println("No tienes el suficiente
dinero ");
    }

    } else if (noAnimal == 6) {
        if (oro >= 3) {
            Scanner scan = new Scanner(System.in);
            int posicion;
            System.out.println("En que posicion lo quiere
poner");
            posicion = scan.nextInt();
            if (posicion <= 4 && posicion >= 0) {
                if (misMascotas[posicion] == null) {
                    misMascotas[posicion] = new Caballo(2, 1,
1);

misMascotas[posicion].setPosicion(posicion);
                    oro -= 3;
                    for(int i =0;i<=4;i++){
                        if(misMascotas[i] instanceof Caballo){
                            ((Caballo)
misMascotas[i]).habilidad(misMascotas, posicion);
                        }
                    }
                } else {
                    System.out.println("Ya hay un animal en
ese lugar ");
                }
            } else {
                System.out.println("La posicion tiene que
estar entre 0 y 4");
            }
        } else {
            System.out.println("No tienes el suficiente
dinero ");
        }

    } else if (noAnimal == 7) {
        if (oro >= 3) {
            Scanner scan = new Scanner(System.in);

```

```

        int posicion;
        System.out.println("En que posicion lo quiere
poner");
        posicion = scan.nextInt();
        if (posicion <= 4 && posicion >= 0) {
            if (misMascotas[posicion] == null) {
                misMascotas[posicion] = new Nutria(1, 2,
1);

misMascotas[posicion].setPosicion(posicion);
                ((Nutria)
misMascotas[posicion]).habilidad(misMascotas);
                oro -= 3;
                for(int i =0;i<=4;i++){
                    if(misMascotas[i] instanceof Caballo){
                        ((Caballo)
misMascotas[i]).habilidad(misMascotas,posicion);
                    }
                }
            } else {
                System.out.println("Ya hay un animal en
ese lugar ");
            }
        } else {
            System.out.println("La posicion tiene que
estar entre 0 y 4");
        }
    } else {
        System.out.println("No tienes el suficiente
dinero ");
    }

}
}

```

```

//encargado de generar aleatoriamente numeros para usar de
referencia a que animal comprar de tier 2
public int generarOpcionAnimalesTier2() {
    int animalGenerado;
    animalGenerado = (int) (Math.random() * (16 - 9) + 9);
    return animalGenerado;
}

```

```

/////Este sera el encargado de mostrar las
opciones(dependiendo generarOpcionAnimales()) de animales de
tier 2
public void mostrandoAnimalesTier2(int noAnimal) {
    if (noAnimal == 9) {

```

```

        System.out.println("9. Sapo");
    } else if (noAnimal == 10) {
        System.out.println("10. Dodo");
    } else if (noAnimal == 11) {
        System.out.println("11. Elefante");
    } else if (noAnimal == 12) {
        System.out.println("12. Puero Espin");
    } else if (noAnimal == 13) {
        System.out.println("13. Pavo Real");
    } else if (noAnimal == 14) {
        System.out.println("14. Rata");
    } else if (noAnimal == 15) {
        System.out.println("15. Zorro");
    } else if (noAnimal == 16) {
        System.out.println("16. Araña");
    }
}
}

```

*//este es el encargado de crear y posicionar al animal en la posicion que el jugador decida*

```

public void animalesTier2ParaComprar(int noAnimal, Mascota[]
misMascotas) {
    if (noAnimal == 9) {
        if (oro >= 3) {
            Scanner scan = new Scanner(System.in);
            int posicion;
            System.out.println("En que posicion lo quiere
poner");
            posicion = scan.nextInt();
            if (posicion <= 4 && posicion >= 0) {
                if (misMascotas[posicion] == null) {
                    misMascotas[posicion] = new Sapo(3, 3,
2);
misMascotas[posicion].setPosicion(posicion);
                    ((Sapo)
misMascotas[posicion]).habilidad(misMascotas);
                    oro -= 3;
                    for(int i =0;i<=4;i++){
                        if(misMascotas[i] instanceof Caballo){
                            ((Caballo)
misMascotas[i]).habilidad(misMascotas,posicion);
                        }
                    }
                } else {
                    System.out.println("Ya hay un animal en
ese lugar ");
                }
            }
        }
    }
}

```

```

        } else {
            System.out.println("La posicion tiene que
estar entre 0 y 4");
        }
    } else {
        System.out.println("No tienes el suficiente
dinero ");
    }

} else if (noAnimal == 10) {
    if (oro >= 3) {
        Scanner scan = new Scanner(System.in);
        int posicion;
        System.out.println("En que posicion lo quiere
poner");
        posicion = scan.nextInt();
        if (posicion <= 4 && posicion >= 0) {
            if (misMascotas[posicion] == null) {
                misMascotas[posicion] = new Dodo(2, 3,
2);
misMascotas[posicion].setPosicion(posicion);
                oro -= 3;
                for(int i =0;i<=4;i++){
                    if(misMascotas[i] instanceof Caballo){
                        ((Caballo)
misMascotas[i]).habilidad(misMascotas,posicion);
                    }
                }
            } else {
                System.out.println("Ya hay un animal en
ese lugar ");
            }
        } else {
            System.out.println("La posicion tiene que
estar entre 0 y 4");
        }
    } else {
        System.out.println("No tienes el suficiente
dinero ");
    }

} else if (noAnimal == 11) {
    if (oro >= 3) {
        Scanner scan = new Scanner(System.in);
        int posicion;
        System.out.println("En que posicion lo quiere
poner");

```



```

        posicion = scan.nextInt();
        if (posicion <= 4 && posicion >= 0) {
            if (misMascotas[posicion] == null) {
                misMascotas[posicion] = new Elefante(3,
5, 2);

misMascotas[posicion].setPosicion(posicion);
                oro -= 3;
                for(int i =0;i<=4;i++){
                    if(misMascotas[i]instanceof Caballo){
                        ((Caballo)
misMascotas[i]).habilidad(misMascotas,posicion);
                    }
                }
            } else {
                System.out.println("Ya hay un animal en
ese lugar ");
            }
        } else {
            System.out.println("La posicion tiene que
estar entre 0 y 4");
        }
    } else {
        System.out.println("No tienes el suficiente
dinero ");
    }

} else if (noAnimal == 12) {
    if (oro >= 3) {
        Scanner scan = new Scanner(System.in);
        int posicion;
        System.out.println("En que posicion lo quiere
poner");
        posicion = scan.nextInt();
        if (posicion <= 4 && posicion >= 0) {
            if (misMascotas[posicion] == null) {
                misMascotas[posicion] = new
PuercoEspin(3, 2, 2);

misMascotas[posicion].setPosicion(posicion);
                oro -= 3;
                for(int i =0;i<=4;i++){
                    if(misMascotas[i]instanceof Caballo){
                        ((Caballo)
misMascotas[i]).habilidad(misMascotas,posicion);
                    }
                }
            } else {

```

```

        System.out.println("Ya hay un animal en
ese lugar ");
    }
    } else {
        System.out.println("La posicion tiene que
estar entre 0 y 4");
    }
    } else {
        System.out.println("No tienes el suficiente
dinero ");
    }
    } else if (noAnimal == 13) {
        if (oro >= 3) {
            Scanner scan = new Scanner(System.in);
            int posicion;
            System.out.println("En que posicion lo quiere
poner");
            posicion = scan.nextInt();
            if (posicion <= 4 && posicion >= 0) {
                if (misMascotas[posicion] == null) {
                    misMascotas[posicion] = new Pavoreal(2,
5, 2);

misMascotas[posicion].setPosicion(posicion);
                    oro -= 3;
                } else {
                    System.out.println("Ya hay un animal en
ese lugar ");
                }
            } else {
                System.out.println("La posicion tiene que
estar entre 0 y 4");
            }
        } else {
            System.out.println("No tienes el suficiente
dinero ");
        }
    }

    } else if (noAnimal == 14) {
        if (oro >= 3) {
            Scanner scan = new Scanner(System.in);
            int posicion;
            System.out.println("En que posicion lo quiere
poner");
            posicion = scan.nextInt();
            if (posicion <= 4 && posicion >= 0) {
                if (misMascotas[posicion] == null) {

```

```

        misMascotas[posicion] = new Rata(4, 5,
2);

misMascotas[posicion].setPosicion(posicion);
        oro -= 3;
        for(int i =0;i<=4;i++){
            if(misMascotas[i] instanceof Caballo){
                ((Caballo)
misMascotas[i]).habilidad(misMascotas,posicion);
            }
        }
    } else {
        System.out.println("Ya hay un animal en
ese lugar ");
    }
} else {
    System.out.println("La posicion tiene que
estar entre 0 y 4");
}
} else {
    System.out.println("No tienes el suficiente
dinero ");
}

    } else if (noAnimal == 15) {
        if (oro >= 3) {
            Scanner scan = new Scanner(System.in);
            int posicion;
            System.out.println("En que posicion lo quiere
poner");
            posicion = scan.nextInt();
            if (posicion <= 4 && posicion >= 0) {
                if (misMascotas[posicion] == null) {
                    misMascotas[posicion] = new Zorro(5, 2,
2);

misMascotas[posicion].setPosicion(posicion);
                    oro -= 3;
                    for(int i =0;i<=4;i++){
                        if(misMascotas[i] instanceof Caballo){
                            ((Caballo)
misMascotas[i]).habilidad(misMascotas,posicion);
                        }
                    }
                } else {
                    System.out.println("Ya hay un animal en
ese lugar ");
                }
            }
        }
    }
}

```

```

        } else {
            System.out.println("La posicion tiene que
estar entre 0 y 4");
        }
    } else {
        System.out.println("No tienes el suficiente
dinero ");
    }
} else if (noAnimal == 16) {
    if (oro >= 3) {
        Scanner scan = new Scanner(System.in);
        int posicion;
        System.out.println("En que posicion lo quiere
poner");
        posicion = scan.nextInt();
        if (posicion <= 4 && posicion >= 0) {
            if (misMascotas[posicion] == null) {
                misMascotas[posicion] = new Arania(2, 2,
2);

misMascotas[posicion].setPosicion(posicion);
                oro -= 3;
                for(int i =0;i<=4;i++){
                    if(misMascotas[i] instanceof Caballo){
                        ((Caballo)
misMascotas[i]).habilidad(misMascotas, posicion);
                    }
                }
            } else {
                System.out.println("Ya hay un animal en
ese lugar ");
            }
        } else {
            System.out.println("La posicion tiene que
estar entre 0 y 4");
        }
    } else {
        System.out.println("No tienes el suficiente
dinero ");
    }
}
}
}

//encargado de generar aleatoriamente numeros para usar de
referencia a que animal comprar de tier 3
public int generarOpcionAnimalesTier3() {
    int animalGenerado;

```

```

        animalGenerado = (int) (Math.random() * (26 - 17) + 17);
        return animalGenerado;
    }
}

```

*//Este sera el encargado de mostrar las opciones(dependiendo  
generarOpcionAnimales()) de animales de tier 3*

```

    public void mostrandoAnimalestIer3(int noAnimal) {
        if (noAnimal == 17) {
            System.out.println("17. Camello");
        } else if (noAnimal == 18) {
            System.out.println("18. Mapache");
        } else if (noAnimal == 19) {
            System.out.println("19. Jirafa");
        } else if (noAnimal == 20) {
            System.out.println("20. Tortuga");
        } else if (noAnimal == 21) {
            System.out.println("21. Caracol");
        } else if (noAnimal == 22) {
            System.out.println("22. Oveja");
        } else if (noAnimal == 23) {
            System.out.println("23. Conejo");
        } else if (noAnimal == 24) {
            System.out.println("24. Buey");
        } else if (noAnimal == 25) {
            System.out.println("25. Canguro");
        } else if (noAnimal == 26) {
            System.out.println("26. Buho");
        }
    }
}

```

*//este es el encargado de crear y posicionar al animal en  
la posicion que el jugador decida*

```

    public void animalesTier3ParaComprar(int noAnimal, Mascota[]
misMascotas, boolean gano) {
        if (noAnimal == 17) {
            if (oro >= 3) {
                Scanner scan = new Scanner(System.in);
                int posicion;
                System.out.println("En que posicion lo quiere
poner");
                posicion = scan.nextInt();
                if (posicion <= 4 && posicion >= 0) {
                    if (misMascotas[posicion] == null) {
                        misMascotas[posicion] = new Camello(2, 5,
3);

```

```

misMascotas[posicion].setPosicion(posicion);

```

```

        oro -= 3;
        for(int i =0;i<=4;i++){
            if(misMascotas[i] instanceof Caballo){
                ((Caballo)
misMascotas[i]).habilidad(misMascotas,posicion);
            }
        }
    } else {
        System.out.println("Ya hay un animal en
ese lugar ");
    }
    } else {
        System.out.println("La posicion tiene que
estar entre 0 y 4");
    }
    } else {
        System.out.println("No tienes el suficiente
dinero ");
    }
}

    } else if (noAnimal == 18) {
        if (oro >= 3) {
            Scanner scan = new Scanner(System.in);
            int posicion;
            System.out.println("En que posicion lo quiere
poner");
            posicion = scan.nextInt();
            if (posicion <= 4 && posicion >= 0) {
                if (misMascotas[posicion] == null) {
                    misMascotas[posicion] = new Mapache(5, 4,
3);
                }

misMascotas[posicion].setPosicion(posicion);
                oro -= 3;
                for(int i =0;i<=4;i++){
                    if(misMascotas[i] instanceof Caballo){
                        ((Caballo)
misMascotas[i]).habilidad(misMascotas,posicion);
                    }
                }
            } else {
                System.out.println("Ya hay un animal en
ese lugar ");
            }
        } else {
            System.out.println("La posicion tiene que
estar entre 0 y 4");
        }
    }
}

```

```

        } else {
            System.out.println("No tienes el suficiente
dinero ");
        }

    } else if (noAnimal == 19) {
        if (oro >= 3) {
            Scanner scan = new Scanner(System.in);
            int posicion;
            System.out.println("En que posicion lo quiere
poner");
            posicion = scan.nextInt();
            if (posicion <= 4 && posicion >= 0) {
                if (misMascotas[posicion] == null) {
                    misMascotas[posicion] = new Jirafa(2, 5,
3);

misMascotas[posicion].setPosicion(posicion);
                    oro -= 3;
                    for(int i =0;i<=4;i++){
                        if(misMascotas[i] instanceof Caballo){
                            ((Caballo)
misMascotas[i]).habilidad(misMascotas,posicion);
                        }
                    }
                } else {
                    System.out.println("Ya hay un animal en
ese lugar ");
                }
            } else {
                System.out.println("La posicion tiene que
estar entre 0 y 4");
            }
        } else {
            System.out.println("No tienes el suficiente
dinero ");
        }

    } else if (noAnimal == 20) {
        if (oro >= 3) {
            Scanner scan = new Scanner(System.in);
            int posicion;
            System.out.println("En que posicion lo quiere
poner");
            posicion = scan.nextInt();
            if (posicion <= 4 && posicion >= 0) {
                if (misMascotas[posicion] == null) {

```

```

        misMascotas[posicion] = new Tortuga(1, 2,
3);

misMascotas[posicion].setPosicion(posicion);
        oro -= 3;
        for(int i =0;i<=4;i++){
            if(misMascotas[i]instanceof Caballo){
                ((Caballo)
misMascotas[i]).habilidad(misMascotas,posicion);
            }
        }
    } else {
        System.out.println("Ya hay un animal en
ese lugar ");
    }
} else {
        System.out.println("La posicion tiene que
estar entre 0 y 4");
    }
} else {
        System.out.println("No tienes el suficiente
dinero ");
    }

} else if (noAnimal == 21) {
    if (oro >= 3) {
        Scanner scan = new Scanner(System.in);
        int posicion;
        System.out.println("En que posicion lo quiere
poner");
        posicion = scan.nextInt();
        if (posicion <= 4 && posicion >= 0) {
            if (misMascotas[posicion] == null) {
                misMascotas[posicion] = new Caracol(2, 2,
3);

misMascotas[posicion].setPosicion(posicion);
                ((Caracol)
misMascotas[posicion]).habilidad(misMascotas, gano);
                oro -= 3;
                for(int i =0;i<=4;i++){
                    if(misMascotas[i]instanceof Caballo){
                        ((Caballo)
misMascotas[i]).habilidad(misMascotas,posicion);
                    }
                }
            } else {

```



```

        System.out.println("Ya hay un animal en
ese lugar ");
    }
    } else {
        System.out.println("La posicion tiene que
estar entre 0 y 4");
    }
    } else {
        System.out.println("No tienes el suficiente
dinero ");
    }

    } else if (noAnimal == 22) {
        if (oro >= 3) {
            Scanner scan = new Scanner(System.in);
            int posicion;
            System.out.println("En que posicion lo quiere
poner");
            posicion = scan.nextInt();
            if (posicion <= 4 && posicion >= 0) {
                if (misMascotas[posicion] == null) {
                    misMascotas[posicion] = new Oveja(2, 2,
3);

misMascotas[posicion].setPosicion(posicion);
                    oro -= 3;
                    for(int i =0;i<=4;i++){
                        if(misMascotas[i] instanceof Caballo){
                            ((Caballo)
misMascotas[i]).habilidad(misMascotas,posicion);
                        }
                    }
                } else {
                    System.out.println("Ya hay un animal en
ese lugar ");
                }
            } else {
                System.out.println("La posicion tiene que
estar entre 0 y 4");
            }
        } else {
            System.out.println("No tienes el suficiente
dinero ");
        }

    } else if (noAnimal == 23) {
        if (oro >= 3) {
            Scanner scan = new Scanner(System.in);

```

```

        int posicion;
        System.out.println("En que posicion lo quiere
poner");
        posicion = scan.nextInt();
        if (posicion <= 4 && posicion >= 0) {
            if (misMascotas[posicion] == null) {
                misMascotas[posicion] = new Conejo(3, 2,
3);

misMascotas[posicion].setPosicion(posicion);
                oro -= 3;
                for(int i =0;i<=4;i++){
                    if(misMascotas[i]instanceof Caballo){
                        ((Caballo)
misMascotas[i]).habilidad(misMascotas,posicion);
                    }
                }
            } else {
                System.out.println("Ya hay un animal en
ese lugar ");
            }
        } else {
            System.out.println("La posicion tiene que
estar entre 0 y 4");
        }
    } else {
        System.out.println("No tienes el suficiente
dinero ");
    }

} else if (noAnimal == 24) {
    if (oro >= 3) {
        Scanner scan = new Scanner(System.in);
        int posicion;
        System.out.println("En que posicion lo quiere
poner");
        posicion = scan.nextInt();
        if (posicion <= 4 && posicion >= 0) {
            if (misMascotas[posicion] == null) {
                misMascotas[posicion] = new Buey(1, 4,
3);

misMascotas[posicion].setPosicion(posicion);
                oro -= 3;
                for(int i =0;i<=4;i++){
                    if(misMascotas[i]instanceof Caballo){
                        ((Caballo)
misMascotas[i]).habilidad(misMascotas,posicion);

```

```

    }
    }
    } else {
        System.out.println("Ya hay un animal en
ese lugar ");
    }
    } else {
        System.out.println("La posicion tiene que
estar entre 0 y 4");
    }
    } else {
        System.out.println("No tienes el suficiente
dinero ");
    }

    } else if (noAnimal == 25) {
        if (oro >= 3) {
            Scanner scan = new Scanner(System.in);
            int posicion;
            System.out.println("En que posicion lo quiere
poner");
            posicion = scan.nextInt();
            if (posicion <= 4 && posicion >= 0) {
                if (misMascotas[posicion] == null) {
                    misMascotas[posicion] = new Canguro(1, 2,
3);

misMascotas[posicion].setPosicion(posicion);
                    oro -= 3;
                    for(int i =0;i<=4;i++){
                        if(misMascotas[i] instanceof Caballo){
                            ((Caballo)
misMascotas[i]).habilidad(misMascotas,posicion);
                        }
                    }
                } else {
                    System.out.println("Ya hay un animal en
ese lugar ");
                }
            } else {
                System.out.println("La posicion tiene que
estar entre 0 y 4");
            }
        } else {
            System.out.println("No tienes el suficiente
dinero ");
        }
    }
}

```

```

        } else if (noAnimal == 26) {
            if (oro >= 3) {
                Scanner scan = new Scanner(System.in);
                int posicion;
                System.out.println("En que posicion lo quiere
poner");
                posicion = scan.nextInt();
                if (posicion <= 4 && posicion >= 0) {
                    if (misMascotas[posicion] == null) {
                        misMascotas[posicion] = new Buho(5, 3,
3);
misMascotas[posicion].setPosicion(posicion);
                        oro -= 3;
                        for(int i =0;i<=4;i++){
                            if(misMascotas[i] instanceof Caballo){
                                ((Caballo)
misMascotas[i]).habilidad(misMascotas, posicion);
                            }
                        }
                    } else {
                        System.out.println("Ya hay un animal en
ese lugar ");
                    }
                } else {
                    System.out.println("La posicion tiene que
estar entre 0 y 4");
                }
            } else {
                System.out.println("No tienes el suficiente
dinero ");
            }
        }
    }
}

//encargado de generar aleatoriamente numeros para usar de
referencia a que animal comprar de tier 1
public int generarOpcionAnimalesTier4() {
    int animalGenerado;
    animalGenerado = (int) (Math.random() * (31 - 27) + 27);
    return animalGenerado;
}

//Este sera el encargado de mostrar las opciones(dependiendo
generarOpcionAnimales()) de animales de tier 3
public void mostrandoAnimalestIer4(int noAnimal) {
    if (noAnimal == 17) {

```

```

        System.out.println("27. Venado");
    } else if (noAnimal == 18) {
        System.out.println("28. Hipopotamo");
    } else if (noAnimal == 19) {
        System.out.println("29. Delfin");
    } else if (noAnimal == 20) {
        System.out.println("30. Puma");
    } else if (noAnimal == 21) {
        System.out.println("31. Llama");
    }
}
}

```

*//este sera el encargado de crear y posicionar al animal en la posicion que el jugador decida*

```

public void animalesTier4ParaComprar(int noAnimal, Mascota[]
misMascotas) {
    if (noAnimal == 27) {
        if (oro >= 3) {
            Scanner scan = new Scanner(System.in);
            int posicion;
            System.out.println("En que posicion lo quiere
poner");
            posicion = scan.nextInt();
            if (posicion <= 4 && posicion >= 0) {
                if (misMascotas[posicion] == null) {
                    misMascotas[posicion] = new Venado(1, 1,
4);
misMascotas[posicion].setPosicion(posicion);
                    oro -= 3;
                    for (int i = 0; i <= 4; i++) {
                        if (misMascotas[i] instanceof
Caballo) {
                            ((Caballo)
misMascotas[i]).habilidad(misMascotas, posicion);
                        }
                    }
                } else {
                    System.out.println("Ya hay un animal en
ese lugar ");
                }
            } else {
                System.out.println("La posicion tiene que
estar entre 0 y 4");
            }
        } else {
            System.out.println("No tienes el suficiente
dinero ");

```

```

    }
    } else if (noAnimal == 28) {
        if (oro >= 3) {
            Scanner scan = new Scanner(System.in);
            int posicion;
            System.out.println("En que posicion lo quiere
poner");
            posicion = scan.nextInt();
            if (posicion <= 4 && posicion >= 0) {
                if (misMascotas[posicion] == null) {
                    misMascotas[posicion] = new Hipopotamo(4,
7, 4);

misMascotas[posicion].setPosicion(posicion);
                    oro -= 3;
                    for (int i = 0; i <= 4; i++) {
                        if (misMascotas[i] instanceof
Caballo) {
                            ((Caballo)
misMascotas[i]).habilidad(misMascotas, posicion);
                        }
                    }
                } else {
                    System.out.println("Ya hay un animal en
ese lugar ");
                }
            } else {
                System.out.println("La posicion tiene que
estar entre 0 y 4");
            }
        } else {
            System.out.println("No tienes el suficiente
dinero ");
        }
    }

    } else if (noAnimal == 29) {
        if (oro >= 3) {
            Scanner scan = new Scanner(System.in);
            int posicion;
            System.out.println("En que posicion lo quiere
poner");
            posicion = scan.nextInt();
            if (posicion <= 4 && posicion >= 0) {
                if (misMascotas[posicion] == null) {
                    misMascotas[posicion] = new Delfin(4, 6,
4);

misMascotas[posicion].setPosicion(posicion);

```

```

        oro -= 3;
        for (int i = 0; i <= 4; i++) {
            if (misMascotas[i] instanceof
Caballo) {
                ((Caballo)
misMascotas[i]).habilidad(misMascotas, posicion);
            }
        }
    } else {
        System.out.println("Ya hay un animal en
ese lugar ");
    }
    } else {
        System.out.println("La posicion tiene que
estar entre 0 y 4");
    }
    } else {
        System.out.println("No tienes el suficiente
dinero ");
    }
}

    } else if (noAnimal == 30) {
        if (oro >= 3) {
            Scanner scan = new Scanner(System.in);
            int posicion;
            System.out.println("En que posicion lo quiere
poner");
            posicion = scan.nextInt();
            if (posicion <= 4 && posicion >= 0) {
                if (misMascotas[posicion] == null) {
                    misMascotas[posicion] = new Puma(3, 7,
4);
                }

                misMascotas[posicion].setPosicion(posicion);
                oro -= 3;
                for (int i = 0; i <= 4; i++) {
                    if (misMascotas[i] instanceof
Caballo) {
                        ((Caballo)
misMascotas[i]).habilidad(misMascotas, posicion);
                    }
                }
            } else {
                System.out.println("Ya hay un animal en
ese lugar ");
            }
        } else {

```

```

        System.out.println("La posicion tiene que
estar entre 0 y 4");
    }
    } else {
        System.out.println("No tienes el suficiente
dinero ");
    }

    } else if (noAnimal == 31) {
        if (oro >= 3) {
            Scanner scan = new Scanner(System.in);
            int posicion;
            System.out.println("En que posicion lo quiere
poner");
            posicion = scan.nextInt();
            if (posicion <= 4 && posicion >= 0) {
                if (misMascotas[posicion] == null) {
                    misMascotas[posicion] = new Llama(3, 6,
4);

misMascotas[posicion].setPosicion(posicion);
                    oro -= 3;
                    for (int i = 0; i <= 4; i++) {
                        if (misMascotas[i] instanceof
Caballo) {
                            ((Caballo)
misMascotas[i]).habilidad(misMascotas, posicion);
                        }
                    }
                } else {
                    System.out.println("Ya hay un animal en
ese lugar ");
                }
            } else {
                System.out.println("La posicion tiene que
estar entre 0 y 4");
            }
        } else {
            System.out.println("No tienes el suficiente
dinero ");
        }
    }

}

}

//este sera el encargado de comprar comidas
public void compraComida(Mascota[] misMascotas, int rondas) {
    if (rondas == 1) {

```



```

        for (int i = 0; i < 2; i++) {
            mostrandComidaTier1(alimentosAleatoriosTier1());
        }

        boolean seguirComprando = true;
        while (seguirComprando) {
            Scanner scan = new Scanner(System.in);
            System.out.println("Digite el numero de la comida
que quiere comprar");
            System.out.println("Si quiere salir presione 0");
            int opcionComida = scan.nextInt();
            if (opcionComida == 0) {
                break;
            }
            comidaParaComprarTier1(opcionComida,
misMascotas);

        }

    } else if (rondas == 2 || rondas == 3) {
        for (int i = 0; i < 2; i++) {
            int tierAleatorio = (int) (Math.random() * (2 -
1) + 1);
            if (tierAleatorio == 1) {

mostrandComidaTier1(alimentosAleatoriosTier1());
            } else if (tierAleatorio == 2) {

mostrandComidaTier2(alimentosAleatoriosTier2());
            }
        }
        boolean seguirComprando = true;
        while (seguirComprando) {
            Scanner scan = new Scanner(System.in);
            System.out.println("Digite el numero de la comida
que quiere comprar");
            System.out.println("Si quiere salir presione 0");
            int opcionComida = scan.nextInt();
            if (opcionComida == 0) {
                break;
            }
            if (misMascotas[opcionComida] != null) {

            } else {
                noHayAnimal(opcionComida, misMascotas);
            }
            if (opcionComida == 1 || opcionComida == 2) {

```

```

        comidaParaComprarTier1(opcionComida,
misMascotas);
    } else if (opcionComida == 3 || opcionComida ==
4) {
        comidaParaComprarTier2(opcionComida,
misMascotas);
    }

}

    } else if (rondas >= 4 && rondas < 6) {
        for (int i = 0; i < 2; i++) {
            int tierAleatorio = (int) (Math.random() * (3 -
1) + 1);
            if (tierAleatorio == 1) {

mostrandComidaTier1(alimentosAleatoriosTier1());
            } else if (tierAleatorio == 2) {

mostrandComidaTier2(alimentosAleatoriosTier2());
            } else if (tierAleatorio == 3) {

mostrandComidaTier3(alimentosAleatoriosTier3());
            }
        }
        boolean seguirComprando = true;
        while (seguirComprando) {
            Scanner scan = new Scanner(System.in);
            System.out.println("Digite el numero de la comida
que quiere comprar");
            System.out.println("Si quiere salir presione 0");
            int opcionComida = scan.nextInt();
            if (opcionComida == 0) {
                break;
            }
            if (misMascotas[opcionComida] != null) {

            } else {
                noHayAnimal(opcionComida, misMascotas);
            }
            if (opcionComida == 1 || opcionComida == 2) {
                comidaParaComprarTier1(opcionComida,
misMascotas);
            } else if (opcionComida == 3 || opcionComida ==
4) {
                comidaParaComprarTier2(opcionComida,
misMascotas);
            }
        }
    }
}

```

```

        } else if (opcionComida >= 5 || opcionComida <=
7) {
            comidaParaComprarTier3(opcionComida,
misMascotas);
        }

    }

    } else if (rondas >= 6) {
        for (int i = 0; i < 2; i++) {
            int tierAleatorio = (int) (Math.random() * (4 -
1) + 1);
            if (tierAleatorio == 1) {

mostrandComidaTier1(alimentosAleatoriosTier1());
            } else if (tierAleatorio == 2) {

mostrandComidaTier2(alimentosAleatoriosTier2());
            } else if (tierAleatorio == 3) {

mostrandComidaTier3(alimentosAleatoriosTier3());
            } else if (tierAleatorio == 4) {

mostrandComidaTier4(alimentosAleatoriosTier4());
            }
        }
        boolean seguirComprando = true;
        while (seguirComprando) {
            Scanner scan = new Scanner(System.in);
            System.out.println("Digite el numero de la comida
que quiere comprar");
            System.out.println("Si quiere salir presione 0");
            int opcionComida = scan.nextInt();
            if (opcionComida == 0) {
                break;
            }
            if (misMascotas[opcionComida] != null) {

            } else {
                noHayAnimal(opcionComida, misMascotas);
            }
            if (opcionComida == 1 || opcionComida == 2) {
                comidaParaComprarTier1(opcionComida,
misMascotas);
            } else if (opcionComida == 3 || opcionComida ==
4) {
                comidaParaComprarTier2(opcionComida,
misMascotas);
            }
        }
    }
}

```

```

        } else if (opcionComida >= 5 || opcionComida <=
7) {
            comidaParaComprarTier3(opcionComida,
misMascotas);
        } else if (opcionComida >= 8 && opcionComida <=
10) {
            comidaParaComprarTier4(opcionComida,
misMascotas);
        }
    }
}
}
}
}

```

```

//comprobando que hay un animal en la posicion que digito
public void noHayAnimal(int opcionAnimal, Mascota[]
misMascotas) {
    if (misMascotas[opcionAnimal] == null) {
        System.out.println("No hay un animal en esta
posicion");
    }
}
}

```

```

//Encargados de generar, mostrar y pode comprar comida de
tier 1
public int alimentosAleatoriosTier1() {
    int comidaGenerada;
    comidaGenerada = (int) (Math.random() * (2 - 1) + 1);
    return comidaGenerada;
}

```

```

public void mostrandComidaTier1(int noComida) {
    if (noComida == 1) {
        System.out.println("1. Manzana");
    } else if (noComida == 2) {
        System.out.println("2. Naranja");
    }
}
}

```

```

public void comidaParaComprarTier1(int noComida, Mascota[]
misMascotas) {
    if (noComida == 1) {
        if (oro >= 3) {
            Scanner scan = new Scanner(System.in);
            int posicion;
            System.out.println("Digite la posicion del Animal
al que se lo quiere dar");
            posicion = scan.nextInt();

```

```

        if (misMascotas[posicion] != null) {
            if (posicion <= 4 && posicion >= 0) {
                misMascotas[posicion].earnLife(1);
                misMascotas[posicion].earnDamage(1);
                oro -= 3;
            } else {
                System.out.println("La posicion tiene que
estar entre 0 y 4");
            }
        } else {
            System.out.println("No hay una animal en esa
posicion");
        }
    } else {
        System.out.println("No tienes el suficiente
dinero ");
    }
}

    } else if (noComida == 2) {
        if (oro >= 3) {
            Scanner scan = new Scanner(System.in);
            int posicion;
            System.out.println("Digite la posicion del Animal
al que se lo quiere dar");
            posicion = scan.nextInt();
            if (misMascotas[posicion] != null) {
                if (posicion <= 4 && posicion >= 0) {

misMascotas[posicion].efectoActivo("Miel");
                oro -= 3;
            } else {
                System.out.println("La posicion tiene que
estar entre 0 y 4");
            }
        } else {
            System.out.println("No hay una animal en esa
posicion");
        }
    } else {
        System.out.println("No tienes el suficiente
dinero ");
    }
}

    }
}
}

```

```

//Encargados de generar, mostrar y poder comprar comida de
tier 2
    public int alimentosAleatoriosTier2() {
        int comidaGenerada;
        comidaGenerada = (int) (Math.random() * (4 - 3) + 3);
        return comidaGenerada;
    }

    public void mostrandComidaTier2(int noComida) {
        if (noComida == 1) {
            System.out.println("3. Pastelito");
        } else if (noComida == 2) {
            System.out.println("4. Hueso de carne");
        }
    }

    public void comidaParaComprarTier2(int noComida, Mascota[]
misMascotas) {
        if (noComida == 3) {
            if (oro >= 3) {
                Scanner scan = new Scanner(System.in);
                int posicion;
                System.out.println("Digite la posicion del Animal
al que se lo quiere dar");
                posicion = scan.nextInt();
                if (misMascotas[posicion] != null) {
                    if (posicion <= 4 && posicion >= 0) {

misMascotas[posicion].efectoActivo("Pastelito");
                    misMascotas[posicion].earnLifeInBatle(3);

misMascotas[posicion].earnDamageInBatle(3);
                    oro -= 3;
                } else {
                    System.out.println("La posicion tiene que
estar entre 0 y 4");
                }
            } else {
                System.out.println("No hay una animal en esa
posicion");
            }
        } else {
            System.out.println("No tienes el suficiente
dinero ");
        }

        } else if (noComida == 4) {
            if (oro >= 3) {

```

```

        Scanner scan = new Scanner(System.in);
        int posicion;
        System.out.println("Digite la posicion del Animal
al que se lo quiere dar");
        posicion = scan.nextInt();
        if (misMascotas[posicion] != null) {
            if (posicion <= 4 && posicion >= 0) {
                misMascotas[posicion].efectoActivo("Hueso
de carne");
                misMascotas[posicion].earnDamage(5);
                oro -= 3;
            } else {
                System.out.println("La posicion tiene que
estar entre 0 y 4");
            }
        } else {
            System.out.println("No hay una animal en esa
posicion");
        }
        System.out.println("No tienes el suficiente
dinero ");
    }
}

//Encargados de generar, mostrar y poder comprar comida de
tier 3
public int alimentosAleatoriosTier3() {
    int comidaGenerada;
    comidaGenerada = (int) (Math.random() * (7 - 5) + 5);
    return comidaGenerada;
}

public void mostrandComidaTier3(int noComida) {
    if (noComida == 1) {
        System.out.println("5. Ajo");
    } else if (noComida == 2) {
        System.out.println("6. Ensalada");
    } else if (noComida == 3) {
        System.out.println("7. Pera");
    }
}

public void comidaParaComprarTier3(int noComida, Mascota[]
misMascotas) {

```

```

        if (noComida == 5) {
            if (oro >= 3) {
                Scanner scan = new Scanner(System.in);
                int posicion;
                System.out.println("Digite la posicion del Animal
al que se lo quiere dar");
                posicion = scan.nextInt();
                if (misMascotas[posicion] != null) {
                    if (posicion <= 4 && posicion >= 0) {

misMascotas[posicion].efectoActivo("Ajo");
                        oro -= 3;
                    } else {
                        System.out.println("La posicion tiene que
estar entre 0 y 4");
                    }
                } else {
                    System.out.println("No hay una animal en esa
posicion");
                }
            } else {
                System.out.println("No tienes el suficiente
dinero ");
            }
        }

        } else if (noComida == 6) {
            if (oro >= 3) {
                darEnsalada(misMascotas);
            } else {
                System.out.println("No tienes el suficiente
dinero ");
            }
        }

        } else if (noComida == 7) {
            if (oro >= 3) {
                Scanner scan = new Scanner(System.in);
                int posicion;
                System.out.println("Digite la posicion del Animal
al que se lo quiere dar");
                posicion = scan.nextInt();
                if (misMascotas[posicion] != null) {
                    if (posicion <= 4 && posicion >= 0) {
                        misMascotas[posicion].earnDamage(2);
                        misMascotas[posicion].earnLife(2);
                        oro -= 3;
                    } else {
                        System.out.println("La posicion tiene que
estar entre 0 y 4");
                    }
                }
            }
        }
    }
}

```



```

    }
    } else {
        System.out.println("No hay una animal en esa
posicion");
    }
    } else {
        System.out.println("No tienes el suficiente
dinero ");
    }
}

```

```

}
}

```

```

public void darEnsalada(Mascota[] misMascotas) {
    int animal1 = (int) (Math.random() * (4 - 0) + 0);
    int animal2 = (int) (Math.random() * (4 - 0) + 0);

```

```

    if (misMascotas[animal1] != null) {
        misMascotas[animal1].earnDamage(1);
        misMascotas[animal1].earnLife(1);
    }
    if (misMascotas[animal2] != null) {
        misMascotas[animal2].earnDamage(1);
        misMascotas[animal2].earnLife(1);
    }
}

```

```

}

```

```

//Encargado de generar, mostrar y poder comprar comprar
comida de tier 4

```

```

public int alimentosAleatoriosTier4() {
    int comidaGenerada;
    comidaGenerada = (int) (Math.random() * (10 - 8) + 8);
    return comidaGenerada;
}

```

```

public void mostrandComidaTier4(int noComida) {
    if (noComida == 1) {
        System.out.println("8. Chile");
    } else if (noComida == 2) {
        System.out.println("9. Chocolate");
    } else if (noComida == 3) {
        System.out.println("10. Sushi");
    }
}
}

```

```

public void comidaParaComprarTier4(int noComida, Mascota[]
misMascotas) {

```

```

        if (noComida == 8) {
            if (oro >= 3) {
                Scanner scan = new Scanner(System.in);
                int posicion;
                System.out.println("Digite la posicion del Animal
al que se lo quiere dar");
                posicion = scan.nextInt();
                if (misMascotas[posicion] != null) {
                    if (posicion <= 4 && posicion >= 0) {

misMascotas[posicion].efectoActivo("Chile");

misMascotas[posicion].earnDamageInBatle(5);
                        oro -= 3;
                    } else {
                        System.out.println("La posicion tiene que
estar entre 0 y 4");
                    }
                } else {
                    System.out.println("No hay una animal en esa
posicion");
                }
            } else {
                System.out.println("No tienes el suficiente
dinero ");
            }
        }

    } else if (noComida == 9) {
        if (oro >= 3) {
            Scanner scan = new Scanner(System.in);
            int posicion;
            System.out.println("Digite la posicion del Animal
al que se lo quiere dar");
            posicion = scan.nextInt();
            if (misMascotas[posicion] != null) {
                if (posicion <= 4 && posicion >= 0) {

misMascotas[posicion].subirXpPorChocolate(1);
                    oro -= 3;
                } else {
                    System.out.println("La posicion tiene que
estar entre 0 y 4");
                }
            } else {
                System.out.println("No hay una animal en esa
posicion");
            }
        } else {
    }
}

```

```
        System.out.println("No tienes el suficiente  
dinero ");  
    }
```

```
    } else if (noComida == 10) {  
        if (oro >= 3) {  
            darSushi(misMascotas);  
            oro -= 3;  
        } else {  
            System.out.println("No tienes el suficiente  
dinero ");  
        }  
    }
```

```
    }  
}
```

```
public void darSushi(Mascota[] misMascotas) {  
    int animal1 = (int) (Math.random() * (4 - 0) + 0);  
    int animal2 = (int) (Math.random() * (4 - 0) + 0);  
    int animal3 = (int) (Math.random() * (4 - 0) + 0);
```

```
    if (misMascotas[animal1] != null) {  
        misMascotas[animal1].earnDamage(1);  
        misMascotas[animal1].earnLife(1);  
    }  
    if (misMascotas[animal2] != null) {  
        misMascotas[animal2].earnDamage(1);  
        misMascotas[animal2].earnLife(1);  
    }  
    if (misMascotas[animal3] != null) {  
        misMascotas[animal3].earnDamage(1);  
        misMascotas[animal3].earnLife(1);  
    }  
}
```

```
}
```

```
    //proceso para ver a las mascotas  
    public void verMascotas(Mascota[] misMascotas) {  
        int espaciosSinMascotas = 0;  
        for (int i = 0; i <= 4; i++) {  
            if (misMascotas[i] != null) {  
                System.out.println(i + " Nombre: " +  
misMascotas[i].getNombre() + " Vida: " +  
misMascotas[i].getLife() + " Daño: " +  
misMascotas[i].getDamage() + " XP: " + misMascotas[i].getXP());  
            } else {  
                espaciosSinMascotas += 1;  
            }  
        }  
    }  
}
```

```

    }
    if (espaciosSinMascotas == 5) {
        System.out.println("No tienes mascotas");
    }
}

}

}

//proceso para ordenar mascotas
public void ordenarMascotas(Mascota[] misMascotas) {
    Scanner scan = new Scanner(System.in);
    System.out.println("1. Ver mascotas");
    System.out.println("2. Mover de lugar cierta Mascota");
    System.out.println("Si al lugar donde quieres mover a tu
mascota hay otra igual, se fusionaran");
    int opcion = scan.nextInt();
    if (opcion == 1) {
        verMascotas(misMascotas);
    } else if (opcion == 2) {
        moverDeLugar(misMascotas);
    }
}

}

//proceso para mover a las mascotas
public void moverDeLugar(Mascota[] misMascotas) {
    Scanner scanner = new Scanner(System.in);
    System.out.println("Digite el animal que quiere
cambiar");
    int animalAMover = scanner.nextInt();
    System.out.println("Digite la posicion a la que lo quiere
mover");
    int posicionAMover = scanner.nextInt();

    if (misMascotas[posicionAMover] != null) {
        if
(misMascotas[animalAMover].getNombre().equals(misMascotas[posici
onAMover].getNombre())) {
            boolean fusion = true;
            misMascotas[posicionAMover].subirXp(fusion);
            misMascotas[animalAMover] = null;
        } else {
            Mascota mascotaAux;
            mascotaAux = misMascotas[posicionAMover];
            misMascotas[posicionAMover] =
misMascotas[animalAMover];
            misMascotas[animalAMover] = mascotaAux;
        }
    }

} else {

```

```

        misMascotas[posicionAMover] =
misMascotas[animalAMover];
        misMascotas[animalAMover] = null;
    }
}

//proceso para vender mascotas
public void venderMascotas(Mascota[] misMascotas) {
    Scanner scanner = new Scanner(System.in);
    int posicion;
    System.out.println("Digite la posicion de la mascota que
quiere vender");
    posicion = scanner.nextInt();
    if (misMascotas[posicion] != null) {
        if (misMascotas[posicion] instanceof Castor) {
            ((Castor)
misMascotas[posicion]).habilidad(misMascotas, true);
        }
        misMascotas[posicion] = null;
        oro += 1;
    } else {
        System.out.println("No hay una mascota para vender en
esa posicion");
    }
}
}

```

Diagrama UML del espacio de batalla



## Codigo

```
public int espacioLibre1(Mascota[] copia) {
    int espacioLibre1 = 0;
    for (int i = 0; i <= 4; i++) {
        if (copia[i] == null) {
            espacioLibre1 += 1;
        }
    }
    return espacioLibre1;
}

//[null][null][mascota][mascota][mascota]
public void ordenarMascotas(Mascota[] copia) {
    if (espacioLibre1(copia) >= 1) {
        int espaciosNulos = 0;
        for (int i = 0; i <= 4; i++) {
            if (copia[i] == null) {
                espaciosNulos += 1;
            } else {
                copia[i - espaciosNulos] = copia[i];
                if (espaciosNulos != 0) {
                    copia[i] = null;
                }
            }
        }
    }
}

}

public void animalConHabilidad(Mascota[] copia, Mascota[]
mascotasEnemigas) {
    for (int i = 0; i <= 4; i++) {
        if (copia[i] instanceof Mosquito) {
            ((Mosquito) copia[i]).habilidad(mascotasEnemigas);
        } else if (copia[i] instanceof Sapo) {
            ((Sapo) copia[i]).habilidad(copia);
        } else if (copia[i] instanceof Dodo) {
            ((Dodo) copia[i]).habilidad(copia);
        } else if (copia[i] instanceof Delfin) {
            ((Delfin) copia[i]).habilidad(mascotasEnemigas);
        }
    }
}

}
```

```

//[Sapo] [M] [M] [M] [M]      [null] [Sapo] [M] [M] [M]
[null] [null] [Sapo] [M] [M]
public void pelea(Mascota[] copia, Mascota[] mascotasEnemigas,
int rondas) {
    ordenarMascotas(copia);
    ordenarMascotas(mascotasEnemigas);
    animalConHabilidad(copia, mascotasEnemigas);
    animalConHabilidad(mascotasEnemigas, copia);
    while (copia[0] != null && mascotasEnemigas[0] != null) {
        if (copia[0] != null) {
            copia[0].atacar(mascotasEnemigas);
            if (copia[0] instanceof Grillo) {
                if (copia[0].getLife() <= 0) {
                    ((Grillo)
mascotasEnemigas[0]).habilidad(copia);
                }
            }
        }
        if (mascotasEnemigas[0] != null) {
            mascotasEnemigas[0].atacar(copia);
            if (copia[0] instanceof Grillo) {
                if (copia[0].getLife() <= 0) {
                    ((Grillo) copia[0]).habilidad(copia);
                }
            }
        }
        if (copia[0].getMuerto()) {
            copia[0] = null;
            for (int i = 1; i <= 4; i++) {
                copia[i - 1] = copia[i];
                if (copia[i] == null) {
                    copia[i - 1] = null;
                }
            }
            if (copia[4] != null) {
                copia[4] = null;
            }
        }
        if (mascotasEnemigas[0].getMuerto()) {
            mascotasEnemigas[0] = null;
            for (int i = 1; i <= 4; i++) {
                mascotasEnemigas[i - 1] = mascotasEnemigas[i];
                if (mascotasEnemigas[i] == null) {
                    mascotasEnemigas[i - 1] = null;
                }
            }
            if (mascotasEnemigas[4] != null) {
                mascotasEnemigas[4] = null;
            }
        }
    }
}

```

```

    }
}

    reporteBatalla(copia, mascotasEnemigas);

}

public int vidasPerdidas(Mascota[] copia, int rondas, boolean
gano, Mascota[] mascotasEnemigas) {
    if (copia[0] == null && mascotasEnemigas[0] != null) {
        if (rondas <= 3) {
            gano = false;
            return 1;
        } else if (rondas > 3 && rondas <= 6) {
            gano = false;
            return 2;

        } else {
            gano = false;
            return 3;

        }
    } else {
        return 0;
    }
}

public int victoriasGanadas(Mascota[] mascotasEnemigas, boolean
gano, Mascota[] copia) {
    if (mascotasEnemigas[0] == null && copia[0] != null) {
        return 1;
    } else {
        return 0;
    }
}

public void reporteBatalla(Mascota[] copia, Mascota[]
mascotasEnemigas) {
    if (copia[0] == null && mascotasEnemigas[0] != null) {
        System.out.println("Perdiste");
        for (int i = 0; i <= 4; i++) {
            if (mascotasEnemigas[i] != null) {
                System.out.println("Mascota viva: " +
mascotasEnemigas[i].getNombre());
                System.out.println("Con " +
mascotasEnemigas[i].getLife() + " de vida");
            }
        }
    }
}

```



```

    }
    } else if (mascotasEnemigas[0] == null && copia[0] != null) {
        System.out.println("Ganaste");
        for (int i = 0; i <= 4; i++) {
            if (copia[i] != null) {
                System.out.println("Mascota viva: " +
copia[i].getNombre());
                System.out.println("Con " + copia[i].getLife() +
" de vida");
            }
        }
    }
    if (copia[0] == null && mascotasEnemigas[0] == null) {
        System.out.println("Empate");
    }
}
}

```