Universidad de San Carlos de Guatemala

Centro Universitario de Occidente

División de Ciencias de la Ingeniería

Laboratorio de Introducción a la Programación y Computación I

Sección “A”

Manual Técnico

Nombre completo: José Javier García Escobar

Registro Académico: 202031408

Fecha: 04 de marzo del 2022

Índice

Introducción

El presente proyecto trata sobre la implementación de un programa en consola consistiendo en un juego recreativo basado Super Auto Pets. Donde el usuario podrá crear su propio equipo de mascotas las cuales pelearan con otros animales.

Herramientas Utilizadas en el desarrollo

- Apache NetBeans IDE 12.0
- Java SE Development Kit 8 Update 281(64-bit), versión 8.0.2810.9

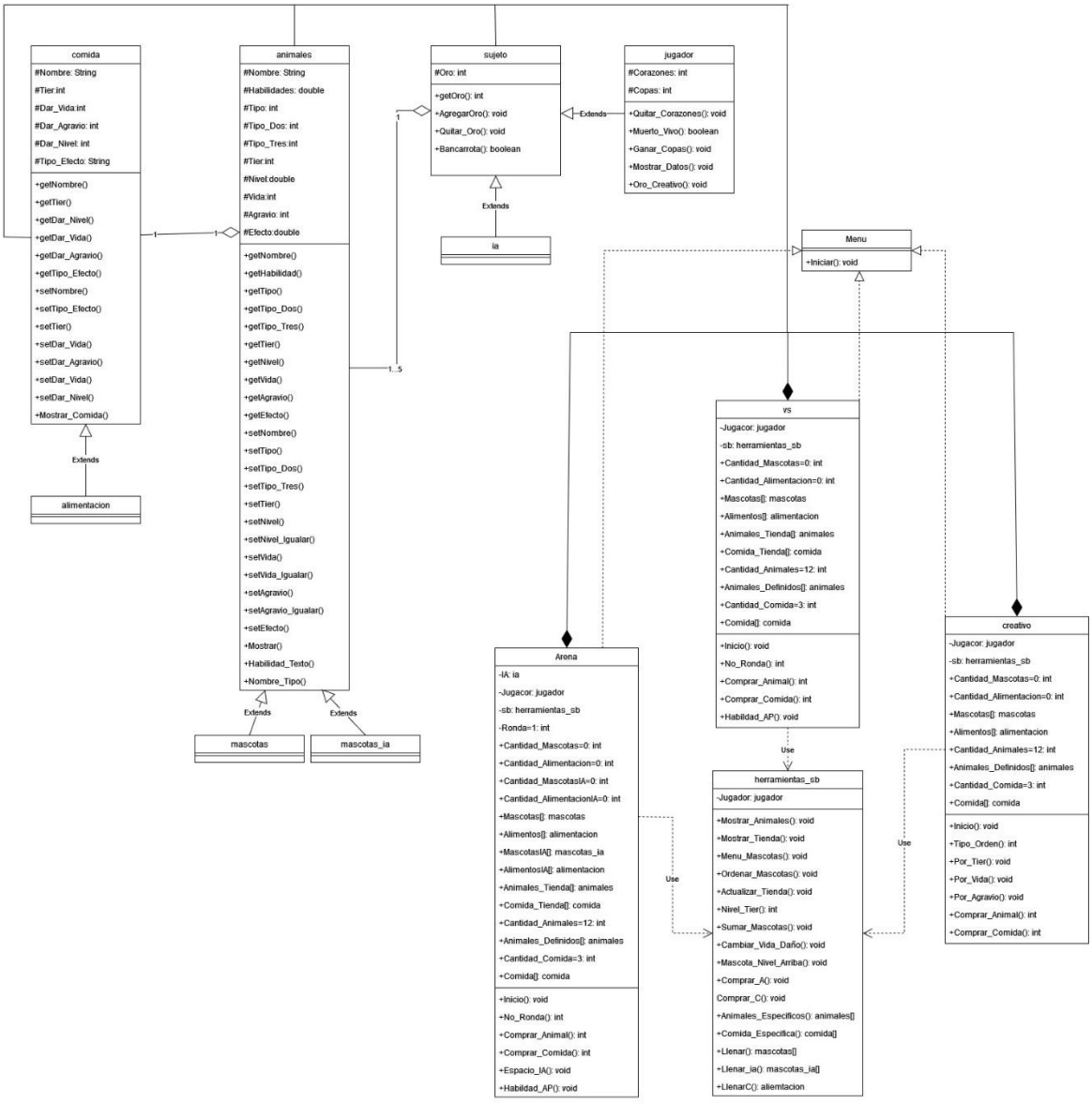
Contenido Técnico

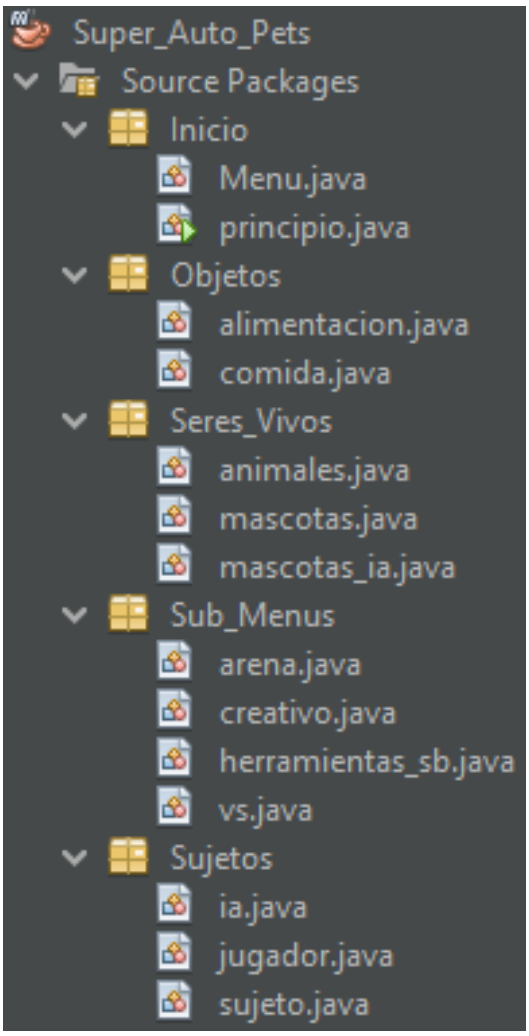
- Sistemas basados en NT (Windows XP, Windows 7, Windows 8, Windows 10, Windows Server) o Terminal de Linux

Análisis y Requerimiento

- Lenguaje de Programación Oracle JAVA (versión del 16 de abril de 2019)
- Compilador por defecto

Contenido Técnico






```
Inicio  Iniciar
Var opicion
Escribir” Bienvenido a SUPER_AUTO_PETS, la copia barata... no espere mucho jeje”
Escribir” Escoja un modo de juego:”
Hacer
        Escribir” 1) Modo Arena -Gana 10 partidas-”
        Escribir” 2) Modo Versus -Pelea con otros jugadores-"
        Escribir” 3) Modo Creativo -Arma tu dream team-"
        Escribir” 0) Salida"
        opcion=entrada
        Si opcion ==1
                arena partida= nueva arena
                partida.inicio()
        Mas si opcion==2

        Mas si opcion==3
                creativo partida=nueva creativo
                partida.inicio()
        Mas si opcion==0
```

```
                Escribir” Te perdiste del mejor juego ;c”
        Fin si
Mientras opcion !=0
Fin Iniciar

Inicio arena
Var IA
Var Jugador
Var sb
Var Ronda=1
Var Cantidad_Mascotas=0, Cantidad_Alimentacion=0
Var Cantidad_MascotasIA=0, Cantidad_AlimentacionIA=0
Var Mascotas[]
Var Alimentacion[]
Var MascotasIA[]
Var AlimentacionIA[]
Var Animales_Tienda[]
Var Comida_Tienda[]
Var Cantidad_Animales=12
Var Animales_Definidos[]
Var Cantidad_Comida=3
Var Comida[]

Constructor arena()
        IA=nuevo ia()
        Jugador=nuevo jugador()
        sb=nuevo herramientas_sb(Jugador)
        Mascotas=nuevo mascotas[5]
        Alimentacion=nuevo alimentacion[5]
        MascotasIA=nuevo mascotas_ia[5]
        AlimentacionIA=nuevo alimentacion[5]
        Animales_Tienda=nuevo animales[5]
        Comida_Definidos=nuevo animales[Cantidad_Animales]
        Comida=nuevo comida[Cantidad_Comida]
Fin Constructor

Vacio publico inicio()
        Var Opcion=0, Cantidad_TiendaA=3, Cantidad_TiendaC=2
        Mascotas=sb.Llenar()
        Alimentacion=sb.LlenarC()
        Animales_Definidos=sb.Animales_Especificados()
        Comida=sb.Comida_Especificados()
        Sb.Actualizar_Tienda(Jugar, Ronda, Opcion, Cantidad_Animales, Cantidad_TiendaA,
Animales_Tienda, Animales_Definidos, Cantidad_TiendaC, Comida, Comida_Tienda)

        Escribir “-------MODO AREMA-------"
        Escribir “--Se le dara una dotacion inicial de 10 de oro para que arme su equipo--"
        Mientras Jugador.Muerto_Vivo()
                Jugador.Mostrar_Datos()
                Escribir “--Cuando este preparado puche la opcion batalla—“
                Escribir “ 1) Animales de tienda"
```

```
                Escribir “ 2) Comida de tienda”
                Escribir “ 3) Actualizar tienda"
                Escribir “ 4) Comprar animal"
                Escribir “ 5) Comprar comida"
                Escribir “ 6) Mirar Mascotas"
                Escribir “ 8) Batalla"
                Opcion=entrada
                Si Opcion==1
                        sb.Mostrar_Animales(Cantidad_TiendaA, Animales_Tienda)
                Mas si Opcion==2
                        sb.Mostrar_Tienda(Comida_Tienda, Cantidad_TiendaC)
                Mas si Opcion==3
                        Cantidad_TiendaC=2;
                          Cantidad_TiendaA=No_Ronda(Ronda);
                          sb.Actualizar_Tienda(Jugador, Ronda, Opcion, Cantidad_Animales
        Cantidad_TiendaA, Animales_Tienda, Animales_Definidos, Cantidad_TiendaC,Comida,
        Comida_Tienda)
                Mas si Opcion==4
                        Escribir “-----------------------------"
                        Escribir “--Que animal desea comprar?--"
                        // se le envia el valor de la posicion en que este el animal en tienda
                          //y la cantidad de animales que hay en tienda
                        Var Animal=entrada
                        Cantidad_TiendaA=Comprar_Animal(Jugador, Animal-1, Cantidad_TiendaA,
        Cantidad_Mascotas, Mascotas, Animales_Tienda)
                Mas si Opcion==5
                        Escribir “-----------------------------"
                        Escribir “—Que comida desea comprar?--"
                Mas si Opcion==6
                        sb.Menu_Mascotas(Jugador, Cantidad_Mascotas, Mascotas, Alimentos)
                Mas si Opcion==8
                        Espacio_IA()
                        Ronda++
                        Jugador.Oro_Ronda()
                Mas si Opcion==0
                        Romper Mientras hace
                Demas
                        Escribir "------------------------"
                          Escribir "--Esa no es una opcion--"
                Fin si
        Fin Mientras
        Escribir “Estas muerto”
Fin inicio

Entero publico No_Ronda(Var Ronda)
        Var Cantidad_TiendaA=0
        Si Ronda>0 ademas Ronda<=3
                Regresar Cantidad_TiendaA=3
        Mas si Ronda>3 ademas Ronda<=6
                Regresar Cantidad_TiendaA=4
        Mas si Ronda>=7
```

```
                Regresar Cantidad_TiendaA=5
        Demas
                Regresar Cantidad_TiendaA
        Fin si
Fin No_Ronda

//Opcion 4
   public int Comprar_Animal(sujeto Quien, int Animal, int Cantidad_TiendaA,
      int Cantidad_M, animales[] Mascotas, animales[] Animales_Tienda){

      int Existencia=5, Vacio=0, Nivel=0;
      // si la seleccion del animal en tienda es coherente y
      //si se tiene el oro suficiente se procede con la primera sentencia de seleccion
      if(Cantidad_M<5&&(Animal>=0&&Animal<=Cantidad_TiendaA)&&(Quien.getOro()>=3)){
         if(Cantidad_M>0){
            for(int i=0; i<Cantidad_M;i++){
               if(Animales_Tienda[Animal].getNombre().equals(Mascotas[i].getNombre())){
                  Existencia=i;
                  break;
               }
            }
            for(int i=0; i<Cantidad_M;i++){
               if(Mascotas[i].getNivel()==3){
                  Nivel=i;
                  break;
               }
            }
         }
         for (int i=0; i<Cantidad_M;i++) {
            if ("".equals(Mascotas[i].getNombre()))Vacio=i;
         }

if(Cantidad_M>0&&Animales_Tienda[Animal].getNombre()==(Mascotas[Existencia].getNombre())&&Masc
otas[Nivel].getNivel()!=3){
            int Sub_Menu=0;
            if(Quien instanceof jugador){
               Scanner entrada=new Scanner(System.in);
               System.out.println("--Ya tiene una mascota de esta clase--");
               System.out.println("--Desea combinarlo?--");
               System.out.println("  1)Si");
               System.out.println("  2)No");
               Sub_Menu=entrada.nextInt();
            }else if(Quien instanceof ia){
               int Numero=(int) (Math.random()*(10));
               if(Numero<=8){Sub_Menu=1;
               } else if(Numero>8&&Numero<=10){Sub_Menu=2;}
            }
            if(Sub_Menu==1){
               sb.Sumar_Mascotas(Existencia, Animal, Mascotas, Animales_Tienda);
            }else{
               sb.Compra_A(Cantidad_M, Animal, Mascotas, Animales_Tienda);
```

```
                if(Quien instanceof jugador)Cantidad_Mascotas+=1;
                if(Quien instanceof ia)Cantidad_MascotasIA+=1;
            }

        }else if(Cantidad_M>0&&(!"".equals(Mascotas[Vacio].getNombre()))){
            sb.Compra_A(Vacio, Animal, Mascotas, Animales_Tienda);

        }else{
            sb.Compra_A(Cantidad_M, Animal, Mascotas, Animales_Tienda);
            if(Quien instanceof jugador)Cantidad_Mascotas+=1;
            if(Quien instanceof ia)Cantidad_MascotasIA+=1;
        }
        // se sobre escribe la tienda para eliminar el animal comprado omitiento
        //la posicion 5 ya que es de tipo null, debo de cambiar eso
        for(int i=(Animal);i<Animales_Tienda.length-1;i++){
            Animales_Tienda[i]=Animales_Tienda[i+1];
        }
        Quien.Quitar_Oro(3);
        return Cantidad_TiendaA-1;
    }else if(Cantidad_M==5){
        for (int i=0; i<Cantidad_M;i++) {
            if (!"".equals(Mascotas[i].getNombre())) {
                sb.Compra_A(i, Animal, Mascotas, Animales_Tienda);
                Cantidad_TiendaA=Cantidad_TiendaA-1;
                break;
            }else if(i==4&&Quien instanceof jugador){
                System.out.println("----------------------------");
                System.out.println("Perdon pero no tiene espacio");
            }
        }
        return Cantidad_TiendaA;
    }else if(Quien.getOro()<3){
        System.out.println("-----------------------");
        System.out.println("No tiene suficiente oro");
        return Cantidad_TiendaA;
    }else {
        System.out.println("--------------------");
        System.out.println("Esa no es una opcion");
        return Cantidad_TiendaA;
    }

}

//Opcion 5
public int Comprar_Comida(sujeto Quien, int Comida, int Cantidad_TiendaC,int Cantidad_A, animales
M[], comida C_T[], comida A[]){
    Scanner entrada=new Scanner(System.in);
    if(Comida>=0&&Comida<2&&Quien.getOro()>3){
        if(C_T[Comida].getTipo_Efecto()==1){
            // espacio donde se se le suma vida y agravio a los animales de tienda
            for(int i=0;i<Animales_Definidos.length;i++){
```

```
            Animales_Definidos[i].setVida(C_T[Comida].getDar_Vida());
            Animales_Definidos[i].setVida(C_T[Comida].getDar_Agravio());
        }
    }else if(C_T[Comida].getTipo_Efecto()==1.1){
        System.out.println("--Ahora no joven plisss--");
        //Espacio para darle comida a mascotas a azar
        for(int i=(Comida);i<C_T.length-1;i++){
            C_T[i]=C_T[i+1];
        }
        Quien.Quitar_Oro(3);
        return Cantidad_TiendaC-1;
    }else if(C_T[Comida].getTipo_Efecto()>1.1){
        // si la comida es mayor a 1.1 nos indica que la comida sera solo para una mscota
        int Mascota=0;
        if(Quien instanceof jugador){
            System.out.println("--A que mascota le quiere dar la comida?--");
            Mascota=entrada.nextInt()-1;
        }else if(Quien instanceof ia){
            Mascota=(int)(Math.random())*Cantidad_MascotasIA;
        }
        if(M[Mascota].getNombre()!=""){
            if(C_T[Comida].getTipo_Efecto()==1.2){
                //tipo 1.2 solo aumaneta algun atribito de los siguienes, si es mayor a 0
                M[Mascota].setVida(C_T[Comida].getDar_Vida());
                M[Mascota].setAgravio(C_T[Comida].getDar_Agravio());
                M[Mascota].setNivel(C_T[Comida].getDar_Nivel());
                for(int i=(Comida);i<C_T.length-1;i++){
                        C_T[i]=C_T[i+1];
                }
                Quien.Quitar_Oro(3);
                return Cantidad_TiendaC-1;
            }else if(C_T[Comida].getTipo_Efecto()>1.2){
                //los tipo 2.0 en adelante son aliemtos que se ejecutan durante batalla
                if(M[Mascota].getEfecto()!=5){
                    //si el espacio ya tiene una comida se debera preguntar cambio o permanecer
                    int Opcion=0;
                    if(Quien instanceof jugador){
                    System.out.println("Comida almacenada: "+A[M[Mascota].getEfecto()].getNombre());
                    System.out.println("--Desea sustitur el alimento anterior?--");
                    System.out.println("   1)Si");
                    System.out.println("   2)No");
                    Opcion=entrada.nextInt();
                    }else if(Quien instanceof jugador){
                        Opcion=(int)(Math.random())*(1)+1;
                    }
                    if(Opcion==1){
                        sb.Comprar_C(M[Mascota].getEfecto(), Comida, C_T, A);
                        M[Mascota].SetEfecto(Comida);
                        for(int i=(Comida);i<C_T.length-1;i++){
                            C_T[i]=C_T[i+1];
                        }
```

```
                    Quien.Quitar_Oro(3);
                    return Cantidad_TiendaC-1;
                }
            }else if(M[Mascota].getEfecto()==5){
                //si el espacio esta vacio, no hay problema
                if(Cantidad_A==5){
                    for(int i=0; i<A.length;i++){
                        if(A[i].getNombre()==""){
                            sb.Comprar_C(i, Comida, C_T, A);
                            M[Mascota].SetEfecto(i);
                            break;
                        }
                    }
                }else{
                    sb.Comprar_C(Cantidad_A, Comida, C_T, A);
                    M[Mascota].SetEfecto(Cantidad_A);
                    if(Quien instanceof jugador)Cantidad_Alimentacion+=1;
                    if(Quien instanceof ia)Cantidad_AlimentacionIA+=1;
                }
                for(int i=(Comida);i<C_T.length-1;i++){
                    C_T[i]=C_T[i+1];
                }
                Quien.Quitar_Oro(3);
                return Cantidad_TiendaC-1;
            }else{
                if(Quien instanceof jugador){
                    System.out.println("--Perdon pero tendra que escoger --");
                    System.out.println("--otra vez la comida y el aminal --");
                }
            }
        }else{
            if(Quien instanceof jugador){
                System.out.println("--No tiene mascota en esa posicion--");
            }else if(Quien instanceof ia){}
        }
    }
    return Cantidad_TiendaC;
}else if(Quien.getOro()<3){
    System.out.println("---------------------------");
    System.out.println("--No tiene suficiente oro--");
    return Cantidad_TiendaC;
}else{
    System.out.println("-------------------------");
    System.out.println("--Esa no es una opcion--");
    return Cantidad_TiendaC;
}
}

public void Espacio_IA(){
    int Opcion=0, Cantidad_TiendaA=3, Cantidad_TiendaC=2;
```

```
        AlimentosIA=sb.LlenarC();
        MascotasIA=sb.Llenar_ia();

        //Animales_Definidos= sb.Animales_Especificados();
        Comida=sb.Comida_Especificados();
        //agreaga objtos a la tienda
        sb.Actualizar_Tienda(IA, Ronda, Opcion, Cantidad_Animales, Cantidad_TiendaA, Animales_Tienda,
                        Animales_Definidos,Cantidad_TiendaC, Comida, Comida_Tienda);
        do{
           int Numero=(int) (Math.random()*(80));
              if(Numero<=5){Opcion=1;
              }else if(Numero>5&&Numero<=45){Opcion=2;
              }else if(Numero>45&&Numero<=65){Opcion=3;
              }else if(Numero>65&&Numero<=80){Opcion=4;}
           if(Opcion==1&&IA.getOro()>=1){
                 Cantidad_TiendaC=2;
                 Cantidad_TiendaA=No_Ronda(Ronda);
                 sb.Actualizar_Tienda(IA, Ronda, Opcion, Cantidad_Animales, Cantidad_TiendaA,
Animales_Tienda,
                                 Animales_Definidos, Cantidad_TiendaC,Comida, Comida_Tienda);
              }else if(Opcion==2&&IA.getOro()>=3){
                 //se debera de realizar una seleccion al azar de los animales de tienda
                 int Numero_Dos=(int) (Math.random()*Cantidad_TiendaA);
                 int Animal;
                  Animal=Numero_Dos;
                 Cantidad_TiendaA=Comprar_Animal(IA, Animal, Cantidad_TiendaA, Cantidad_MascotasIA,
MascotasIA, Animales_Tienda);
              }else if(Opcion==3&&IA.getOro()>=3){

                 int C=(int) (Math.random()*Cantidad_TiendaC);
                 //Comprar_Comida(sujeto Quien, int Comida, int Cantidad_TiendaC,int Cantidad_A,
animales M[], comida C_T[], comida A[]
                 Cantidad_TiendaC=Comprar_Comida(IA, C, Cantidad_TiendaC, Cantidad_AlimentacionIA,
MascotasIA,Comida_Tienda, AlimentosIA);
              }else if(Opcion==4){
                 sb.Menu_Mascotas(IA, Cantidad_Mascotas, Mascotas, Alimentos);
              }
        }while(IA.getOro()!=0);
     }

public herramientas_sb(jugador Jugador) {
        this.Jugador=Jugador;
     }

     //despliega una lista de animales que esta en tienda o en nuestro poder
     //Opcion 1 u Opcion 6
     public void Mostrar_Animales(int Cantidad_Animales, animales[] Animales){
        for(int i=0;i<Cantidad_Animales;i++){
           System.out.println((i+1)+")-------------------------------------");
           Animales[i].Mostrar();
```

```
        }
    }
    //Opcion 2
    public void Mostrar_Tienda(comida[] Comida_Tienda, int Cantidad_Tienda){
        for(int i=0;i<Cantidad_Tienda;i++){
            System.out.println((i+1)+")-------------------------------------");
            Comida_Tienda[i].Mostrar_Comida();
        }
    }

    //Opcion 6
    public void Menu_Mascotas(sujeto Quien, int Cantidad_Mascotas, mascotas Mascotas[], alimentacion
Alimentos[]){
        Scanner entrada= new Scanner(System.in);
        int Opcion=0;
        do{
            if(Cantidad_Mascotas>0){
                if(Quien instanceof jugador){
                Mostrar_Animales(Cantidad_Mascotas, Mascotas);
                System.out.println("  1) Ordenar mascotas");
                System.out.println("  2) Sumar mascotas");
                System.out.println("  3) Vender mascota");
                System.out.println("  4) Comida de la mascota");
                System.out.println("  0) Regresar");
                Opcion=entrada.nextInt();
                }else if(Quien instanceof ia){
                    int Numero= (int) (Math.random())*(25);
                    if(Numero>9&&Numero<=13){Opcion=1;}
                    else if(Numero>13&&Numero<=21){Opcion=2;}
                    else if(Numero>21&&Numero<=25){Opcion=3;}
                }
                if(Opcion==1){
                    int Primero=0, Segundo=0;
                    if(Quien instanceof jugador){
                        System.out.println("--Que mascotas quiere cambiar de posicion?--");
                        //debera de escoger primero el animalito que quiere mover de espacio
                        Primero=entrada.nextInt();
                        System.out.println("--Segundo animalito:");
                        //luego seleccionar en que posicion quiere dejarlo
                        Segundo=entrada.nextInt();
                    }else if(Quien instanceof ia){
                        Primero=(int)(Math.random())*(Cantidad_Mascotas);
                        Segundo=(int)(Math.random())*(Cantidad_Mascotas);
                    }

if(Primero>=0&&Primero<=Cantidad_Mascotas&&Segundo>=0&&Segundo<=Cantidad_Mascotas){
                        Ordenar_Mascotas(Primero-1, Segundo-1, Mascotas);
                    }else{
                        if(Quien instanceof jugador){
                            System.out.println("Perdon pero no existe mascota en una de las 2 posicines");
                        }
```

```
            }
        }else if(Opcion==2){
            int Primero=0, Segundo=0;
            if(Quien instanceof jugador){
            System.out.println("--Que mascotas quiere sumar?--");
            System.out.println("--Recuerde que debe de ser el mismo animalito para ganar experiencia--
");
            //debera de escoger el animalito que quiera sumar
            Primero=entrada.nextInt();
            System.out.println("--Segundo animalito:");
            //luego seleccionar el animalito que es igual
            Segundo=entrada.nextInt();
            }else if(Quien instanceof ia){
                Primero=(int)(Math.random())*(Cantidad_Mascotas);
                Segundo=(int)(Math.random())*(Cantidad_Mascotas);
            }

            if(Mascotas[Primero-1].getEfecto()!=5&&Mascotas[Segundo-1].getEfecto()!=5){
                int Comidax2=0;
                if(Quien instanceof jugador){
                System.out.println("En las dos mascotas que selecciono tiene un alimento");
                System.out.println("Cual de los 2 alimentos quiere mantener?");
                System.out.println("1) " +Mascotas[Primero-1].getEfecto()+", de la posicion: "+Primero);
                System.out.println("2) " +Mascotas[Segundo-1].getEfecto()+", de la posicion: "+Segundo);
                Comidax2=entrada.nextInt();
                }else if(Quien instanceof ia){
                    Comidax2=(int)(Math.random())*(1)+1;
                }
                if(Comidax2==2){
                    Mascotas[Primero-1].SetEfecto(Mascotas[Segundo-1].getEfecto());

                    Alimentos[Segundo-1]=new alimentacion("",0,0,0,0,0);
                }
            }
            Sumar_Mascotas(Primero-1, Segundo-1, Mascotas, Mascotas);

        }else if(Opcion==3){
            int M=0;
            if(Quien instanceof jugador){
                System.out.println("--Que mascota quiere vender?--");
                M=entrada.nextInt()-1;
            }else if(Quien instanceof ia){
                M=(int)(Math.random())*(Cantidad_Mascotas);
            }
            if(M>=0&&M<Cantidad_Mascotas){
                Jugador.Agregar_Oro((int)Mascotas[M].getNivel());
                Alimentos[M]=new alimentacion("",0,0,0,0,0);
                Mascotas[M]=new mascotas("",0,0,0,0,0,0,0,0,5);
            }else{
                if(Quien instanceof jugador){
                    System.out.println("-----------------------------------");
```

```
                    System.out.println("--Solo tiene "+Cantidad_Mascotas+" mascotas--");
                }
            }
        }else if(Opcion==4){
            System.out.println("--De que animal quiere ver la comida?--");
            int M=entrada.nextInt()-1;

            if(M<Cantidad_Mascotas){
                if(Mascotas[M].getNombre()==""){
                    System.out.println("--------------------------------");
                    System.out.println("No hay un animal en esa posicion");
                }else if(Alimentos[Mascotas[M].getEfecto()].getNombre()!=""){
                    Alimentos[Mascotas[M].getEfecto()].Mostrar_Comida();
                }else if (Alimentos[Mascotas[M].getEfecto()].getNombre()==""){
                    System.out.println("----------------------------------------");
                    System.out.println("La mascota "+Mascotas[M].getNombre()+" no tiene comida");
                }
            }else{
                System.out.println("---------------------------------------");
                System.out.println("Solo tiene "+Cantidad_Mascotas+" mascotas");
            }
        }
    }while(Opcion!=0);
}

//Opcion 6.1
public void Ordenar_Mascotas(int Primero, int Segundo, animales Mascotas[]){
    //Espacio donde permite al jugador ordenar a sus mascotas
    animales Mascota_Guardada[]=new animales[2];
    Mascota_Guardada[0]=Mascotas[Primero];
    Mascota_Guardada[1]=Mascotas[Segundo];
    Mascotas[Segundo]=Mascota_Guardada[0];
    Mascotas[Primero]=Mascota_Guardada[1];
}


//A_T(AnimalesTienda)--A_D(Animales_Definidos)
public void Actualizar_Tienda(sujeto Quien, int Ronda, int Opcion,int Cantidad_Animales, int
Cantidad_TiendaA,
                    animales A_T[], animales A_D[],int Cantidad_TiendaC, comida C[], comida C_T[]){

    if(Jugador.Bancarrota()){
        System.out.println("-----------------------------------------------");
        System.out.println("No tiene oro suficiente para realiar esta accion");
    }else {
        int Contador=0, Tier;
        Tier=Nivel_Tier(Ronda);
        do{
            //genera un numero al azar de la cantidad de animales que hay definidos
            int Animal_Azar=(int) (Math.random()*(Cantidad_Animales));
```

```
            //si el animal se encuentra en el tier por ronda que se establece (si es del tier maximo o es
menor que el maximo)
            //podra aparecer en la tienda
            //de lo contrario el ciclo seguira hasta completar el espacio de tienda pemitido por ronda
            if(A_D[Animal_Azar].getTier()<=Tier){
            A_T[Contador]= A_D[Animal_Azar];
            Contador++;
            }
         // si el espacio en tienda se termina ya no agregara objetos a la tienda al momento de actualizar
         }while(Contador<Cantidad_TiendaA);
         Contador=0;
         do{
            //genera un numero al azar de la cantidad de comida definida
            int Comida_Azar=(int) (Math.random()*(C.length));



            //si la se encuentra en el tier por ronda que se establece (si es del tier maximo o es menor que
el maximo)
            //podra aparecer en la tienda
            //de lo contrario el ciclo seguira hasta completar el espacio de tienda pemitido por ronda
            if(C[Comida_Azar].getTier()<=Tier){
            C_T[Contador]= C[Comida_Azar];
            Contador++;
            }
         // si el espacio en tienda se termina ya no agregara objetos a la tienda al momento de actualizar
         }while(Contador<Cantidad_TiendaC);

         //cada actualizacion es menos uno de oro, si el juego esta iniciando no se
         //restara oro, solo se actualiza para tener algo en tienda
         if(Opcion!=0){
            if(Quien instanceof jugador){
            System.out.println("----------------------------------");
            System.out.println("Tienda actualizada, menos 1 de oro");
            }
            Quien.Quitar_Oro(1);
         }
      }
   }

   public int Nivel_Tier(int Ronda){
      int Tier=1;
      if(Ronda<2){
         return Tier;
      }else if(Ronda>=2&&Ronda<4){
         Tier=2;
         return Tier;
      }else if(Ronda>=4&&Ronda<6){
```

```
            Tier=3;
            return Tier;
        }else if(Ronda>=6&&Ronda<8){
            Tier=4;
            return Tier;
        }else if(Ronda>=8&&Ronda<10){
            Tier=5;
            return Tier;
        }else if(Ronda>=10&&Ronda<12){
            Tier=6;
            return Tier;
        }else if(Ronda>=12){
            Tier=7;
            return Tier;
        }else{
            return Tier;
        }
    }

    public void Sumar_Mascotas(int Primero, int Segundo, animales Animal_Uno[], animales
Animal_Dos[]){
        //Espacio donde permite al jugador subir de nivel a sus mascotas


if(Animal_Uno[Primero]!=Animal_Dos[Segundo]&&Animal_Uno[Primero].getNombre().equals(Animal_Do
s[Segundo].getNombre())){


            //sentencia de seleccion, depende del nivel que tenga la mascota principal
            if(Animal_Uno[Primero].getNivel()>=1&&Animal_Uno[Primero].getNivel()<2){
                Cambiar_Vida_Daño(Primero, Segundo, Animal_Uno, Animal_Dos);
                //se le sumara el punteo de ex designado por nivel
                Animal_Uno[Primero].setNivel(0.5);

                //si al momento de ganar ex sube de nivel se hara la sumatoria de sus caracteristicas (vida,
agravio, habilidad)
                if(Animal_Uno[Primero].getNivel()==2) Mascota_Nivel_Arriba(Primero, Animal_Uno);

                //el segundo animal sera borrado del proceso,
                //se toma en cuenta que puede quedar un espacio null si se toma otra opcion para eliminarlo
                Animal_Dos[Segundo]=new mascotas("",0,0,0,0,0,0,0,0,0);

            }else if(Animal_Uno[Primero].getNivel()>=2&&Animal_Uno[Primero].getNivel()<3){
                Cambiar_Vida_Daño(Primero, Segundo, Animal_Uno, Animal_Dos);
                //se le sumara el punteo de ex designado por nivel
                Animal_Uno[Primero].setNivel(0.33);

                //para tener un nuemro redondo para indicar el nivel exacto
                if(Animal_Uno[Primero].getNivel()==2.99) Animal_Uno[Primero].setNivel(0.01);
```

```
            //si al momento de ganar ex sube de nivel se hara la sumatoria de sus caracteristicas (vida,
agravio, habilidad)
            if(Animal_Uno[Primero].getNivel()==3) Mascota_Nivel_Arriba(Primero, Animal_Uno);

            //el segundo animal sera borrado del proceso,
            //se toma en cuenta que puede quedar un espacio null si se toma otra opcion para eliminarlo
            Animal_Dos[Segundo]=new mascotas("",0,0,0,0,0,0,0,0,0);

         }else if(Animal_Uno[Primero].getNivel()==3){
            System.out.println("----------------------------------------");
            System.out.println("--El nivel 3 es el nivel maximo, desde--");
            System.out.println("--este nivel solo podra subir         --");
            System.out.println("--vida y agravio                      --");
         }
      }else if(Animal_Uno[Primero]==Animal_Dos[Segundo]){
         System.out.println("--------------------------");
         System.out.println("--La posicion es la misma--");
      }else {
         System.out.println("------------------------------");
         System.out.println("--Los animales no son iguales--");
      }
   }

   public void Cambiar_Vida_Daño(int Primero, int Segundo, animales Animal_Uno[], animales
Animal_Dos[]){
      //si el primero es el mas alto sera la base para todo el proceso

if(Animal_Uno[Primero].getVida()>=Animal_Dos[Segundo].getVida()||Animal_Uno[Primero].getAgravio()>
=Animal_Dos[Segundo].getAgravio()){
         Animal_Uno[Primero].setVida(1);
         Animal_Uno[Primero].setAgravio(1);
         //si el segunso es el mas alto se sobre escribe el primero con los datos del segundo
      }else{
         Animal_Uno[Primero].setNivel_Igualar(Animal_Dos[Segundo].getNivel());
         Animal_Uno[Primero].setVida_Igualar(Animal_Dos[Segundo].getVida()+1);
         Animal_Uno[Primero].setAgravio_Igualar(Animal_Dos[Segundo].getAgravio()+1);
      }
   }

   public void Mascota_Nivel_Arriba(int Primero, animales Animal_Uno[]){
      Animal_Uno[Primero].setVida(1);
      Animal_Uno[Primero].setAgravio(1);
   }

   public void Compra_A(int Cantidad_Mascotas, int Animal, animales Mascotas[], animales
Animales_Tienda[]){
      Mascotas[Cantidad_Mascotas].setNombre(Animales_Tienda[Animal].getNombre());
      Mascotas[Cantidad_Mascotas].setTipo(Animales_Tienda[Animal].getTipo());
      Mascotas[Cantidad_Mascotas].setTipo_Dos(Animales_Tienda[Animal].getTipo_Dos());
      Mascotas[Cantidad_Mascotas].setTier(Animales_Tienda[Animal].getTier());
      Mascotas[Cantidad_Mascotas].setNivel(Animales_Tienda[Animal].getNivel());
```

```
        Mascotas[Cantidad_Mascotas].setVida(Animales_Tienda[Animal].getVida());
        Mascotas[Cantidad_Mascotas].setAgravio(Animales_Tienda[Animal].getAgravio());
    }

    public void Comprar_C(int Cantidad_Alimentacion, int Comida, comida C_T[], comida A[]){
        A[Cantidad_Alimentacion].setNombre(C_T[Comida].getNombre());
        A[Cantidad_Alimentacion].setTier(C_T[Comida].getTier());
        A[Cantidad_Alimentacion].setDar_Vida(C_T[Comida].getDar_Vida());
        A[Cantidad_Alimentacion].setDar_Agravio(C_T[Comida].getDar_Agravio());
        A[Cantidad_Alimentacion].setDar_Nivel(C_T[Comida].getDar_Nivel());
        A[Cantidad_Alimentacion].setTipo_Efecto(C_T[Comida].getTipo_Efecto());
    }

    public animales[] Animales_Especificados(){
        //(String Nombre, double Habilidad, int Tipo, int Tipo_Dos, int Tipo_Tres, int Tier, double Nivel, int
Vida, int Agravio, int Efecto) {
        animales[] Animales;
        Animales= new mascotas[]{new mascotas("Hormiga",0.1,1,4,0,1,1,1,2,5),
            new mascotas("Pescado",0.2,3, 0, 0, 1, 1, 3, 2, 5), new mascotas("Mosquito",1.1,2, 0, 0, 1, 1, 2, 2,
5),
            new mascotas("Grillo",1.2,1, 0, 0, 1, 1, 1, 2, 5), new mascotas("Castor",0.1,4, 3, 0, 1, 1, 1, 2, 5),
            new mascotas("Caballo",1.1,6, 7, 0, 1, 1, 1, 2, 5), new mascotas("Nutria",0.1,6, 0, 0, 1, 1, 1, 2, 5),
            new mascotas("Escarabajo",0.3,1, 0, 0, 1, 1, 2, 3, 5), new mascotas("Sapo",0.4,4, 3, 0, 2, 1, 3, 3,
5),
            new mascotas("Dodo",1.1,2, 0, 0, 2, 1, 2, 3, 5),new mascotas("Elefante",1.3,6, 4, 0, 2, 1, 3, 5,5),
            new mascotas("Puerco",1.2,8, 4, 0, 2, 1, 3, 2, 5)};
        return Animales;
    }

    public final comida[] Comida_Especificados(){
        //(String Nombre, int Tier, int Dar_Vida, int Dar_Agravio, int Dar_Nivel)
        comida[] C;
        C= new comida[]{new comida("Manzana", 1, 1, 1, 0,1.2), new comida("Naranja", 1, 0, (int) 0.1,
0,2.2),
            new comida("Miel", 1, 0, 0, 0,2.2)};
        return C;
    }

    public mascotas[] Llenar(){
        mascotas[] M;
        M= new mascotas[]{new mascotas("",0,0,0,0,0,0,0,0,5), new mascotas("",0,0,0,0,0,0,0,0,5),
          new mascotas("",0,0,0,0,0,0,0,0,5), new mascotas("",0,0,0,0,0,0,0,0,5),
          new mascotas("",0,0,0,0,0,0,0,0,5), new mascotas("",0,0,0,0,0,0,0,0,5)};

        return M;

    }

    public mascotas_ia[] Llenar_ia(){
        mascotas_ia[] M;
        M= new mascotas_ia[]{new mascotas_ia("",0,0,0,0,0,0,0,0,5), new mascotas_ia("",0,0,0,0,0,0,0,0,5),
```

```
            new mascotas_ia("",0,0,0,0,0,0,0,0,5), new mascotas_ia("",0,0,0,0,0,0,0,0,5),
            new mascotas_ia("",0,0,0,0,0,0,0,0,5), new mascotas_ia("",0,0,0,0,0,0,0,0,5)};

        return M;

    }

    public alimentacion[] LlenarC(){
        alimentacion[] A;
        A=new alimentacion[]{new alimentacion("",0,0,0,0,0), new alimentacion("",0,0,0,0,0), new
alimentacion("",0,0,0,0,0),
        new alimentacion("",0,0,0,0,0), new alimentacion("",0,0,0,0,0), new alimentacion("",0,0,0,0,0),};
        return A;
    }
}

public class creativo {
    private jugador Jugador;
    private herramientas_sb sb;

    int Cantidad_Mascotas;
    int Cantidad_Alimentacion;

    mascotas[] Mascotas;
    alimentacion[] Alimentos;

    int Cantidad_Animales=12/*55*/;
    animales[] Animales_Definidos;
    int Cantidad_Comida=3/*18*/;
    comida[] Comida;

    public creativo(){
        Jugador=new jugador();
        sb=new herramientas_sb(Jugador);

        Mascotas=new mascotas[6];

        Alimentos=new alimentacion[5];

        Animales_Definidos=new animales[Cantidad_Animales];
        Comida=new comida[Cantidad_Comida];
    }

    public void inicio() throws InterruptedException{
        Scanner entrada=new Scanner(System.in);
        int Opcion=0, Ronda=1;
        Cantidad_Mascotas=0;
        Cantidad_Alimentacion=0;

        Comida=sb.Comida_Especificados();
        Alimentos=sb.LlenarC();
```

```
Mascotas=sb.Llenar();
Animales_Definidos= sb.Animales_Especificados();

Jugador.Oro_Creativo();
System.out.println("-------MODO CREATIVO-------");
System.out.println("--En este espacio podra crear un dream team");
System.out.println("--Para que luche con otros jugadores--");
System.out.println("--Se le dara una dotacion de: "+Jugador.getOro()+"--");
while(Jugador.Muerto_Vivo()){
   Jugador.Mostrar_Datos();
   System.out.println("--Cuando este preparado puche la opcion batalla--");
   System.out.println(" 1) Clasificacion de animales");
   System.out.println(" 2) Comprar animal");
   System.out.println(" 3) Comprar comida");
   System.out.println(" 4) Mirar Mascotas");
   System.out.println(" 5) Batalla");
   Opcion=entrada.nextInt();
   if(Opcion==1){
   System.out.println("  1) Clasificacion por Tier");
   System.out.println("  2) Clasificacion por Vida");
   System.out.println("  3) Clasificacion por Agravio");
   int Opcion_Dos=entrada.nextInt();
      if(Opcion_Dos==1){
         int Orden;
         do{
            Orden=Tipo_Orden();
         }while(Orden==0);
         Por_Tier(Orden);
      }else if(Opcion_Dos==2){
         int Orden;
         do{
            Orden=Tipo_Orden();
         }while(Orden==0);
         Por_Vida(Orden);
      }else if(Opcion_Dos==3){
         int Orden;
         do{
            Orden=Tipo_Orden();
         }while(Orden==0);
         Por_Agravio(Orden);
      }
   }else if(Opcion==2){
      System.out.println("------------------------------------------------------------");
      System.out.println("--Escriba el Nombre del animal con su inicial mayuscula--");
      System.out.println("--Que animal desea comprar? --");
      String Animal=entrada.next();
      // se le envia el valor de la posicion en que este el animal en tienda
      //y la cantidad de animales que hay en tienda
      Comprar_Animal(Animal, Mascotas);
   }else if(Opcion==3){
```

```
            sb.Mostrar_Tienda(Comida, Cantidad_Comida);
            System.out.println("----------------------------");
            System.out.println("--Que comida desea comprar?--");
            int C=entrada.nextInt();
            Comprar_Comida(Mascotas, C-1, Cantidad_Comida, Comida, Alimentos);
        }else if(Opcion==4){
            sb.Menu_Mascotas(Jugador, Cantidad_Mascotas, Mascotas, Alimentos);
        }else if(Opcion==5){
            Jugador.Oro_Creativo();
            Ronda+=1;
        }else if(Opcion==0){
            break;
        }else{
            System.out.println("------------------------");
            System.out.println("--Esa no es una opcion--");
        }
    }


}

public int Tipo_Orden(){
    Scanner entrada =new Scanner(System.in);
    System.out.println("Desea ver el orden en forma...");
    System.out.println("  1) De menor a mayor");
    System.out.println("  2) De mayor a menor");
    int Orden=entrada.nextInt();
    if(Orden==1||Orden==2){
    return Orden;
    }else{
        System.out.println("Esa no es una opcion");
        Orden=0;
        return Orden;
    }

}

public void Por_Tier(int Orden) throws InterruptedException{
    if(Orden==1){
        for(int i=0;i<Animales_Definidos.length;i++){
            for(int j=1;j<Animales_Definidos.length-i;j++){
                if(Animales_Definidos[j-1].getTier()>Animales_Definidos[j].getTier()){
                    animales temp=Animales_Definidos[j];
                    Animales_Definidos[j]=Animales_Definidos[j-1];
                    Animales_Definidos[j-1]=temp;

                }
            }
        }
        for(int i=0;i<Animales_Definidos.length;i++){
            System.out.println((i+1)+")---------------------------------------");
```

```
            Animales_Definidos[i].Mostrar();
            if(i>0&&i%3==0){
                System.out.println("Espere...");
                Thread.sleep(2000);
            }
        }
    }else if(Orden==2){
        for(int i=0;i<Animales_Definidos.length;i++){
            for(int j=1;j<Animales_Definidos.length-i;j++){
                if(Animales_Definidos[j-1].getTier()<Animales_Definidos[j].getTier()){
                    animales temp=Animales_Definidos[j];
                    Animales_Definidos[j]=Animales_Definidos[j-1];
                    Animales_Definidos[j-1]=temp;

                }
            }
        }
        for(int i=Animales_Definidos.length-1;i>=0;i--){
            System.out.println((12-i)+")---------------------------------------");
            Animales_Definidos[i].Mostrar();
            if(i>0&&i%3==0){
                System.out.println("Espere...");
                Thread.sleep(2000);
            }
        }
    }
}

public void Por_Vida(int Orden) throws InterruptedException{
    if(Orden==1){
        for(int i=0;i<Animales_Definidos.length;i++){
            for(int j=1;j<Animales_Definidos.length-i;j++){
                if(Animales_Definidos[j-1].getVida()>Animales_Definidos[j].getVida()){
                    animales temp=Animales_Definidos[j];
                    Animales_Definidos[j]=Animales_Definidos[j-1];
                    Animales_Definidos[j-1]=temp;

                }
            }
        }
        for(int i=0;i<Animales_Definidos.length;i++){
            System.out.println((i+1)+")---------------------------------------");
            Animales_Definidos[i].Mostrar();
            if(i>0&&i%3==0){
                System.out.println("Espere...");
                Thread.sleep(2000);
            }
        }

    }else if(Orden==2){
        for(int i=0;i<Animales_Definidos.length;i++){
```

```
            for(int j=1;j<Animales_Definidos.length-i;j++){
                if(Animales_Definidos[j-1].getVida()<Animales_Definidos[j].getVida()){
                    animales temp=Animales_Definidos[j];
                    Animales_Definidos[j]=Animales_Definidos[j-1];
                    Animales_Definidos[j-1]=temp;

                }
            }
        }
        for(int i=0;i<Animales_Definidos.length;i++){
            System.out.println((i+1)+")-------------------------------------");
            Animales_Definidos[i].Mostrar();
            if(i>0&&i%3==0){
                System.out.println("Espere...");
                Thread.sleep(2000);
            }
        }
    }
}

public void Por_Agravio(int Orden) throws InterruptedException{
    if(Orden==1){
        for(int i=0;i<Animales_Definidos.length;i++){
            for(int j=1;j<Animales_Definidos.length-i;j++){
                if(Animales_Definidos[j-1].getAgravio()>Animales_Definidos[j].getAgravio()){
                    animales temp=Animales_Definidos[j];
                    Animales_Definidos[j]=Animales_Definidos[j-1];
                    Animales_Definidos[j-1]=temp;

                }
            }
        }
        for(int i=0;i<Animales_Definidos.length;i++){
            System.out.println((i+1)+")-------------------------------------");
            Animales_Definidos[i].Mostrar();
            if(i>0&&i%3==0){
                System.out.println("Espere...");
                Thread.sleep(2000);
            }
        }

    }else if(Orden==2){
        for(int i=0;i<Animales_Definidos.length;i++){
            for(int j=1;j<Animales_Definidos.length-i;j++){
                if(Animales_Definidos[j-1].getAgravio()<Animales_Definidos[j].getAgravio()){
                    animales temp=Animales_Definidos[j];
                    Animales_Definidos[j]=Animales_Definidos[j-1];
                    Animales_Definidos[j-1]=temp;

                }
            }
```

```
            }
            for(int i=0;i<Animales_Definidos.length;i++){
                System.out.println((i+1)+")-------------------------------------");
                Animales_Definidos[i].Mostrar();
                if(i>0&&i%3==0){
                    System.out.println("Espere...");
                    Thread.sleep(2000);
                }
            }
        }
    }

    public void Comprar_Animal(String Animal, mascotas Mascotas[]){
        int Existencia=5, Animal_Tienda=99, Vacio=0, Nivel=0;
        // si la seleccion del animal en tienda es coherente y
        //si se tiene el oro suficiente se procede con la primera sentencia de seleccion
        for(int i=0; i<Animales_Definidos.length;i++){
            if(Animales_Definidos[i].getNombre().equals(Animal)){
                Animal_Tienda=i;
                break;
            }
        }


if(Cantidad_Mascotas<5&&(Animal_Tienda>=0&&Animal_Tienda<=Animales_Definidos.length)&&(Jugad
or.getOro()>=3)){
            if(Cantidad_Mascotas>0){
                for(int i=0; i<Cantidad_Mascotas;i++){
                    if(Animales_Definidos[Animal_Tienda].getNombre().equals(Mascotas[i].getNombre())){
                        Existencia=i;
                        break;
                    }
                }
                for(int i=0; i<Cantidad_Mascotas;i++){
                    if(Mascotas[i].getNivel()==3){
                        Nivel=i;
                        break;
                    }
                }
            }
            for (int i=0; i<Cantidad_Mascotas;i++) {
                if ("".equals(Mascotas[i].getNombre()))Vacio=i;
            }

if(Cantidad_Mascotas>0&&Animales_Definidos[Animal_Tienda].getNombre()==(Mascotas[Existencia].get
Nombre())&&Mascotas[Nivel].getNivel()!=3){
                Scanner entrada=new Scanner(System.in);
                System.out.println("--Ya tiene una mascota de esta clase--");
                System.out.println("--Desea combinarlo?--");
                System.out.println("  1)Si");
                System.out.println("  2)No");
```

```
            int Sub_Menu=entrada.nextInt();

            if(Sub_Menu==1){
               sb.Sumar_Mascotas(Existencia, Animal_Tienda, Mascotas, Animales_Definidos);
            }else{
               sb.Compra_A(Cantidad_Mascotas, Animal_Tienda, Mascotas, Animales_Definidos);
               Cantidad_Mascotas=Cantidad_Mascotas+1;
            }

         }else if(Cantidad_Mascotas>0&&("".equals(Mascotas[Vacio].getNombre()))){
            sb.Compra_A(Vacio, Animal_Tienda, Mascotas, Animales_Definidos);

         }else{
            sb.Compra_A(Cantidad_Mascotas, Animal_Tienda, Mascotas, Animales_Definidos);
            Cantidad_Mascotas=Cantidad_Mascotas+1;
         }
         // se sobre escribe la tienda para eliminar el animal comprado omitiento
         //la posicion 5 ya que es de tipo null, debo de cambiar eso

         Jugador.Quitar_Oro(3);


      }else if(Cantidad_Mascotas==5){
         for (int i=0; i<Cantidad_Mascotas;i++) {
            if ("".equals(Mascotas[i].getNombre())) {
               sb.Compra_A(i, Animal_Tienda, Mascotas, Animales_Definidos);
               break;
            }else if(i==4){
               System.out.println("----------------------------");
               System.out.println("Perdon pero no tiene espacio");
            }
         }

      }else if(Jugador.getOro()<3){
         System.out.println("-----------------------");
         System.out.println("No tiene suficiente oro");

      }else {
         System.out.println("-------------------------------------");
         System.out.println("--Escribio mal el nombre del animal,--");
         System.out.println("--O esa no es una opcion--");

      }

   }

   public void Comprar_Comida(mascotas M[], int Comida, int Cantidad_TiendaC, comida C_T[],
alimentacion A[]){
      Scanner entrada=new Scanner(System.in);
      if(Comida>=0&&Comida<2&&Jugador.getOro()>3){
         if(C_T[Comida].getTipo_Efecto()==1){
```

```
        for(int i=0;i<Animales_Definidos.length;i++){
            Animales_Definidos[i].setVida(C_T[Comida].getDar_Vida());
            Animales_Definidos[i].setVida(C_T[Comida].getDar_Agravio());
        }
    }else if(C_T[Comida].getTipo_Efecto()==1.1){
        System.out.println("--Ahora no joven plisss--");
        Jugador.Quitar_Oro(3);
    }else if(C_T[Comida].getTipo_Efecto()>1.1){
        System.out.println("--A que mascota le quiere dar la comida?--");
        int Mascota=entrada.nextInt()-1;
        if(M[Mascota].getNombre()!=""){
            if(C_T[Comida].getTipo_Efecto()==1.2){
                M[Mascota].setVida(C_T[Comida].getDar_Vida());
                M[Mascota].setAgravio(C_T[Comida].getDar_Agravio());
                M[Mascota].setNivel(C_T[Comida].getDar_Nivel());
                for(int i=(Comida);i<C_T.length-1;i++){
                        C_T[i]=C_T[i+1];
                }
                Jugador.Quitar_Oro(3);
            }if(C_T[Comida].getTipo_Efecto()>1.2){
                if(M[Mascota].getEfecto()!=5){
                    System.out.println("Comida almacenada: "+M[Mascota].getEfecto());
                    System.out.println("--Desea sustitur el alimento anterior?--");
                    System.out.println("   1)Si");
                    System.out.println("   2)No");
                    int Opcion=entrada.nextInt();

                    if(Opcion==1){
                        sb.Comprar_C(Cantidad_Alimentacion, Comida, C_T, A);
                        Jugador.Quitar_Oro(3);
                    }
                }else if(M[Mascota].getEfecto()==5){
                    //tengo que mandar a sobre escribir a alimentacion
                    sb.Comprar_C(Cantidad_Alimentacion, Comida, C_T, A);
                    M[Mascota].SetEfecto(Comida);
                    Cantidad_Alimentacion=Cantidad_Alimentacion+1;
                    Jugador.Quitar_Oro(3);
                }else{
                    System.out.println("--Perdon pero tendra de escoger --");
                    System.out.println("--otra vez la comida y el aminal--");
                }
            }
        }else{
            System.out.println("--No tiene mascota en esa posicion--");
        }
    }
}else if(Jugador.getOro()<3){
    System.out.println("---------------------------");
    System.out.println("--No tiene suficiente oro--");
}else{
    System.out.println("------------------------");
```

```
            System.out.println("--Esa no es una opcion--");
        }
    }

}


public class sujeto {
    protected int Oro;

    public sujeto(){
        Oro=10;
    }

    public int getOro(){
        return Oro;
    }

    public void Oro_Ronda() {
        Oro=10;
    }

    public void Agregar_Oro(int Agregar_Oro){
        Oro=Oro+Agregar_Oro;
    }

    public void Quitar_Oro(int Quitar_Oro){
        int Oro_Final=Oro-Quitar_Oro;
        Oro_Final=Oro_Final >= 0 ? Oro_Final: 0;
        Oro = Oro_Final;
    }

    public boolean Bancarrota(){
        return Oro==0;
    }
}

public class jugador extends sujeto {
    private int Corazones;
    private int Copas;


    public jugador(){
        Corazones=10;
        Copas=0;
    }

    public void Quitar_Corazones(int Quitar_Corazones){
        int Corazones_Final=Corazones-Quitar_Corazones;
        Corazones_Final=Corazones_Final >= 0 ? Corazones_Final: 0;
        Corazones -= Corazones_Final;
```

```
    }

    public boolean Muerto_Vivo(){
        return Corazones>=0;
    }

    public void Ganar_Copas(int Mas_Copas){
        Copas=Copas+Mas_Copas;
    }

    public void Mostrar_Datos(){
        System.out.println("---------------------------------------");
        System.out.println("|| Oro: "+Oro+"|| Vida: "+Corazones+"||");
        System.out.println("---------------------------------------");
    }

    public void Oro_Creativo() {
        Oro=150;
    }
}

public class animales implements Cloneable{
    protected String Nombre;
    protected double Habilidad;
    protected int Tipo;
    protected int Tipo_Dos;
    protected int Tipo_Tres;
    protected int Tier;
    protected double Nivel;
    protected int Vida;
    protected int Agravio;
    protected int Efecto;

    public animales(String Nombre, double Habilidad, int Tipo, int Tipo_Dos, int Tipo_Tres, int Tier,
                    double Nivel, int Vida, int Agravio, int Efecto) {
        this.Nombre = Nombre;
        this.Habilidad=Habilidad;
        this.Tipo = Tipo;
        this.Tipo_Dos = Tipo_Dos;
        this.Tipo_Tres = Tipo_Tres;
        this.Tier = Tier;
        this.Nivel = Nivel;
        this.Vida = Vida;
        this.Agravio = Agravio;
        this.Efecto=Efecto;
    }

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

```
public double getHabilidad(){
    return Habilidad;
}

public int getTipo(){
    return Tipo;
}

public int getTipo_Dos(){
    return Tipo_Dos;
}

public int getTipo_Tres(){
    return Tipo_Tres;
}

public int getTier(){
    return Tier;
}

public double getNivel(){
    return Nivel;
}

public int getVida(){
    return Vida;
}

public int getAgravio(){
    return Agravio;
}

public void setNombre(String N){
    Nombre=N;
}

public void setTipo(int T){
    Tipo=T;
}

public void setTipo_Dos(int T_Dos){
    Tipo_Dos=T_Dos;
}

public void setTipo_Tres(int T_Tres){
    Tipo_Tres=T_Tres;
}

public void setTier(int Ti){
    Tier=Ti;
}
```

```
    public int getEfecto(){
        return Efecto;
    }

    public void SetEfecto(int Efecto){
        this.Efecto=Efecto;
    }

    public void setNivel(double ex){
        Nivel=Nivel+ex;
    }

    public void setVida(int Agregar){
        Vida=Vida+Agregar;
    }

    public void setAgravio(int Agregar){
        Agravio=Agravio+Agregar;
    }

    public void setNivel_Igualar(double ex){
        Nivel=ex;
    }

    public void setVida_Igualar(int Agregar){
        Vida=Agregar;
    }

    public void setAgravio_Igualar(int Agregar){
        Agravio=Agregar;
    }


    public void Mostrar(){
        System.out.println("Tier: "+this.Tier+"--Nombre: " + this.Nombre+"--Nivle: "+this.Nivel+"--Agravio:
"+this.Agravio+"--Vida: "+this.Vida);
        Habilidad_Texto(Nombre);
        System.out.print("Tipo: ");
        Nombre_Tipo(this.Tipo);
        if(this.Tipo_Dos!=0){
            System.out.print("--");
            Nombre_Tipo(this.Tipo_Dos);
        }
        if(this.Tipo_Tres!=0){
            System.out.print("--");
            Nombre_Tipo(this.Tipo_Tres);
        }
        System.out.println("");
        System.out.println("----------------------------------------");
    }
```

```
    public void Habilidad_Texto(String Nombre){
        if("Hormiga".equals(Nombre)){
            System.out.println("Compañerismo: Da a un aliado al azar: (+2/+1)(+4/+2)(+6/+3)");
        }else if("Pescado".equals(Nombre)){
            System.out.println("Level-up: Da a todos los aliados: (+1/+1)(+2/+2)");
        }else if("Mosquito".equals(Nombre)){
            System.out.println("Piquete inicial: Al inicio de la batalla realiza 1 de daño a: 1/2/3 enemigos");
        }else if("Grillo".equals(Nombre)){
            System.out.println("Desmayo: Convoca a un grillo zombie con estadisticas: (1/1)(2/2)(3/3)");
        }else if("Castor".equals(Nombre)){
            System.out.println("Represa: Da a 2 aliados al azar: +1/+2/+3 de HP");
        }else if("Caballo".equals(Nombre)){
            System.out.println("Rugido alidao: Da: +1/+2/+3 de ATK hasta ek final de la batalla");
        }else if("Nutria".equals(Nombre)){
            System.out.println("Compra: Da a un aleado aleatorio: (+1/+1)(+2/+2)(+3/+3)");
        }else if("Escarabajo".equals(Nombre)){
            System.out.println("Come comida de la tienda: Otorga a las mascotas de la tienda: +1/+2/+3 de
HP");
        }else if("Sapo".equals(Nombre)){
            System.out.println("Metamorfosis: Copia la salud del aliado mas saludable");
        }
    }

    public void Nombre_Tipo(int modelo){
        switch (modelo) {
            case 1:
                System.out.print("Insecto");
                break;
            case 2:
                System.out.print("Volador");
                break;
            case 3:
                System.out.print("Acuatico");
                break;
            case 4:
                System.out.print("Terrestre");
                break;
            case 5:
                System.out.print("Reptil");
                break;
            case 6:
                System.out.print("Mamifero");
                break;
            case 7:
                System.out.print("Domestico");
                break;
            case 8:
                System.out.print("Solitario");
                break;
            case 9:
```

```
            System.out.print("Desertico");
            break;
        default:
            break;
    }
}
```

Responsables

@autor José García