Universidad de San Carlos de Guatemala
Centro Universitario de Occidente
División de Ciencias de la Ingeniería
Tercer Semestre
Docente: Ing. Oliver Sierra
Curso: Lab.Introducción a la Programación y Computación 1
Código de Curso: 2796

Manual Técnico - Algoritmos

Estudiante: González Alcántara, Andrés Fernando
No. Carné:202131199

Quetzaltenango, 18 de abril 2022

ESCOGER ARENA

```
publico void elegirArena(String nombre, int codigo){

    //RECIBIREMOS EL TIPO DE MODO
    int numero;
    modoarena arena = new modoarena();
    modoversus versus = new modoversus();
    modocreativo creative = new modocreativo();
    Scanner scan1 = new Scanner(System.in);
    boolean error = false;

    hacer{
    Imprimir("Jugador: "+ nombre ); //SALUDOS JEJE
    Imprimir("\n-----------------------------------------------------");
    Imprimir("                Arena de Batallas                  ");
    Imprimir("-----------------------------------------------------");
    Imprimir("1) Pantano");
    Imprimir("2) Nubes ");
    Imprimir("3) Mar ");
    Imprimir("4) Bosque ");
    Imprimir("5) Granja ");
    Imprimir("6) (sin campo)");
    Imprimir("7) Sabana ");
    Imprimir("8) Salir ");
    Imprimir("---------------Escoja el numero de mapa----------------");
    numero = scan1.nextInt();
    scan1.nextLine();

    switch(numero){
       si se da el caso 1:
          Imprimir("Bienvenido a Arena Pantano");
          romper;
       si se da el caso 2:
          Imprimir("Bienvenido a Arena Nubes");
          romper;
       si se da el caso 3:
          Imprimir("Bienvenido a Arena Mar");
          romper;
       si se da el caso 4:
          Imprimir("Bienvenido a Arena Bosque");
          romper;
       si se da el caso 5:
          Imprimir("Bienvenido a Arena Granja");
          romper;
       si se da el caso 6:
          Imprimir("Bienvenido a Arena sin Campo");
```

```
            romper;
        si se da el caso 7:
            Imprimir("Bienvenido a Arena Sabana");
            romper;
        si se da el caso 8:
        Imprimir("Volviendo al menu principal");
        error = false;
        romper;
        defecto:
            Imprimir("Este mapa no existe");
            error = true;
        romper;
    }

}mientras(error);
    si(numero !=8)
        {
            //leer tipo batalla
            //iniciar Switch
            // si se da el caso 1 ir modo arena
            // si se da el caso 2 ir modo versus
            switch(codigo){
                si se da el caso 1:
                    arena.inicio(numero);
                    romper;
                si se da el caso 2:
                    versus.inicio();
                    romper;
                si se da el caso 3:
                    creative.inicio();
                    romper;
            }
        }
    }
}
```

MODOS DE JUEGO

MENU BATALLAS

```
publico clase menubatallas {
   Scanner in =new Scanner(System.in);
   Random random = new Random();
   objetos Objeto = new objetos();
   int oro=10, ronda=0, opcion = 1;
   int [] mascotas1 = {0,0,0,0,0};
   boolean back,foodon = false;
   publico int [][] irAMenu(int [][] matriz,int round)
   {
      oro=10;
      ronda=round;//convertir en global
      mascotas1 = generarMascota();
      hacer
      {
         Imprimir(" ORO = " + oro);
         Imprimir("-------------- M E N U  D E  B A T A L L A ---------------");
         Imprimir("1)..... C O M P R A R  M A S C O T A S");
         Imprimir("2).....C O M P R A R  C O M I D A");
         Imprimir("3)..... O R D E N A R  M A S C O T A S");
         Imprimir("4)..... U N I R  M A S C O T A S");
         Imprimir("5)..... V E N D E R  M A S C O T A S");
         Imprimir("6)..... B A T A L L A");
         opcion = in.nextInt();
         in.nextLine();
         switch(opcion)
         {
            si se da el caso 1:  matriz = elegirMascotas(matriz); romper;
            si se da el caso 2:  elegirComida(); romper;
            si se da el caso 3:  matriz = ordenarMascotas(matriz); romper;
            si se da el caso 4:  matriz = unirMascotas(matriz);romper;
            si se da el caso 5:  matriz = venderMascotas(matriz); romper;
            si se da el caso 6:
                  si(oro>0)
                  {
                     Imprimir("------------ AUN TIENES ORO ----------");
                     Imprimir("PRESIONA LA OPCION 6  DE NUEVO PARA IR A LA
BATALLA" );
                     opcion = in.nextInt();
                     si (opcion==6)
                     Imprimir("------------------- FUEGO ----------------------------");
                     demas
                     Imprimir("----------- INGRESE UNA NUEVA OPCION  ------------");
                  }
                  demas
```

```
                    Imprimir("------------------- FUEGO ------------------------------");
                romper;
            defecto:    Imprimir("-- ESTA OPCION NO ESTA DISPONIBLE --");
romper;
        }
    }mientras (opcion != 6);

    devolver matriz;
  }
  publico int [][] venderMascotas(int [][] matriz)
  {
    int pet,conpet=0;
    para(int a = 0;a<5;a++)
    si(matriz[a][0]!=0)
    conpet++;
    verEquipo(matriz,true);
    Imprimir("------- INGRESE MASCOTA ----- ");
    pet= in.nextInt();
    pet--;
    si(pet>=0 && pet< conpet)
    {
      oro +=  matriz[pet][3];
      para (int a =0;a<10;a++)
      matriz[pet][a] = 0;
      matriz = ordenEquipo(matriz);

      verEquipo(matriz,true);
    }
    demas
    Imprimir(" POSICION INVALIDA ");
    devolver matriz;
  }
  publico int [][] ordenEquipo (int [][] matriz)
  {
    int f=0;
    mientras(f<4)
    {
      si (matriz[f][0]==0)
      {
        para (int a =0;a<10;a++)
        {
          matriz[f][a] = matriz[f+1][a];
          matriz[f+1][a]=0;
        }

      }
      f++;
```

```
        }

        devolver matriz;
    }
    publico int [][] unirMascotas(int matriz[][])
    {
        boolean fus=false;
        verEquipo(matriz,true);
        int a1,a2,conMasc=0;
        para(int a = 0;a<5;a++)
        si(matriz[a][0]!=0)
        conMasc++;
        Imprimir("INGRESE LA PRIMER MASCOTA");
        a1= in.nextInt();
        Imprimir("INGRESE LA SEGUNDA MASCOTA ");
        a2= in.nextInt();
        a1--;
        a2--;
        si((a1>=0 && a1 < conMasc) && (a2>=0 && a2 < conMasc))
        {
            si( (matriz[a1][4]!=0) && ( (matriz[a1][4]==matriz[a2][4]) ||
(matriz[a1][4]==matriz[a2][5]) || (matriz[a1][4]==matriz[a2][6])) )
            fus=true;
            demas si( (matriz[a1][5]!=0) && ((matriz[a1][5]==matriz[a2][5]) ||
(matriz[a1][5]==matriz[a2][6])) )
            fus=true;
            demas si( (matriz[a1][6]!=0) && (matriz[a1][6]==matriz[a2][6]))
            fus= true;
            demas
            Imprimir("\n ---- UNIDOS XD ----");


            si(fus)
            {
                matriz[a1][1] += 1;
                matriz[a1][2] += 1;
                matriz[a1][7] += 1;
                matriz[a1][9] += matriz[a2][9];

                matriz[a2][0] =0;
                matriz[a2][1] =0;
                matriz[a2][2] =0;
                matriz[a2][3] =0;
                matriz[a2][4] =0;
                matriz[a2][5] =0;
                matriz[a2][6] =0;
                matriz[a2][7] =0;
```

```
            matriz[a2][8] =0;
            matriz[a2][9] =0;

            si(matriz[a1][7]>=5)
            {

               matriz[a1][3]=3;
            }
            demas si(matriz[a1][7]>=2)
            {
               matriz[a1][3]=2;
            }

         }
         ordenEquipo(matriz);
         verEquipo(matriz,true);
         }
      demas
      {
         Imprimir(" POSICION INVALIDA ");

      }


      devolver matriz;
   }
   publico int [][] elegirMascotas(int [][] matriz)
   {
      int codepet,codearray;

      hacer
      {
         back = false;
         ordenArreglo();
         para (int c=0;c<5;c++)
         {
            si (mascotas1[c]!=0)
            back=true;
         }
         si (back)
         {
            verMascota(mascotas1);
            Imprimir("----------  Ingrese el codigo de la mascota para agregarlo al
equipo -------");
            Imprimir("---- Ingrese 0 para regresar al menu anterior----");
            codearray = in.nextInt();
            in.nextLine();
```

```
            codearray--;
            si (codearray>=0&&codearray<=4)
            {
                si (mascotas1[codearray]!=0)
                {
                    codepet = mascotas1[codearray];
                    matriz = agregarMascota(matriz, codepet,codearray,true);
                    verEquipo(matriz,true);

                }
                demas{
                    Imprimir("---------- LA MASCOTA NO FUE ENCONTRADA ---------");
                }
            }
            demas si(codearray<0)
            {
                back=false;
            }
            demas
            {
                Imprimir("---------- LA MASCOTA NO FUE ENCONTRADA ---------");
            }
        }
        demas{
            Imprimir("\n------------------------------------------------------------");
            Imprimir("------------- NINGUNA MASCOTA DISPONIBLE ------------");
            Imprimir("------------------------------------------------------------\n");
        }

    }mientras ((oro>0) && (back));
    devolver matriz;
}
publico void verEquipo (int matriz[][],boolean team)
{
    si(team)
    {
        Imprimir("\n------------------------------------------------------------");
        Imprimir("------------------ TU EQUIPO ES -----------------");
        Imprimir("------------------------------------------------------------\n");
    }
    demas{
        Imprimir("\n------------------------------------------------------------");
        Imprimir("--------------- EL EQUIPO DE TU OPONENTE ES --------------");
        Imprimir("------------------------------------------------------------\n");
    }
    para (int c=0;c<5;c++)
    {
```

```
            si (matriz[c][0]!=0)
            {
                verDatosE(matriz[c][0],
c,matriz[c][9],matriz[c][1],matriz[c][2],matriz[c][3],false);
            }
        }
        Imprimir("\n-----------------------------------------------------------");
        Imprimir("-----------------------------------------------------------\n");
    }
    publico void verMascota(int pets [])
    {
        Imprimir("---------------- MASCOTAS DISPONIBLES ----------------");
        para (int cont = 0; cont <5; cont++)
        {
            si (pets[cont] != 0)
            {
                verDatosE(pets[cont],cont,0,0,0,0,true);
            }
        }

    }
    publico void verDatosE(int codigo, int num, int exp, int life, int att, int level,
boolean pet)
    {
        int Ndates [] = Objeto.getNdates(codigo);//{getDefensa(), getAtaque(),
getVida()};
        String Sdates [] =
Objeto.getSdates(codigo);//{getNombre(),getTipo1(),getTipo2(),getTipo3()};
        num++;
        si (pet)
        {
            Imprimir("No.   Nombre     Ataque    VIDA  EXP   TIPO1     TIPO2
TIPO3");
            Imprimir(num+ "  " +Sdates[0] + "      " + Ndates[1] + "        " + Ndates[2] + "
" +exp +"   "+ Sdates[1] + " " + Sdates[2] + " " + Sdates[3]);
        }
        demas
        {
            Imprimir("No.   Nombre     Ataque    VIDA  EXP   TIPO1     TIPO2
TIPO3");
            Imprimir(num+ "  " +Sdates[0] + "      " + att+ "        " + life + "   " +exp +"
"+ Sdates[1] + " " + Sdates[2] + " " + Sdates[3]);
        }
    }
    publico int [][] agregarMascota(int matriz [][], int codepet, int codearray,boolean
team)
    {
```

```
         boolean exito = false;
        si (team)
        {

            si (oro>=2)
            {
                para (int f=0;f<5;f++)
                {
                    si (matriz[f][0]==0 && exito==false)
                    {//{getDefensa(), getAtaque(), getVida()};
{getNombre(),getTipo1(),getTipo2(),getTipo3()};
                        int []Ndates = Objeto.getNdates(codepet);
                        String []Sdates = Objeto.getSdates(codepet);
                        matriz[f][0]= codepet;
                        matriz[f][1] = Ndates [2];
                        matriz[f][2] = Ndates [1];
                        matriz[f][3] = 1;
                        matriz[f][4] = convertir(Sdates[1]);
                        matriz[f][5] = convertir(Sdates[2]);
                        matriz[f][6] = convertir(Sdates[3]);
                        matriz[f][7] = 1;
                        matriz[f][8] = 0;
                        matriz[f][9] = 0;
                        /*si (foodon)
                        sumar*/
                        oro-=2;
                        mascotas1[codearray]=0;
                        exito=true;
                    }
                }
            }
            demas
            {
                Imprimir("\n---------- EL ORO ES ESCAZO----------");
                back = false;
            }
            si(!exito)
            {
                Imprimir("\n- NO SE PUEDEN AGREGAR MAS MASCOTAS -");
                back = false;
            }
        }
        demas{
            para (int f=0;f<5;f++)
                {
                    si (matriz[f][0]==0 && !exito)
```

```
                {//{getDefensa(), getAtaque(), getVida()};
{getNombre(),getTipo1(),getTipo2(),getTipo3()};
                    int []Ndates = Objeto.getNdates(codepet);
                    String []Sdates = Objeto.getSdates(codepet);
                    matriz[f][0]= codepet;
                    matriz[f][1] = Ndates [2];
                    matriz[f][2] = Ndates [1];
                    matriz[f][3] = 1;
                    matriz[f][4] = convertir(Sdates[1]);
                    matriz[f][5] = convertir(Sdates[2]);
                    matriz[f][6] = convertir(Sdates[3]);
                    matriz[f][7] = 1;
                    matriz[f][8] = 0;
                    matriz[f][9] = 0;
                    /*si (foodon)
                    sumar*/
                    mascotas1[0]=0;
                    exito=true;
                }
            }
        }
        devolver matriz;
    }
    publico void ordenArreglo()
    {
        int f=0;
        mientras(f<4)
        {
            si (mascotas1[f]==0)
            {
                mascotas1[f] = mascotas1[f+1];
                mascotas1[f+1]=0;
            }
            f++;
        }
    }
    publico int convertir (String ty)
    {
        int n=0;
        switch(ty)
        {
        si se da el caso "Insecto" :   n=1; romper;
        si se da el caso "Volador" :   n=2; romper;
        si se da el caso "Acuatico" :   n=3; romper;
        si se da el caso "Terrestre" :   n=4; romper;
        si se da el caso "Reptil" :   n=5; romper;
        si se da el caso "Mamifero" :   n=6; romper;
```

```
    si se da el caso "Domestico" :   n=7; romper;
    si se da el caso "Solitario" :   n=8; romper;
    si se da el caso "Desertico" :   n=9; romper;
    defecto: n=0; romper;
    }

    devolver n;
}
publico void elegirComida()
{

}
publico int [][] ordenarMascotas(int matriz[][])
{
    int [][] matriz2 = new int [5][10];
    verEquipo(matriz,true);
    int [] posicion=new int [5];
    int numP,contMasc=0,error=0;
    para(int a = 0;a<5;a++)
    si(matriz[a][0]!=0)
    contMasc++;

    para (int f =0;f<contMasc ; f++)
    {
        Imprimir("---- INGRESE NUEVA POSICION DE MASCOTA "+ (f+1)+ " ---");
        numP = in.nextInt();
        numP--;
        si(numP>=0 && numP < contMasc)
        {
            error=0;
            si(f!=0)
            {
                para(int fr = f;fr>0;fr--)
                {
                    si(numP==posicion[fr-1])
                    {
                        error++;
                    }
                }
                si(error==0)
                {
                    posicion[f]=numP;
                }
                demas
                {
                    Imprimir("------------ POSICION DUPLICADA ----------");
                    f--;
```

```
                }
            }
            demas
            {
                posicion[f]=numP;
            }
        }
        demas
        {
            Imprimir("------------ POSICION INVALIDA --------------");
            f--;
        }
    }
    matriz2 = cambiarOrden(matriz, matriz2, posicion, contMasc);
    verEquipo(matriz2,true);
    devolver matriz2;
}
publico int [][] cambiarOrden(int [][] matriz1, int [][]matriz2,int order[],int
cantmasc)
{
    para (int f=0;f<cantmasc;f++)
    {
        matriz2[order[f]][0]=matriz1[f][0];
        matriz2[order[f]][1]=matriz1[f][1];
        matriz2[order[f]][2]=matriz1[f][2];
        matriz2[order[f]][3]=matriz1[f][3];
        matriz2[order[f]][4]=matriz1[f][4];
        matriz2[order[f]][5]=matriz1[f][5];
        matriz2[order[f]][6]=matriz1[f][6];
        matriz2[order[f]][7]=matriz1[f][7];
        matriz2[order[f]][8]=matriz1[f][8];
        matriz2[order[f]][9]=matriz1[f][9];

    }

    devolver matriz2;
}
publico int [][] generarOp(int [][]matriz,int round)
{
    ronda=round;
    int matrizop[][] = new int[5][10];
    int petres[]={0,0,0,0,0};
    int conpet = 0;
    para(int a = 0;a<5;a++)
    {
        matrizop[a][0]=0;
        si(matriz[a][0]!=0)
```

```
        conpet++;
    }
    mascotas1 = generarMascota();
    petres = generarMascota();
    mascotas1 [3] = petres [1];
    mascotas1 [4] = petres [2];
    si (conpet==5)
    conpet--;
    para(int f = 0;f<=conpet;f++)
    {
        ordenArreglo();
        int codepet = mascotas1[0];
        matrizop = agregarMascota(matrizop, codepet, 0, false);
    }
    devolver matrizop;
}
publico int [] generarMascota()
{
    int n1=0,n2=0,n3=0,n4=0,n5=0;
    switch(ronda)
    {
        si se da el caso 1,2,3:
            si (ronda==1)//tier 1
            {
                n1=random.nextInt(8-1)+1;
                n2=random.nextInt(8-1)+1;
                n3=random.nextInt(8-1)+1;
            }
            demas{//tier2
                n1=random.nextInt(16-1)+1;
                n2=random.nextInt(16-1)+1;
                n3=random.nextInt(16-1)+1;
            }
            romper;
        si se da el caso 4,5,6://4animales, 4 y 5 tier 3, 6 tier 4
            si (ronda==6) //tier 4
            {
                n1=random.nextInt(35-20)+20;
                n2=random.nextInt(35-20)+20;
                n3=random.nextInt(35-20)+20;
                n4=random.nextInt(35-20)+20;
            }
            demas{//tier 3
                n1=random.nextInt(27-1)+1;
                n2=random.nextInt(27-1)+1;
                n3=random.nextInt(27-1)+1;
                n4=random.nextInt(27-1)+1;
```

```
            }
            romper;
        si se da el caso 7:
        defecto : //5animales
            si (ronda==7)//tierr 4
            {
                n1=random.nextInt(35-20)+20;
                n2=random.nextInt(35-20)+20;
                n3=random.nextInt(35-20)+20;
                n4=random.nextInt(35-20)+20;
                n5=random.nextInt(35-20)+20;
            }
            demas si(ronda <=9)//tier 5
            {
                n1=random.nextInt(43-30)+30;
                n2=random.nextInt(43-30)+30;
                n3=random.nextInt(43-30)+30;
                n4=random.nextInt(43-30)+30;
                n5=random.nextInt(43-30)+30;
            }
            demas si (ronda<=11)//tier 6
            {
                n1=random.nextInt(52-40)+40;
                n2=random.nextInt(52-40)+40;
                n3=random.nextInt(52-40)+40;
                n4=random.nextInt(52-40)+40;
                n5=random.nextInt(52-40)+40;
            }
            demas //tier 7
            {
                n1=random.nextInt(54-1)+1;
                n2=random.nextInt(54-1)+1;
                n3=random.nextInt(54-1)+1;
                n4=random.nextInt(54-1)+1;
                n5=random.nextInt(54-1)+1;
            }

    }

    devolver new int [] {n1,n2,n3,n4,n5};
}
publico int [] generarComida()
{
    int n1=0,n2=0;
    switch(ronda)
    {
        si se da el caso 1,2,3:
```

```
        si (ronda==1)//tier 1
        {
            n1=in.nextInt(3-1)+1;
            n2=in.nextInt(3-1)+1;
        }
        demas{//tier2
            n1=in.nextInt(6-1)+1;
            n2=in.nextInt(6-1)+1;
        }
        romper;
    si se da el caso 4,5,6:
        si (ronda==6) //tier 4
        {
            n1=in.nextInt(13-1)+1;
            n2=in.nextInt(13-1)+1;
        }
        demas{//tier 3
            n1=in.nextInt(10-1)+1;
            n2=in.nextInt(10-1)+1;
        }
        romper;
    si se da el caso 7:
    defecto:
        si (ronda==7)//tierr 4
        {
            n1=in.nextInt(13-1)+1;
            n2=in.nextInt(13-1)+1;
        }
        demas si(ronda <=9)//tier 5
        {
            n1=in.nextInt(15-1)+1;
            n2=in.nextInt(15-1)+1;
        }
        demas si (ronda<=11)//tier 6
        {
            n1=in.nextInt(17-1)+1;
            n2=in.nextInt(17-1)+1;
        }
        demas //tier 7
        {
            n1=in.nextInt(18-1)+1;
            n2=in.nextInt(18-1)+1;
        }

}

devolver new int [] {n1,n2};
```

```
    }
}
```

MODO ARENA

```
publico clase modoarena extends objetos
{
   Random random = new Random();
   objetos Object = new objetos();
   menubatallas menuBatalla = new menubatallas();
   int estado=0, ronda=0, arenaCJ=0, estado2=0;
   int equipo1[][] = new int [5][10];
   int equipo2[][] = new int [5][10];
   int t1r[][] = new int [5][10];
   int t2r[][] = new int [5][10];
   int equipoAtaque [][] = new int [5][10];
   int equipoDefensa [][] = new int [5][10];
   int cont;
   boolean haymuertos=false;

   publico void inicio(int arena)
   {
      arenaCJ = arena;
      Imprimir("\n------------------------------------------------------------");
      Imprimir("----------------------   MODO ARENA   ----------------------");
      Imprimir("------------------------------------------------------------\n");


      hacer{
         estado2=0;
         ronda++;
         equipo1 = menuBatalla.irAMenu(equipo1,ronda);
         equipo2 = menuBatalla.generarOp(equipo1,ronda);
         para (int f = 0;f<5;f++)
         {
            para (int c =0;c<10;c++)
            {
               t1r[f][c] = equipo1 [f][c];
               t2r[f][c] = equipo2 [f][c];
            }
         }
         verEquipos();
         inicioRonda();
         batalla();
         //finishround();
      }mientras (estado==0);
   }

   publico void inicioRonda()
   {
      habilidadInicial(1);
```

```
    habilidadInicial(2);
    verEquipos();
    hacer {
        haymuertos=false;
        BuscarSiHayMuertes(1);
        BuscarSiHayMuertes(2);
        revisarMuertes();
    }mientras(haymuertos);
}

publico void revisarMuertes()
{
    para (int f=0;f<5;f++)
    {
        si(t1r[f][0]!=0 && t1r[f][1]<=0)
        haymuertos=true;
        si(t2r[f][0]!=0 && t2r[f][1]<=0)
        haymuertos=true;
    }

}

publico void verEquipos()
{
    menuBatalla.verEquipo(t1r, true);
    menuBatalla.verEquipo(t2r, false);
}

publico void batalla()
{
    hacer
    {
        turnos(1);
        turnos(2);
        hacer {
            haymuertos=false;
            BuscarSiHayMuertes(1);
            BuscarSiHayMuertes(2);
            revisarMuertes();
        }mientras(haymuertos);
        verEquipos();
        estado();
    }mientras(estado2==0);
}

publico void estado()
{
```

```
    boolean ht1=false, ht2=false;
    para (int f=0;f<5;f++)
    {
        si(t1r[f][0]!=0)//todavia tiene jugadores el equipo 1, no ha perdido
        ht1 = true;
        si(t2r[f][0]!=0)//todavia tiene jugadores el equipo 2, no ha ganado
        ht2 = true;
    }
    si (!ht2 && ht1)// si todavia tiene jugadores y el otro no, gano
    estado2=1;
    si (!ht1 && ht2)// si no tiene jugadores y el segundo si, perdio
    estado2=2;
    si (!ht1 && !ht2)// si ninguno tiene, empate
    estado2 =3;
}

publico void turnos(int turn)
{
    int attack= equipoAtaque[0][2];
    granturn(turn);
    ordenEquipo(true);
    ataque(0, 0, attack, turn);
    restoreturn(turn);
}

publico void terminarRonda()
{

}

publico void BuscarSiHayMuertes (int turn)
{
    String nombre ="";
    granturn(turn);
    int cont=0;
    para (int cnt =0;cnt <5;cnt ++)
    {
        si (equipoAtaque[cnt][0]!=0)
        {
            si (equipoAtaque[cnt][1]<=0)
            {
                si(cont==0)
                Imprimir("EQUIPO "+turn+": ");
                nombre = Object.getNombre(equipoAtaque[cnt][0]);
                Imprimir(nombre + " HA FALLECIDO");
                danioAmigo(cnt);
                muerteHabilidad(equipoAtaque[cnt][0],cnt);
```

```
                cont++;
            }
        }
    }
    ordenEquipo(true);
    restoreturn(turn);
}

publico void danioAmigo(int posatt)
{
    si(posatt<4)
    {
        posatt++;
    }
    si(equipoAtaque[posatt][0]==25)
    {
        int level= equipoAtaque[posatt][3];
        level *= 2;
        equipoAtaque[posatt][1] += 20;
        equipoAtaque[posatt][2] += level;
        Imprimir("MEJOR ME PROTEJO: BUEY HA GANADO MELON ARMOUR
Y " + level + " DE ATAQUE");
    }
}

publico void muerteHabilidad (int codigoDeAtaque, int posicionAtaque)
{
    int level = equipoAtaque[posicionAtaque][3], aliado;
    String nameatt = Object.getNombre(codigoDeAtaque);
    switch (codigoDeAtaque)
    {
        si se da el caso 1: //hormiga da a aliado
        boolean error=false;
        int cantderror=0;
        hacer{
            aliado = generarNumAleatorio(4, 0);
            si (aliado == posicionAtaque)
                error=true;
            si (equipoAtaque[aliado][0]==0 || equipoAtaque[aliado][1]<=0)
                error= true;
                cantderror++;
            si (!error)
            {
                equipoAtaque[aliado][2]+=level *2;
                equipoAtaque[aliado][1]+=level;
            }
            si (cantderror ==10)
```

```
            error = false;
        }mientras (error);
            eliminarMascota(posicionAtaque);
            Imprimir("HORMIGA HA DADO A ALIADO" + (level *2) + "/" + level );
        romper;
        si se da el caso 4: // grillo convoca a un grillo zombi
            invocar(codigoDeAtaque,posicionAtaque, level,55);
            romper;
        si se da el caso 12: level = level * 2;
            Imprimir("PUERCO ESPIN  HIZO "+ level + " DE  DAÑO A TODAS LAS
MASCOTAS");
            para (int f =0; f<5;f++)
            {
                //equipo 1
                si (posicionAtaque!=0)
                {
                    equipoAtaque[f][1] -= level;
                }
                equipoDefensa[f][1] -= level;
                eliminarMascota(posicionAtaque);
            }
            romper;
        si se da el caso 14:
        invocar(codigoDeAtaque,posicionAtaque, level,56);
            romper;
        si se da el caso 16:
        invocar(codigoDeAtaque,posicionAtaque, level,60);
            romper;
        si se da el caso 18:
            int pos1 = posicionAtaque-1, pos2 = posicionAtaque +1;
            si (pos1 >=0)
            {
                String name = Object.getNombre(equipoAtaque[pos1][0]);
                equipoAtaque[pos1][2]  += equipoAtaque[pos1][2] * level;
                Imprimir(" MAPACHE HA DADO "+level+"x  A ALIADO DE
ADELANTE");
            }
            si (pos2<=4)
            {
                String name = Object.getNombre(equipoAtaque[pos1][0]);
                equipoAtaque[pos2][2]  += equipoAtaque[pos2][2] * level;
                Imprimir(" MAPACHE HA DADO "+level+"x  A ALIADO DE ATRAS");
            }
            eliminarMascota(posicionAtaque);
            romper;
        si se da el caso 20:
            int f=posicionAtaque+1;
```

```
            int cnt=1;
            mientras(f!=4 && cnt <= level)
            {
               equipoAtaque[f][1]+=20;
               f++;
               cnt++;
            }
            Imprimir("TORTUGA HA DADO ARMADURA DE MELON A "+ (cnt-1)+ "
ALIADOS DETRAS");
            eliminarMascota(posicionAtaque);
            romper;
         si se da el caso 22:
         invocar(codigoDeAtaque,posicionAtaque, level,57); romper;
         si se da el caso 28:
         invocar(codigoDeAtaque,posicionAtaque, level,58);  romper;
         si se da el caso 48:
            level *=2;
            para (f=0;f<5;f++)
            {
               si (equipoAtaque[f][0] !=0 && equipoAtaque[f][0] != codigoDeAtaque)
               {
                  equipoAtaque[f][1] += level;
                  equipoAtaque[f][2] += level;
               }
            }
            Imprimir("MAMUT HA DADO A TODOS SUS AMIGOS " + level+
"/"+level);
            eliminarMascota(posicionAtaque);
            romper;
         defecto: eliminarMascota(posicionAtaque);
      }
   }

   publico void eliminarMascota(int posatt)
   {
      equipoAtaque[posatt][0] = 0;
      equipoAtaque[posatt][1] = 0;
      equipoAtaque[posatt][2] = 0;
      equipoAtaque[posatt][3] = 0;
      equipoAtaque[posatt][4] = 0;
      equipoAtaque[posatt][5] = 0;
      equipoAtaque[posatt][6] = 0;
      equipoAtaque[posatt][7] = 0;
      equipoAtaque[posatt][8] = 0;
      equipoAtaque[posatt][9] = 0;
      ordenEquipo(true);
   }
```

```
publico void ordenEquipo(boolean positivo)
{
   si (positivo)
   {
      para (int f =0; f<5;f++)
      {
      si (equipoAtaque[f][0]==0 && f<4)
      {

         equipoAtaque[f][0] = equipoAtaque[f+1][0];
         equipoAtaque[f][1] = equipoAtaque[f+1][1];
         equipoAtaque[f][2] = equipoAtaque[f+1][2];
         equipoAtaque[f][3] = equipoAtaque[f+1][3];
         equipoAtaque[f][4] = equipoAtaque[f+1][4];
         equipoAtaque[f][5] = equipoAtaque[f+1][5];
         equipoAtaque[f][6] = equipoAtaque[f+1][6];
         equipoAtaque[f][7] = equipoAtaque[f+1][7];
         equipoAtaque[f][8] = equipoAtaque[f+1][8];
         equipoAtaque[f][9] = equipoAtaque[f+1][9];
         equipoAtaque[f+1][0] = 0;
         equipoAtaque[f+1][1] = 0;
         equipoAtaque[f+1][2] = 0;
         equipoAtaque[f+1][3] = 0;
         equipoAtaque[f+1][4] = 0;
         equipoAtaque[f+1][5] = 0;
         equipoAtaque[f+1][6] = 0;
         equipoAtaque[f+1][7] = 0;
         equipoAtaque[f+1][8] = 0;
         equipoAtaque[f+1][9] = 0;
      }
      }
   }
   demas{
      para (int f =4; f>=0;f--)
      {
         si (equipoAtaque[f][0]==0 && f>0)
         {
            equipoAtaque[f][0] = equipoAtaque[f-1][0];
            equipoAtaque[f][1] = equipoAtaque[f-1][1];
            equipoAtaque[f][2] = equipoAtaque[f-1][2];
            equipoAtaque[f][3] = equipoAtaque[f-1][3];
            equipoAtaque[f][4] = equipoAtaque[f-1][4];
            equipoAtaque[f][5] = equipoAtaque[f-1][5];
            equipoAtaque[f][6] = equipoAtaque[f-1][6];
            equipoAtaque[f][7] = equipoAtaque[f-1][7];
            equipoAtaque[f][8] = equipoAtaque[f-1][8];
```

```
                equipoAtaque[f][9] = equipoAtaque[f-1][9];
                equipoAtaque[f-1][0] = 0;
                equipoAtaque[f-1][1] = 0;
                equipoAtaque[f-1][2] = 0;
                equipoAtaque[f-1][3] = 0;
                equipoAtaque[f-1][4] = 0;
                equipoAtaque[f-1][5] = 0;
                equipoAtaque[f-1][6] = 0;
                equipoAtaque[f-1][7] = 0;
                equipoAtaque[f-1][8] = 0;
                equipoAtaque[f-1][9] = 0;
            }
        }
    }
}

publico void invocar(int codeatt,int posatt, int level,int codeinv)
{
    String nameatt = Object.getNombre(codeatt), nameinv =
Object.getNombre(codeinv);
    eliminarMascota(posatt);
    ordenEquipo(false);
    si (codeatt==4)
    {
        equipoAtaque[0][0] = codeinv;
        equipoAtaque[0][1] = level;
        equipoAtaque[0][2] = level;
        equipoAtaque[0][3] = 1;
        Imprimir( nameatt +" HA INVOCADO A "+ nameinv);
    }
    si (codeatt == 14)
    {
        int posi=0;
        para(int cant =1;cant <=level;cant++)
        {
            si (equipoAtaque[posi][0]==0)
            {
                equipoAtaque[posi][0] = codeinv;
                equipoAtaque[posi][1] =1;
                equipoAtaque[posi][2] =1;
                equipoAtaque[posi][3] =1;
                posi++;
            }
        }
        Imprimir( nameatt +" HA INVOCADO A "+ nameinv);
    }
    si (codeatt == 16)
```

```
    {
        equipoAtaque[0][0]= codeinv;
        equipoAtaque[0][1] =1;
        equipoAtaque[0][2] =1;
        equipoAtaque[0][3] = level;
        si (level ==3)
        {
            equipoAtaque[0][1] +=1;
            equipoAtaque[0][2] +=1;
        }
        Imprimir( nameatt +" HA INVOCADO A "+ nameinv);
    }
    si (codeatt == 22)
    {
        int posi=0;
        para(int cant =1;cant <=2;cant++)
        {
            si (equipoAtaque[posi][0]==0)
            {
                level *=2;
                equipoAtaque[posi][0] = codeinv;
                equipoAtaque[posi][1] =level;
                equipoAtaque[posi][2] =level;
                equipoAtaque[posi][3] =1;
                posi++;
            }
        }
        Imprimir( nameatt +" HA INVOCADO A "+ nameinv);
    }
    si (codeatt==28)
    {
        level *=5;
        equipoAtaque[0][0] = codeinv;
        equipoAtaque[0][1] = level;
        equipoAtaque[0][2] = level;
        equipoAtaque[0][3] = 1;
        Imprimir( nameatt +" HA INVOCADO A "+ nameinv);
    }
    ordenEquipo(true);
}

publico void habilidadInicial(int turno)
{
    cont=0;
    granturn(turno);
    para(int f =0 ; f <5; f++)
    inicaHabilidad(equipoAtaque [f][0], f,turno);
```

```
        restoreturn(turno);
    }

    publico void inicaHabilidad(int codigoAtaque, int posicionAtaque, int turno)
    {
        int nivel = equipoAtaque[posicionAtaque][3];
        int vDefensa =0, vAtaque =0, ataque=0, ataqueA=0, ataqueD=0,
segundaPosicion=0,vida =0, max=1, min=100;
        String nameattack = getNombre(codigoAtaque);
        switch(codigoAtaque)
        {
            si se da el caso 3,9,10,11,17,19,31,38,39,41,49,53,54:
            si (cont ==0)
            Imprimir("EQUIPO "+turno+": ");
            cont++;
            romper;
        }
        switch (codigoAtaque)
        {
            si se da el caso 3:
            Imprimir("HABILIDAD INICIAL "+ nameattack) ;
            para (int c = 1;c <= nivel;c++)
                {
                    int posdef = generarNumAleatorio(4, 0);
                    si(equipoDefensa[posdef][0]==0 && posdef >0)
                    {
                        posdef--;
                    }
                    si(equipoDefensa[posdef][0]==0 && posdef >0)
                    {
                        posdef--;
                    }
                    si(equipoDefensa[posdef][0]==0 && posdef >0)
                    {
                        posdef--;
                    }
                    si(equipoDefensa[posdef][0]==0 && posdef >0)
                    {
                        posdef--;
                    }

                    ataque(posicionAtaque, posdef, 1,turno);
                }
            romper;
            si se da el caso 9: //metamorfosis(sapo): copiar vida del aliado con mas
vida(no es permanente)
            max=1;
```

```
        para(int f=0;f<5;f++){
           si(equipoAtaque[f][1]>max){
              max = equipoAtaque[f][1];
              equipoAtaque [posicionAtaque][1]= max;
           }
        }
        Imprimir("HABILIDAD INICIAL "+ nameattack) ;
        Imprimir("HA COPIADO LA VIDA DEL ALIADO MAS FUERTE");
        romper;
        si se da el caso 10:
        segundaPosicion = posicionAtaque -1;
        si (segundaPosicion>=0)
        {
          ataque = equipoAtaque[segundaPosicion][2];
          ataque =  ataque + ataque * ((50 *nivel)/100);
          equipoAtaque[segundaPosicion][2] = ataque;
          Imprimir("HABILIDAD INICIAL "+ nameattack) ;
          Imprimir("HA DADO ATAQUE AL DE ADELANTE");
        }
        romper;
        si se da el caso 11:
        segundaPosicion = posicionAtaque + 1;
        para (int c = 1;c <= nivel;c++)
           {
              si (segundaPosicion<=4)
              {
                 equipoAtaque[segundaPosicion][1]-=1;
                 segundaPosicion++;
              }
           }
           Imprimir("HABILIDAD INICIAL "+ nameattack) ;
           Imprimir("HA HECHO 1 DE DAÑO A " + nivel + " ALIADO(S) DE
ATRAS");
        romper;
        si se da el caso 17: //Joroba(camello): Dar amigo detrás (+1/+2)/(+2/+4) /
(+3/+6)
        segundaPosicion = posicionAtaque + 1;
        si (segundaPosicion<=4)
        {
           ataque = 1 * nivel;
           vida = 2 * nivel;
           equipoAtaque[segundaPosicion][1]+= vida;
           equipoAtaque[segundaPosicion][2]+= ataque;
           Imprimir("HABILIDAD INICIAL "+ nameattack) ;
           Imprimir("HA COPIADO LA VIDA DEL ALIADO MAS FUERTE");
        }
        romper;
```

```
        si se da el caso 19:// Fortaleza aliada(jirafa): Da 1, 2 o 3 amigos por
delante +1/+1 al finalizar el turno de compra.
        segundaPosicion = posicionAtaque - 1;
        para (int c = 1;c <= nivel;c++)
           {
              si (segundaPosicion>=0)
              {
                 equipoAtaque[segundaPosicion][1]+= 1;
                 equipoAtaque[segundaPosicion][2]+= 1;
                 segundaPosicion--;

              }
           }
           Imprimir("HABILIDAD INICIAL "+ nameattack) ;
           Imprimir("HA DADO 1/1 A " + nivel + " ALIADO(S) DE DELANTE");
        romper;
        si se da el caso 31: // Salpicón(delfin): reparte 5/10/15 de daño al enemigo
con la salud más baja al comenzar la batalla.
        para(int f=0;f<5;f++){
           si(equipoDefensa[f][1] < min){
              min = equipoDefensa[f][1];
              segundaPosicion = f;
           }
        }
        equipoDefensa[segundaPosicion][1] -= (5*nivel);
        romper;
        si se da el caso 38: //Aguja(escorpion): tiene un attack de veneno innato (el
veneno ejecuta a la mascota enemiga sin importar cuánta vida tenga)
        equipoDefensa[0][1] -= 1000;
        Imprimir("HABILIDAD INICIAL "+ nameattack) ;
        Imprimir("HA EJECUTADO A UN ENEMIGO AL AZAR");
        romper;
        si se da el caso 39: // Estampida(rinoceronte): Inflige 4/8/12 de daño al
primer enemigo.
        equipoDefensa[0][1] -= 4*nivel;
        Imprimir("HABILIDAD INICIAL "+ nameattack) ;
        Imprimir("HA HECHO "+ nivel*4 + " DE DAÑO AL PRIMER ENEMIGO");
        romper;
        si se da el caso 41:// Mordida(cocodrilo): (Comienzo de la batalla) inflige
8/16/24 de daño al último enemigo.  romper;
        para(int f =4; f>=0; f--){
           si(equipoDefensa[f][0]!=0){
              equipoDefensa[f][1] -= 8*nivel;
           }
        }
        Imprimir("HABILIDAD INICIAL "+ nameattack) ;
        Imprimir("HA HECHO "+ nivel*8 + " DE DAÑO AL ULTIMO ENEMIGO");
```

```
        si se da el caso 49: //Zarpazo(leopardo): al iniciar la batalla inflige 50 % de daño ATQ a 1/2/3 enemigos aleatorios .
        para (int c = 1;c <= nivel;c++)
            {
                int posdef = generarNumAleatorio(4, 0);
                ataque = (equipoAtaque[posicionAtaque][2]/2);
                si(equipoDefensa[posdef][0]==0 && posdef >0)
                {
                    posdef--;
                }
                si(equipoDefensa[posdef][0]==0 && posdef >0)
                {
                    posdef--;
                }
                si(equipoDefensa[posdef][0]==0 && posdef >0)
                {
                    posdef--;
                }
                si(equipoDefensa[posdef][0]==0 && posdef >0)
                {
                    posdef--;
                }
                Imprimir("HABILIDAD INICIAL "+ nameattack);
                ataque(posicionAtaque, posdef, ataque, turno);
            }
        romper;
        si se da el caso 53://  Habilidades por nivel(quetzal): (1) Agrega a su vida la suma de toda la vida de los animales tipos aves. (2) Hace lo del nivel 1 y agrega a su daño la suma de todo el daño del daño de todas las aves. (3) Hace lo del nivel 2 pero con todos los animales.
        Imprimir("HABILIDAD INICIAL "+ nameattack);
        si(nivel==1){
            para(int f=0;f<5;f++){
                si(equipoAtaque[f][0]!=53){
                    si(equipoAtaque[f][4]==2 | equipoAtaque[f][5]==2 || equipoAtaque[f][6]==2){
                        vAtaque += equipoAtaque[f][1];
                    }
                }
                si(equipoDefensa[f][4]==2 | equipoDefensa[f][5]==2 || equipoDefensa[f][6]==2){
                    vAtaque += equipoDefensa[f][1];
                }
            }
            equipoAtaque[posicionAtaque][1] += vAtaque+vDefensa;
            Imprimir("Agrega a su vida la suma de toda la vida de los animales tipos aves.");
```

```
        }
        si(nivel==2){
            para(int f=0;f<5;f++){
                si(equipoAtaque[f][0]!=53){
                    si(equipoAtaque[f][4]==2 | equipoAtaque[f][5]==2 ||
equipoAtaque[f][6]==2){
                        vAtaque += equipoAtaque[f][1];
                        ataqueA += equipoAtaque[f][2];
                    }
                }
                si(equipoDefensa[f][4]==2 | equipoDefensa[f][5]==2 ||
equipoDefensa[f][6]==2){
                    vAtaque += equipoDefensa[f][1];
                    ataqueD += equipoDefensa[f][2];
                }
            }
            equipoAtaque[posicionAtaque][1] += vAtaque+vDefensa;
            equipoAtaque[posicionAtaque][2] += ataqueA+ataqueD;
            Imprimir("Agrega a su vida la suma de toda la vida de los animales tipos
aves y la suma de todo el daño del daño de todas las aves.");
        }
        si(nivel==3){
            para(int f=0;f<5;f++){
                si(equipoAtaque[f][0]!=53){
                    vAtaque += equipoAtaque[f][1];
                    ataqueD += equipoAtaque[f][2];
                }
                    vAtaque += equipoDefensa[f][1];
                    ataqueD += equipoDefensa[f][2];
            }
            equipoAtaque[posicionAtaque][1] += vAtaque+vDefensa;
            equipoAtaque[posicionAtaque][2] += ataqueA+ataqueD;
            Imprimir("Agrega a su vida la suma de toda la vida y la suma de todo el
daño de todos los animales");
        }

        si se da el caso 54: // Habilidades por nivel(camaleon): (1) Copia la vida del
enemigo más fuerte (2) Copia la vida y el daño del enemigo más fuerte  (3) Copia
la vida, el daño y la habilidad del enemigo más fuerte romper;
        romper;

    }

}

publico void ataque(int posatt, int posdef, int ataque, int turno)
{
```

```
        boolean repeat = false;
        hacer{
            int codeatt = equipoAtaque[posatt][0], codedef = equipoDefensa[posdef][0];
            String nameatt= Object.getNombre(codeatt), namedef =
Object.getNombre(codedef);
            int nivel =equipoDefensa[posdef][3];//exclusivo para panda

            si (codeatt == 44)
            {
                si(nivel ==1){
                ataque -= (ataque /2);
                Imprimir("PANDA REDUJO EL ATAQUE UN 50/100");
                }
                demas si ( nivel == 2){
                ataque -= ( (3*ataque) /5);
                Imprimir("PANDA REDUJO EL ATAQUE UN 60/100");
                }
                demas{
                ataque -=  ( (4*ataque) /5);
                Imprimir("PANDA REDUJO EL ATAQUE UN 80/100");
                }
            }


            si (codeatt == 50)
            {
                //ESCUDO DE COCO
            }
            Imprimir(nameatt +"  HIZO  " +ataque+" DE DAÑO A "+ namedef);
            equipoDefensa [posdef][1] -= ataque;

            si (codeatt==30 && equipoDefensa [posdef][1] <=0)
            {
                nivel = equipoAtaque[posatt][3];
                nivel = nivel *2 ;
                equipoAtaque[posatt][1] += nivel;
                equipoAtaque[posatt][2] += nivel;
                Imprimir("HIPOPOTAMO HA DERRIBADO A "+namedef + " Y HA
OBTENIDO "+ nivel +"/"+nivel);
            }
            si (codedef == 13)
            {
                equipoDefensa[posdef][2] += ataque /2;
                Imprimir("PAVOREAL HA GANADO 50/100 DE ATAQUE RECIBIDO");
            }

            si (posdef!=4)
```

```
        codedef = equipoDefensa[posdef+1][0];
        si (codedef == 37)
        {
            repeat = true;
            int mattemp[][] = equipoDefensa;
            equipoDefensa = equipoAtaque;
            equipoAtaque = mattemp;
            codedef = codeatt;
            codeatt = 37;
        }

    }mientras(repeat);
}

publico void battack(int posatt)
{

}

publico void granturn (int turn)
{
    si(turn==1)
    {
        para(int f=0;f<5;f++)
        {
            para(int c=0;c<10;c++)
            {
                equipoAtaque[f][c]= t1r[f][c];
                equipoDefensa[f][c]= t2r[f][c];
            }
        }
    }
    demas
    {
        para(int f=0;f<5;f++)
        {
            para(int c=0;c<10;c++)
            {
                equipoDefensa[f][c]= t1r[f][c];
                equipoAtaque[f][c]= t2r[f][c];
            }
        }
    }
}

publico void restoreturn (int turn)
{
```

```
        si(turn==1)
        {
            para(int f=0;f<5;f++)
            {
                para(int c=0;c<10;c++)
                {
                    t1r[f][c] = equipoAtaque[f][c];
                    t2r[f][c] = equipoDefensa[f][c];
                }
            }
        }
        demas
        {
            para(int f=0;f<5;f++)
            {
                para(int c=0;c<10;c++)
                {
                    t1r[f][c] = equipoDefensa[f][c];
                    t2r[f][c] = equipoAtaque[f][c];
                }
            }
        }
    }

    publico int generarNumAleatorio(int max, int min)
    {
        devolver random.nextInt(max-(min-1))+1+(min-1);
    }
}
```

PRINCIPAL - MENU

```
publico clase menu {
    publico static void main(String[] args) throws Exception {

        escogerarena tipoar = new escogerarena();
        int opcion;
        String nombre;
        Scanner scan = new Scanner(System.in);
        boolean hayerror = false;
        Imprimir("\nIngrese su nombre: ");
        nombre = scan.nextLine();

        hacer{
        Imprimir("\n------------------------------------------------------------");
        Imprimir("|  <3  S U P E R  A U T O  P E T S  <3 |");
        Imprimir("------------------------------------------------------------");
        Imprimir("\n------------------ Escoja una opcion : ------------------");
        Imprimir("1) ..... M O D O  A R E N A ");
        Imprimir("2) ..... M O D O  V E R S U S ");
        Imprimir("3) ..... M O D O  C R E A T I V O ");
        Imprimir("4) ..... R E P O R T E S ");
        Imprimir("5) ..... S A L I R  D E L  J U E G O ");
        Imprimir("\n--------------- Ingrese una opcion --------------");
        opcion = scan.nextInt();
        scan.nextLine();

        switch(opcion){
            si se da el caso 1:
                tipoar.elegirArena(nombre, opcion);
                //mandar tipo batalla a ARENA
                hayerror = true;
            romper;
            si se da el caso 2:
                tipoar.elegirArena(nombre, opcion);
                hayerror = true;
            romper;
            si se da el caso 3:
                tipoar.elegirArena(nombre, opcion);
                hayerror = true;
            romper;
            si se da el caso 4:
                Imprimir("---------REPORTES---------");
                hayerror = true;
            romper;
            si se da el caso 5:
                Imprimir("Saliendo del juego. Esperamos su pronto regreso a esta
aventura");
```

```
            hayerror = false;
        romper;
        defecto:
            Imprimir("Esta opcion no existe. Por favor ingrese una opcion valida");
            hayerror = true;
        romper;
      }
   }mientras(hayerror);
}
}
```

RECURSOS

ANIMALES

```
publico clase animales {
        publico int ataque, defensa, vida, codigo;
        publico String  tipo1,tipo2, tipo3, nombre;

        publico animales(int ataque, int defensa, int vida, int codigo,String tipo1,String
tipo2, String tipo3, String name){
            this.codigo = codigo;
            this.ataque = ataque;
            this.defensa = defensa;
            this.vida = vida;
            this.tipo1 = tipo1;
            this.tipo2 = tipo2;
            this.tipo3 = tipo3;
            this.nombre = name;
        }
        publico int getVida() {
            devolver vida;
        }
        publico int getAtaque() {
            devolver ataque;
        }
        publico int getDefensa(){
            devolver defensa;
        }
        publico String getTipo1(){
            devolver tipo1;
        }
        publico String getTipo2(){
            devolver tipo2;
        }
        publico String getTipo3(){
            devolver tipo3;
        }
        publico String getNombre()
        {
            devolver nombre;
        }
        publico void setAtaque(int ataque){
            this.ataque = ataque;
        }
        publico void setVida(int vida){
            this.vida = vida;
        }
        publico void setTipo1(String tipo1){
            this.tipo1 = tipo1;
```

```
    }
    publico void setTipo2(String tipo2){
       this.tipo2 = tipo2;
    }
    publico void setTipo3(String tipo3){
       this.tipo3 = tipo3;
    }
    publico void setDefensa(int defensa){
       this.defensa = defensa;
    }

    publico int [] getNdates()
    {
       int Ndates [] = {getDefensa(), getAtaque(), getVida()};
       devolver Ndates;
    }
    publico String [] getSdates()
    {
       devolver new String[] {getNombre(),getTipo1(),getTipo2(),getTipo3()};
    }
}
```

COMIDA

```
publico clase comida {
    int  codigo,tipo;
    String  nombre;

    publico comida(int codigo,String nombre, int tipo){
        this.codigo = codigo;
        this.nombre = nombre;
        this.tipo = tipo;
    }

    publico String getNombre(){
        devolver nombre;
    }
    publico void setNombre(String nombre){
        this.nombre = nombre;
    }
    publico int getTipo(){
        devolver tipo;
    }
}
```

OBJETOS

```
publico clase objetos{
    protegido animales Hormiga = new animales(2, 2, 2, 1, "Insecto",
"Terrestre","","Hormiga");
    protegido animales Pescado = new animales(2, 3, 3, 2, "Acuatico",
"","","Pescado");
    protegido animales Mosquito = new animales(2, 1, 2, 3, "Volador",
"","","Mosquito");
    protegido animales Grillo = new animales(1, 3, 2, 4, "Insecto", "","","Grillo");
    protegido animales Castor = new animales(2, 3, 2, 5, "Terrestre",
"Acuatico","","Castor");
    protegido animales Caballo = new animales(2, 4, 1, 6, "Mamifero",
"Domestico","","Caballo");
    protegido animales Nutria = new animales(1, 3, 2, 7, "Mamifero", "","","Nutria");
    protegido animales Escarabajo = new animales(2, 5, 3, 8, "Insecto",
"","","Escarabajo");
    protegido animales Sapo = new animales(3, 4, 3, 9, "Terrestre",
"Acuatico","","Sapo");
    protegido animales Dodo = new animales(2, 5, 3, 10, "Volador", "","","Dodo");
    protegido animales Elefante = new animales(3, 7, 5, 11, "Mamifero",
"Terrestre","","Elefante");
    protegido animales Puercoespin = new animales(3, 3, 2, 12, "Solitario",
"Terrestre","","Puercoespin");
    protegido animales Pavoreal = new animales(2, 4, 5, 13, "Domestico",
"Solitario","","Pavoreal");
    protegido animales Rata = new animales(4, 4, 5, 14, "Terrestre",
"Solitario","","Rata");
    protegido animales Zorro = new animales(5, 3, 2, 15, "Solitario",
"Terrestre","","Zorro");
    protegido animales Arania = new animales(2, 3, 2, 16, "Insecto", "","","Araña");
    protegido animales Camello = new animales(2, 4, 5, 17, "Mamifero",
"Desertico","","Camello");
    protegido animales Mapache = new animales(5, 3, 4, 18, "Solitario",
"","","Mapache");
    protegido animales Jirafa = new animales(2, 5, 5, 19, "Mamifero",
"Terrestre","","Jirafa");
    protegido animales Tortuga = new animales(1, 5, 2, 20, "Reptil", "","","Tortuga");
    protegido animales Caracol = new animales(2, 4, 2, 21, "Insecto",
"Solitario","","Caracol");
    protegido animales Oveja = new animales(2, 3, 2, 22, "Domestico",
"Terrestre","","Oveja");
    protegido animales Conejo = new animales(3, 2, 2, 23, "Mamifero",
"","","Conejo");
    protegido animales Lobo = new animales(3, 3, 4, 24, "Solitario",
"Terrestre","","Lobo");
    protegido animales Buey = new animales(1, 5, 4, 25, "Mamifero", "","","Buey");
```

```
    protegido animales Canguro = new animales(1, 4, 2, 26, "Mamifero",
"Terrestre","","Canguro");
    protegido animales Buho = new animales(5, 2, 3, 27, "Volador",
"Solitario","","Buho");
    protegido animales Venado = new animales(1, 3, 1, 28, "Mamifero",
"","","Venado");
    protegido animales Loro = new animales(5, 2, 3, 29, "Volador", "","","Loro");
    protegido animales Hipopotamo = new animales(4, 6, 7, 30, "Acuatico",
"Terrestre","","Hipopotamo");
    protegido animales Delfin = new animales(4, 4, 6, 31, "Acuatico", "","","Delfin");
    protegido animales Puma = new animales(3, 4, 7, 32, "Mamifero",
"Terrestre","","Puma");
    protegido animales Ballena = new animales(3, 7, 8, 33, "Acuatico",
"","","Ballena");
    protegido animales Ardilla = new animales(2, 3, 5, 34, "Domestico",
"","","Ardilla");
    protegido animales LLama = new animales(3, 4, 6, 35, "Terrestre",
"","","LLama");
    protegido animales Foca = new animales(3, 3, 8, 36, "Acuatico",
"Mamifero","","Foca");
    protegido animales Jaguar = new animales(7, 4, 4, 36, "Mamifero",
"Terrestre","","Jaguar");
    protegido animales Escorpion = new animales(1, 2, 1, 37, "Desertico",
"Solitario","","Escorpion");
    protegido animales Rinoceronte = new animales(5, 5, 8, 38, "Desertico",
"Terrestre","","Rinoceronte");
    protegido animales Mono = new animales(1, 3, 2, 39, "Mamifero", "","","Mono");
    protegido animales Cocodrilo = new animales(8, 4, 4, 40, "Reptil",
"Solitario","","Cocodrilo");
    protegido animales Vaca = new animales(4, 4, 6, 41, "Mamifero",
"Terrestre","","Vaca");
    protegido animales Chompipe = new animales(3, 3, 4, 42, "Terrestre",
"Volador","","Chompipe");
    protegido animales Panda = new animales(5, 4, 5, 43, "Mamifero",
"Solitario","","Panda");
    protegido animales Gato = new animales(4, 3, 5, 44, "Mamifero",
"Domestico","","Gato");
    protegido animales Tigre = new animales(4, 4, 3, 45, "Terrestre",
"Mamifero","","Tigre");
    protegido animales Serpiente = new animales(6, 3, 6, 46, "Reptil",
"Terrestre","Desertico","Serpiente");
    protegido animales Mamut = new animales(3, 5, 10, 47, "Mamifero",
"Terrestre","Solitario","Mamut");
    protegido animales Leopardo = new animales(10, 4, 4, 48, "Mamifero",
"Terrestre","","Leopardo");
    protegido animales Gorila = new animales(6, 4, 9, 49, "Mamifero",
"Terrestre","","Gorila");
```

```
    protegido animales Pulpo = new animales(8, 3, 8, 50, "Acuatico",
"Solitario","","Pulpo");
    protegido animales Mosca = new animales(5, 3, 5, 52, "Volador",
"Insecto","","Mosca");
    protegido animales Quetzal = new animales(10, 7, 10, 53, "Volador",
"Solitario","","Quetzal");
    protegido animales Camaleon = new animales(8, 5, 8, 54, "Reptil",
"Solitario","","Camaleon");
    protegido animales Grillo_Zombie = new animales(1, 4, 1, 55, "Insecto", "",
"","Grillo_Zombie");
    protegido animales Dirty_Rat = new animales(1, 2, 1, 56, "Terrestre", "Solitario",
"","Dirty_Rat");
    protegido animales Carnero = new animales(2, 4, 2, 57, "Domestico",
"Terrestre", "","Carnero");
    protegido animales Autobus = new animales(5, 5, 5, 58, "Terrestre", "",
"","Autobus");
    protegido animales Zombie_Fly = new animales(5, 3, 5, 59, "Volador", "Insecto",
"","Zombie_Fly");
    protegido animales Miniarania = new animales(1,  1,  1,  60, "Insecto","", "",
"Miniarania");
    //atack  def  vid   codi  tipo  tipo2 tipo3
    protegido comida Manzana = new comida(1 , "Manzana" , 0 );
    protegido comida Naranja = new comida(2 , "Naranja" , 1 );
    protegido comida Miel = new comida(3 , "Miel" , 1 );
    protegido comida Pastelito = new comida(4 , "Pastelito" , 1 );
    protegido comida Huesodecarne = new comida(5 , "Huesodecarne" , 1 );
    protegido comida Pastillaparadormir = new comida(6 , "Pastillaparadormir" , 1 );
    protegido comida Ajo = new comida(7 , "Ajo" , 1 );
    protegido comida Ensalada = new comida(8 , "Ensalada" , 0 );
    protegido comida Comidaenlatada = new comida(9 , "Comidaenlatada" , 1 );
    protegido comida Pera = new comida(10 , "Pera" , 0 );
    protegido comida Chile = new comida(11 , "Chile" , 1 );
    protegido comida Chocolate = new comida(12 , "Chocolate" , 0 );
    protegido comida Sushi = new comida(13 , "Sushi" , 0 );
    protegido comida Melon = new comida(14 , "Melon" , 1 );
    protegido comida Hongo = new comida(15 , "Hongo" , 1 );
    protegido comida Pizza = new comida(16 , "Pizza" , 1 );
    protegido comida Carne = new comida(17 , "Carne" , 1 );
    protegido comida Gelatina = new comida(18 , "Gelatina" , 1 );


    publico int [] getNdates(int codigo)
    {
        int dates []={0,0,0};
        switch(codigo)
        {
            si se da el caso 1:  dates = Hormiga.getNdates(); romper;
```

si se da el caso 2: dates = Pescado.getNdates(); romper;
si se da el caso 3: dates = Mosquito.getNdates(); romper;
si se da el caso 4: dates = Grillo.getNdates(); romper;
si se da el caso 5: dates = Castor.getNdates(); romper;
si se da el caso 6: dates = Caballo.getNdates(); romper;
si se da el caso 7: dates = Nutria.getNdates(); romper;
si se da el caso 8: dates = Escarabajo.getNdates(); romper;
si se da el caso 9: dates = Sapo.getNdates(); romper;
si se da el caso 10: dates = Dodo.getNdates(); romper;
si se da el caso 11: dates = Elefante.getNdates(); romper;
si se da el caso 12: dates = Puercoespin.getNdates(); romper;
si se da el caso 13: dates = Pavoreal.getNdates(); romper;
si se da el caso 14: dates = Rata.getNdates(); romper;
si se da el caso 15: dates = Zorro.getNdates(); romper;
si se da el caso 16: dates = Arania.getNdates(); romper;
si se da el caso 17: dates = Camello.getNdates(); romper;
si se da el caso 18: dates = Mapache.getNdates(); romper;
si se da el caso 19: dates = Jirafa.getNdates(); romper;
si se da el caso 20: dates = Tortuga.getNdates(); romper;
si se da el caso 21: dates = Caracol.getNdates(); romper;
si se da el caso 22: dates = Oveja.getNdates(); romper;
si se da el caso 23: dates = Conejo.getNdates(); romper;
si se da el caso 24: dates = Lobo.getNdates(); romper;
si se da el caso 25: dates = Buey.getNdates(); romper;
si se da el caso 26: dates = Canguro.getNdates(); romper;
si se da el caso 27: dates = Buho.getNdates(); romper;
si se da el caso 28: dates = Venado.getNdates(); romper;
si se da el caso 29: dates = Loro.getNdates(); romper;
si se da el caso 30: dates = Hipopotamo.getNdates(); romper;
si se da el caso 31: dates = Delfin.getNdates(); romper;
si se da el caso 32: dates = Puma.getNdates(); romper;
si se da el caso 33: dates = Ballena.getNdates(); romper;
si se da el caso 34: dates = Ardilla.getNdates(); romper;
si se da el caso 35: dates = LLama.getNdates(); romper;
si se da el caso 36: dates = Foca.getNdates(); romper;
si se da el caso 37: dates = Jaguar.getNdates(); romper;
si se da el caso 38: dates = Escorpion.getNdates(); romper;
si se da el caso 39: dates = Rinoceronte.getNdates(); romper;
si se da el caso 40: dates = Mono.getNdates(); romper;
si se da el caso 41: dates = Cocodrilo.getNdates(); romper;
si se da el caso 42: dates = Vaca.getNdates(); romper;
si se da el caso 43: dates = Chompipe.getNdates(); romper;
si se da el caso 44: dates = Panda.getNdates(); romper;
si se da el caso 45: dates = Gato.getNdates(); romper;
si se da el caso 46: dates = Tigre.getNdates(); romper;
si se da el caso 47: dates = Serpiente.getNdates(); romper;
si se da el caso 48: dates = Mamut.getNdates(); romper;

```
        si se da el caso 49:  dates = Leopardo.getNdates(); romper;
        si se da el caso 50:  dates = Gorila.getNdates(); romper;
        si se da el caso 51:  dates = Pulpo.getNdates(); romper;
        si se da el caso 52:  dates = Mosca.getNdates(); romper;
        si se da el caso 53:  dates = Quetzal.getNdates(); romper;
        si se da el caso 54:  dates = Camaleon.getNdates(); romper;
        si se da el caso 55:  dates = Grillo_Zombie.getNdates(); romper;
        si se da el caso 56:  dates = Dirty_Rat.getNdates(); romper;
        si se da el caso 57:  dates = Carnero.getNdates(); romper;
        si se da el caso 58:  dates = Autobus.getNdates(); romper;
        si se da el caso 59:  dates = Zombie_Fly.getNdates(); romper;
    }

  devolver dates;
}
publico String [] getSdates(int codigo)
{
    String dates [] = {"","","","",""};
    switch(codigo){
        si se da el caso 1:  dates = Hormiga.getSdates(); romper;
        si se da el caso 2:  dates = Pescado.getSdates(); romper;
        si se da el caso 3:  dates = Mosquito.getSdates(); romper;
        si se da el caso 4:  dates = Grillo.getSdates(); romper;
        si se da el caso 5:  dates = Castor.getSdates(); romper;
        si se da el caso 6:  dates = Caballo.getSdates(); romper;
        si se da el caso 7:  dates = Nutria.getSdates(); romper;
        si se da el caso 8:  dates = Escarabajo.getSdates(); romper;
        si se da el caso 9:  dates = Sapo.getSdates(); romper;
        si se da el caso 10:  dates = Dodo.getSdates(); romper;
        si se da el caso 11:  dates = Elefante.getSdates(); romper;
        si se da el caso 12:  dates = Puercoespin.getSdates(); romper;
        si se da el caso 13:  dates = Pavoreal.getSdates(); romper;
        si se da el caso 14:  dates = Rata.getSdates(); romper;
        si se da el caso 15:  dates = Zorro.getSdates(); romper;
        si se da el caso 16:  dates = Arania.getSdates(); romper;
        si se da el caso 17:  dates = Camello.getSdates(); romper;
        si se da el caso 18:  dates = Mapache.getSdates(); romper;
        si se da el caso 19:  dates = Jirafa.getSdates(); romper;
        si se da el caso 20:  dates = Tortuga.getSdates(); romper;
        si se da el caso 21:  dates = Caracol.getSdates(); romper;
        si se da el caso 22:  dates = Oveja.getSdates(); romper;
        si se da el caso 23:  dates = Conejo.getSdates(); romper;
        si se da el caso 24:  dates = Lobo.getSdates(); romper;
        si se da el caso 25:  dates = Buey.getSdates(); romper;
        si se da el caso 26:  dates = Canguro.getSdates(); romper;
        si se da el caso 27:  dates = Buho.getSdates(); romper;
        si se da el caso 28:  dates = Venado.getSdates(); romper;
```

```
        si se da el caso 29:  dates = Loro.getSdates(); romper;
        si se da el caso 30:  dates = Hipopotamo.getSdates(); romper;
        si se da el caso 31:  dates = Delfin.getSdates(); romper;
        si se da el caso 32:  dates = Puma.getSdates(); romper;
        si se da el caso 33:  dates = Ballena.getSdates(); romper;
        si se da el caso 34:  dates = Ardilla.getSdates(); romper;
        si se da el caso 35:  dates = LLama.getSdates(); romper;
        si se da el caso 36:  dates = Foca.getSdates(); romper;
        si se da el caso 37:  dates = Jaguar.getSdates(); romper;
        si se da el caso 38:  dates = Escorpion.getSdates(); romper;
        si se da el caso 39:  dates = Rinoceronte.getSdates(); romper;
        si se da el caso 40:  dates = Mono.getSdates(); romper;
        si se da el caso 41:  dates = Cocodrilo.getSdates(); romper;
        si se da el caso 42:  dates = Vaca.getSdates(); romper;
        si se da el caso 43:  dates = Chompipe.getSdates(); romper;
        si se da el caso 44:  dates = Panda.getSdates(); romper;
        si se da el caso 45:  dates = Gato.getSdates(); romper;
        si se da el caso 46:  dates = Tigre.getSdates(); romper;
        si se da el caso 47:  dates = Serpiente.getSdates(); romper;
        si se da el caso 48:  dates = Mamut.getSdates(); romper;
        si se da el caso 49:  dates = Leopardo.getSdates(); romper;
        si se da el caso 50:  dates = Gorila.getSdates(); romper;
        si se da el caso 51:  dates = Pulpo.getSdates(); romper;
        si se da el caso 52:  dates = Mosca.getSdates(); romper;
        si se da el caso 53:  dates = Quetzal.getSdates(); romper;
        si se da el caso 54:  dates = Camaleon.getSdates(); romper;
        si se da el caso 55:  dates = Grillo_Zombie.getSdates(); romper;
        si se da el caso 56:  dates = Dirty_Rat.getSdates(); romper;
        si se da el caso 57:  dates = Carnero.getSdates(); romper;
        si se da el caso 58:  dates = Autobus.getSdates(); romper;
        si se da el caso 59:  dates = Zombie_Fly.getSdates(); romper;
    }
    devolver dates;
}
publico String getNombre(int codigo)
{
    String dates="";
    switch(codigo){
        si se da el caso 1:  dates = Hormiga.getNombre(); romper;
        si se da el caso 2:  dates = Pescado.getNombre(); romper;
        si se da el caso 3:  dates = Mosquito.getNombre(); romper;
        si se da el caso 4:  dates = Grillo.getNombre(); romper;
        si se da el caso 5:  dates = Castor.getNombre(); romper;
        si se da el caso 6:  dates = Caballo.getNombre(); romper;
        si se da el caso 7:  dates = Nutria.getNombre(); romper;
        si se da el caso 8:  dates = Escarabajo.getNombre(); romper;
        si se da el caso 9:  dates = Sapo.getNombre(); romper;
```

si se da el caso 10: dates = Dodo.getNombre(); romper;
si se da el caso 11: dates = Elefante.getNombre(); romper;
si se da el caso 12: dates = Puercoespin.getNombre(); romper;
si se da el caso 13: dates = Pavoreal.getNombre(); romper;
si se da el caso 14: dates = Rata.getNombre(); romper;
si se da el caso 15: dates = Zorro.getNombre(); romper;
si se da el caso 16: dates = Arania.getNombre(); romper;
si se da el caso 17: dates = Camello.getNombre(); romper;
si se da el caso 18: dates = Mapache.getNombre(); romper;
si se da el caso 19: dates = Jirafa.getNombre(); romper;
si se da el caso 20: dates = Tortuga.getNombre(); romper;
si se da el caso 21: dates = Caracol.getNombre(); romper;
si se da el caso 22: dates = Oveja.getNombre(); romper;
si se da el caso 23: dates = Conejo.getNombre(); romper;
si se da el caso 24: dates = Lobo.getNombre(); romper;
si se da el caso 25: dates = Buey.getNombre(); romper;
si se da el caso 26: dates = Canguro.getNombre(); romper;
si se da el caso 27: dates = Buho.getNombre(); romper;
si se da el caso 28: dates = Venado.getNombre(); romper;
si se da el caso 29: dates = Loro.getNombre(); romper;
si se da el caso 30: dates = Hipopotamo.getNombre(); romper;
si se da el caso 31: dates = Delfin.getNombre(); romper;
si se da el caso 32: dates = Puma.getNombre(); romper;
si se da el caso 33: dates = Ballena.getNombre(); romper;
si se da el caso 34: dates = Ardilla.getNombre(); romper;
si se da el caso 35: dates = LLama.getNombre(); romper;
si se da el caso 36: dates = Foca.getNombre(); romper;
si se da el caso 37: dates = Jaguar.getNombre(); romper;
si se da el caso 38: dates = Escorpion.getNombre(); romper;
si se da el caso 39: dates = Rinoceronte.getNombre(); romper;
si se da el caso 40: dates = Mono.getNombre(); romper;
si se da el caso 41: dates = Cocodrilo.getNombre(); romper;
si se da el caso 42: dates = Vaca.getNombre(); romper;
si se da el caso 43: dates = Chompipe.getNombre(); romper;
si se da el caso 44: dates = Panda.getNombre(); romper;
si se da el caso 45: dates = Gato.getNombre(); romper;
si se da el caso 46: dates = Tigre.getNombre(); romper;
si se da el caso 47: dates = Serpiente.getNombre(); romper;
si se da el caso 48: dates = Mamut.getNombre(); romper;
si se da el caso 49: dates = Leopardo.getNombre(); romper;
si se da el caso 50: dates = Gorila.getNombre(); romper;
si se da el caso 51: dates = Pulpo.getNombre(); romper;
si se da el caso 52: dates = Mosca.getNombre(); romper;
si se da el caso 53: dates = Quetzal.getNombre(); romper;
si se da el caso 54: dates = Camaleon.getNombre(); romper;
si se da el caso 55: dates = Grillo_Zombie.getNombre(); romper;
si se da el caso 56: dates = Dirty_Rat.getNombre(); romper;

```
        si se da el caso 57:  dates = Carnero.getNombre(); romper;
        si se da el caso 58:  dates = Autobus.getNombre(); romper;
        si se da el caso 59:  dates = Zombie_Fly.getNombre(); romper;
    }
    devolver dates;
}
publico int getTipo(int codigo)
{
    int tipo =0;
    switch(codigo)
    {
        si se da el caso 1:  tipo = Manzana.getTipo(); romper;
        si se da el caso 2:  tipo = Naranja.getTipo(); romper;
        si se da el caso 3:  tipo = Miel.getTipo(); romper;
        si se da el caso 4:  tipo = Pastelito.getTipo(); romper;
        si se da el caso 5:  tipo = Huesodecarne.getTipo(); romper;
        si se da el caso 6:  tipo = Pastillaparadormir.getTipo(); romper;
        si se da el caso 7:  tipo = Ajo.getTipo(); romper;
        si se da el caso 8:  tipo = Ensalada.getTipo(); romper;
        si se da el caso 9:  tipo = Comidaenlatada.getTipo(); romper;
        si se da el caso 10:  tipo = Pera.getTipo(); romper;
        si se da el caso 11:  tipo = Chile.getTipo(); romper;
        si se da el caso 12:  tipo = Chocolate.getTipo(); romper;
        si se da el caso 13:  tipo = Sushi.getTipo(); romper;
        si se da el caso 14:  tipo = Melon.getTipo(); romper;
        si se da el caso 15:  tipo = Hongo.getTipo(); romper;
        si se da el caso 16:  tipo = Pizza.getTipo(); romper;
        si se da el caso 17:  tipo = Carne.getTipo(); romper;
        si se da el caso 18:  tipo = Gelatina.getTipo(); romper;
    }
    devolver tipo;
}
publico String getName(int codigo)
{
    String nombre = "";
    switch(codigo)
    {
    si se da el caso 1:  nombre = Manzana.getNombre(); romper;
    si se da el caso 2:  nombre = Naranja.getNombre(); romper;
    si se da el caso 3:  nombre = Miel.getNombre(); romper;
    si se da el caso 4:  nombre = Pastelito.getNombre(); romper;
    si se da el caso 5:  nombre = Huesodecarne.getNombre(); romper;
    si se da el caso 6:  nombre = Pastillaparadormir.getNombre(); romper;
    si se da el caso 7:  nombre = Ajo.getNombre(); romper;
    si se da el caso 8:  nombre = Ensalada.getNombre(); romper;
    si se da el caso 9:  nombre = Comidaenlatada.getNombre(); romper;
    si se da el caso 10:  nombre = Pera.getNombre(); romper;
```

```
    si se da el caso 11:  nombre = Chile.getNombre(); romper;
    si se da el caso 12:  nombre = Chocolate.getNombre(); romper;
    si se da el caso 13:  nombre = Sushi.getNombre(); romper;
    si se da el caso 14:  nombre = Melon.getNombre(); romper;
    si se da el caso 15:  nombre = Hongo.getNombre(); romper;
    si se da el caso 16:  nombre = Pizza.getNombre(); romper;
    si se da el caso 17:  nombre = Carne.getNombre(); romper;
    si se da el caso 18:  nombre = Gelatina.getNombre(); romper;
    }
   devolver nombre;
}
```