EJERCICIOS TEMA 1 PROGRAMACIÓN IMPERATIVA

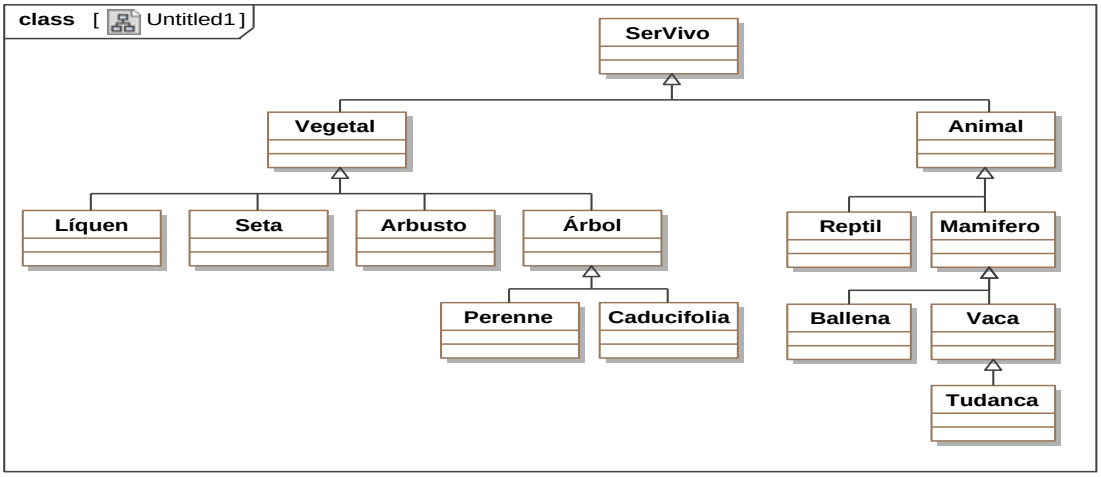

Figura 1. Jerarquía de clases sobre seres vivos

1. Dada la jerarquía de clases de la Figura 1, indicar cuáles son los tipos que poseen las siguientes variables.

   a. pepe : SerVivo;
   b. boletus : Seta;
   c. pino : Perenne;
   d. mobyDick: Ballena;
   e. paca : Tudanca;
   f. x : Arbol.
   g. Lagartija : Reptil;

2. Dado el siguiente trozo de código, indique cuál es el tipo real y cuál es el tipo declarada de las variables *x*, *an*, *y*, *m*, *v*, *veg* y *vac*.

```
public void foo(Animal veg, Vaca vac) {
   SerVivo  x  = veg;
   Animal   an = new Tudanca();
   SerVivo  y  = an;
   Mamifero m  = vac;
   Vaca     v  = vac;
} // foo

public void test() {
   Reptil juancho = new Reptil();
   Vaca   pepa    = new Tudanca();
   foo(juancho,pepa);
} // test
```

3. Para el código que se muestra a continuación, indique qué llamadas a métodos son correctas y cuales son incorrectas y por qué.

```
public void foo(Animal veg, Vaca vac) {...}

public void test2() {

      SerVivo  ameba   = new SerVivo();
      Vegetal  helecho = new Vegetal();
      Liquen   musgo   = new Liquen();
      Seta     boletus = new Seta();
      Arbusto  acebo   = new Arbusto();
      Arbol    encina  = new Arbol();
      Perenne  pino    = new Perenne();
      Caducifolia roble = new Caducifolia();
      Animal   lemur   = new Animal();
      Reptil   juancho = new Reptil();
      Mamifero osoYogi = new Mamifero();
      Ballena  moby    = new Ballena();
      Vaca     pepa    = new Tudanca();
      Tudanca  nena    = new Tudanca();

      foo(ameba,helecho);
      foo(lemur,helecho);
      foo(helecho,lemur);
      foo(moby,helecho);
      foo(pepa,pino);
      foo(helecho,pino);
      foo(pepa,nena);
      foo(osoYogi,moby);
      foo(lemur,pepa);
      foo(juancho,pepa);
      foo(osoYogi,pepa);
      foo(osoYogi,nena);
      foo(nena,nena);

} // test2
```

4. Indique cuáles de los siguientes métodos es correcto y cuál es incorrecto y por qué.

```
public void test3() {
      Mamifero [] mamiferos = new Mamifero[20];
      mamiferos[0] = new Tudanca();
      mamiferos[1] = new Ballena();
      mamiferos[2] = new Vaca();
      mamiferos[3] = new Ballena();
} // test3

public void test4() {
      Mamifero [] bichos = new Mamifero[20];
      bichos[0] = new Tudanca();
      bichos[1] = new Ballena();
      bichos[2] = new Reptil();
      bichos[3] = new Ballena();
} // test4

public void test5() {
      SerVivo [] cosas = new SerVivo[20];
      cosas[0] = new Seta();
      cosas[1] = new Seta();
      cosas[2] = new Tudanca();
      cosas[3] = new Tudanca();
} //test5
```

5. Indique cuáles de los siguientes métodos es correcto y cuál es incorrecto y por qué.

```
public void test6() {

      List<Mamifero> mamiferos = new ArrayList<Mamifero>();
      mamiferos.set(0, new Tudanca());
      mamiferos.set(1, new Tudanca());
      mamiferos.set(2, new Tudanca());
      mamiferos.set(3, new Ballena());

} // test6

public void test7() {

      List<Mamifero> mamiferos = new ArrayList<Mamifero>();
      mamiferos.set(0, new Tudanca());
      mamiferos.set(1, new Tudanca());
      mamiferos.set(2, new Tudanca());
      mamiferos.set(3, new Tudanca());

      Tudanca pepa = mamiferos.get(1);

} // test7
```

6. Indique cuáles de los siguientes métodos es correcto y cuál es incorrecto.

```
public void test8() {
  List<Tudanca> mamiferos = new ArrayList<Tudanca>();
  mamiferos.set(0, new Tudanca());
  mamiferos.set(1, new Tudanca());
  mamiferos.set(2, new Vaca());
  mamiferos.set(3, new Tudanca());
} // test8

public void test9() {

  List<Tudanca> mamiferos = new ArrayList<Tudanca>();
  mamiferos.set(0, new Tudanca());
  mamiferos.set(1, new Tudanca());

  Tudanca tud = mamiferos.get(0);

} // test9
```

7. Como bien es sabido, un buen pasiego debe saber hacer tanto quesadas como sobaos, dos magníficos productos típicos de tan bella tierra. Por otro lado, un lebaniego es capaz de producir tanto un magnífico orujo como un magnífico cocido. Uniendo ambas habilidades, es posible crear una nueva especie superior de ser humano. Dicho nuevo superhombre (o supermujer) será el *PasiegoLebaniego*, y será capaz de producir por sí mismo ricos cocidos, orujos, quesadas y sobaos.

   Aplicar el patrón *mixin* para crear la clase *PasiegoLebaniego* a partir de las clases proporcionadas en la Figura 2 mediante herencia múltiple en un lenguaje tipo Java que no soporte herencia múltiple entre clases, pero si permita herencia múltiple de interfaces.

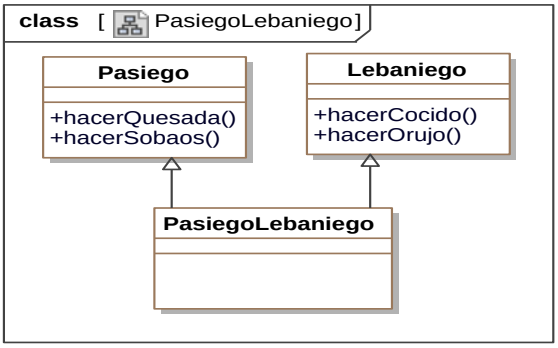

Figura 2. Jerarquía de clases para la clase *PasiegoLebaniego*

Pablo Sánchez Barreiro.