Programa de interfaz, clase abstracta y polimorfismo

El programa ejemplifica brevemente el polimorfismo de objetos que heredan desde una interfaz y una clase abstracta en este caso.

En primer lugar se tiene una interfaz llamada Animal, esta solamente contiene de la firma del método emitirSonido() que sera el que implementen las futuras clases que tengan herencia de esta.

```
interface Animal {
    void emitirSonido();
}
```

Luego tenemos la clase abstracta Mamifero que implementa la interfaz Animal, aquí se tiene el método mover(), que sera el que definan su comportamiento las clases que hereden de esta

```
abstract class Mamifero implements Animal {

    // Método abstracto a implementar
    public abstract void mover();
}
```

Después tenemos dos clases que heredan de Mamifero, ambas con los mismos métodos, que son Persona y Perro, en estas definimos los métodos que heredamos tanto de la clase abstracta como de la interfaz

```
class Persona extends Mamifero {
    @Override
    public void emitirSonido() {
        System.out.println("Habla una persona.");
    }

    @Override
    public void mover() {
        System.out.println("La persona camina.");
    }
}

class Perro extends Mamifero {
```

```
    @Override
    public void emitirSonido() {
        System.out.println("Un perro ladra.");
    }

    @Override
    public void mover() {
        System.out.println("El perro corre.");
    }
}
```

Ya en el método main, creamos varios objetos del tipo Perro y Persona asignándolos a variables de referencia del tipo Animal, esto con el fin de explotar el polimorfismo. Ejemplificando esto creamos un arrayList y agregamos en el los objetos creados, luego con un ciclo foreach iteramos el arrayList y mandamos a llamar los métodos. en el caso de mover() fue necesario hacer un downcasting ya que el tipo animal no conoce el método mover()

```
public static void main(String[] args) {
        Animal a1 = new Persona();
        Animal a2 = new Perro();
        Animal a3 = new Persona();
        Animal a4 = new Perro();

        List<Animal> animales = new ArrayList<>();

        animales.add(a1);
        animales.add(a2);
        animales.add(a3);
        animales.add(a4);

        for (Animal A : animales) {
            A.emitirSonido();
            if (A instanceof Mamifero) {
                ((Mamifero)A).mover();    //downcasting
            }
            System.out.println("----------");
        }

    }
```

Muestra por pantalla los resultados de los métodos dependiendo del tipo de objeto