```
class Dillo {
  int length ;
  boolean isDead ;

  Dillo (int length, boolean isDead) {
    this.length = length ;
    this.isDead = isDead ;
  }

  // determines whether Dillo is dead and longer than 60
  boolean canShelter() {
    return (this.isDead && this.length > 60);
  }

  // determine whether Dillo length less than given one
  public boolean lengthBelow(int someLength) {
    return this.length < someLength;
  }
}
```

```
class Boa implements IAnimal {
  String name ;
  int length ;
  String eats ;

  Boa (String name, int length, String eats) {
    this.name = name ;
    this.length = length ;
    this.eats = eats ;
  }

  // determine whether Boa length less than given one
  public boolean lengthBelow(int someLength) {
    return this.length < someLength;
  }
}
```

implements IAnimal { ← says Boa is a valid animal

method required by interface must be public

```
class Cage {
  int size;              // length of cage in inches
  IAnimal resident;      // any animal can live in cage

  Cage(int size, IAnimal resident) {
    this.size = size;
    this.resident = resident;
  }

  // determine whether animal will fit in this cage
  boolean checkResidentFits() {
    return resident.lengthBelow(this.size);
  }
}
```

Want to allow either Boa or Dillo in a ~~cage~~ cage.

Java needs to be sure that every resident (an IAnimal) has a lengthBelow method. Listing the method in the interface does that

```
interface IAnimal {

  public boolean lengthBelow (int someLength);

}
```

↑ header of method that every IAnimal will have

(interface is not in a class - it's its own thing, can be in own file)