# Aufgabe 1

## Gegeben

### Klasse

|  |  |
| --- | --- |
|  |  |
|  |  |
|  | public class Oryx **implements Comparable<Oryx>** { |
|  |  |
|  | /\*\* Alter des Tiers \*/ |
|  | private Integer age; |
|  | /\*\* Schulterhöhe \*/ |
|  | private Double acromion; |
|  | /\*\* Besondere Merkmale \*/ |
|  | private String characteristics; |
|  | /\*\* Eindeutige Bezeichnung \*/ |
|  | private final String description; |
|  |  |
|  | public Oryx(final String description) { |
|  | age = 0; |
|  | acromion = 0d; |
|  | this.description = description; |
|  | } |
|  | public String getDescription() { |
|  | return description; |
|  | } |
|  | public Integer getAge() { |
|  | return age; |
|  | } |
|  | public void setAge(Integer age) { |
|  | this.age = age; |
|  | } |
|  | public Double getAcromion() { |
|  | return acromion; |
|  | } |
|  | public void setAcromion(Double acromion) { |
|  | this.acromion = acromion; |
|  | } |
|  | public String getCharacteristics() { |
|  | return characteristics; |
|  | } |
|  | public void setCharacteristics(String characteristics) { |
|  | this.characteristics = characteristics; |
|  | } |
|  |  |
|  | **@Override** |
|  | **public int compareTo(Oryx rhs) {** |
|  | **return rhs.description.compareTo(description);** |
|  | **}** |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  | } |

### Testklasse

public class OryxSetTest {

@Test

public void shouldBeTreeSet() {

// given

final TreeSet<Oryx> collection = new TreeSet<>();

final Oryx oryx1 = new Oryx("Otavi");

final Oryx oryx2 = new Oryx("Outjo");

// when

collection.add(oryx1);

collection.add(oryx2);

// then

assertThat(collection.size(), is(2));

assertThat(collection, contains(oryx2, oryx1));

}

}

## Frage

Wird dieser JUnit-Test "Grün"? Wenn nicht, was muss in der Klasse 'Oryx' geändert werden, damit der Test korrekt durchläuft? Passen Sie gegebenenfalls die Klasse 'Oryx' entsprechend an.

# Aufgabe 2

## Gegeben

### Klasse

public class Klipspringer {

public enum Sex { MALE, FEMALE, NOT\_SPECIFIC };

/\*\* Eindeutige Bezeichnung \*/

private final String description;

/\*\* Geschlecht \*/

public Sex sex;

/\*\* Alter des Tiers \*/

private Integer age;

public Klipspringer(String description) {

super();

this.description = description;

sex = Sex.NOT\_SPECIFIC;

age = 0;

}

public Sex getSex() {

return sex;

}

public void setSex(Sex sex) {

this.sex = sex;

}

public Integer getAge() {

return age;

}

public void setAge(Integer age) {

this.age = age;

}

public String getDescription() {

return description;

}

}

### Testklasse

public class KlipspringerCollectionTest {

private Collection<Klipspringer> collection = new ArrayList<>();

@Before

public void setup() {

final Klipspringer animal = new Klipspringer("Aldebaran");

animal.setSex(Sex.FEMALE);

collection.add(animal);

}

@Test

public void shouldBeContains() {

// given

final Klipspringer animal = new Klipspringer("Aldebaran");

animal.setSex(Sex.FEMALE);

// when

boolean isExists = collection.contains(animal);

// then

assertThat(isExists, is(true));

}

}

## Frage

Der Test zeigt das Soll-Verhalten der Klasse auf. Läuft der JUnit-Test durch bzw. wird er 'Grün'? Wenn nein, beschreiben was geändert werden müsste, damit der Test korrekt durchläuft.

# Aufgabe 3

## Gegeben

public class Child implements Comparable<Child> {

private final int age;

private final String name;

public Child(int age, String name) {

super();

this.age = age;

this.name = name;

}

public int getAge() {

return age;

}

public String getName() {

return name;

}

@Override

public int compareTo(Child other) {

return this.age - other.age;

}

}

Und dieser Code:

Set<Child> sortedList = new TreeSet<>();

sortedList.add(new Child(9, "Mélina"));

sortedList.add(new Child(3, "Sonia"));

sortedList.add(new Child(9, "Jaquelina"));

sortedList.add(new Child(11, "Alais"));

sortedList.forEach(c -> System.out.println(c.getName()));

## Frage

Wie sieht das Resultat aus? Was wird in der Console ausgegeben?

## Ausgabe

Sonia

Mélina

Alais