

Fundamentalstest Vaccination

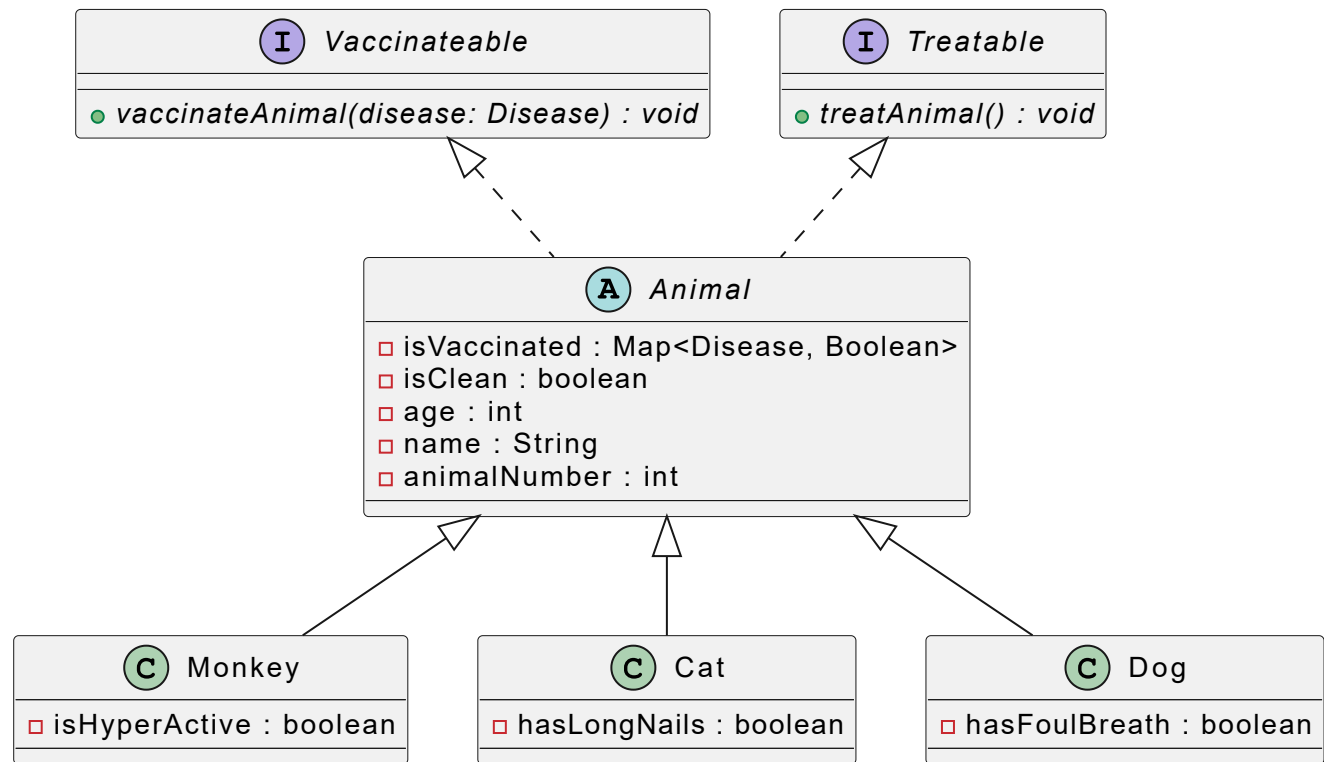
OPDRACHT 1

UML legende


C	Legende
□	private property
□	<i>private abstract property</i>
○	public property
◇	protected property
■	private method()
■	<i>private abstract method()</i>
●	public method()
◆	protected method()


UML

Deel 1



Deel 2

 AnimalShelter
<div> <div>□ animals : List<Animal></div> <div>□ animalId : int</div> </div>
<div> <div>● printAnimals() : void</div> <div>● sortAnimals() : void</div> <div>● sortAnimalsByName() : void</div> <div>● sortAnimalsByAge() : void</div> <div>● printAnimalsNotVaccinated(disease : Disease) : void</div> <div>● findAnimal(animalNumber : int) : Animal</div> <div>● findAnimal(name : String) : Animal</div> <div>● treatAnimal(animalNumber : int) : void</div> <div>● treatAnimal(name : String) : void</div> <div>● treatAllAnimals() : void</div> <div>● findOldestAnimal() : Animal</div> <div>● countAnimals() : int</div> <div>● addAnimal() : void</div> </div>

 Disease
<div> <div>CHICKENPOCKS</div> <div>FLUE</div> <div>POLIO</div> <div>HEPATITISA</div> </div>

Beschouw bevenstaand schema.

Maak de klassen zoals bovenaan aangegeven. Maak ook telkens een constructor zonder parameter en een constructor waar je als parameterlijst alle gegevens meegeeft buiten een eventuele map of lijst. De map of lijst initialiseer je in de declaratiestatement.

Let wel op! De map met vaccinatiegegevens zal initieel bestaan uit een map waar initieel alle Diseases in staan die we hebben en waar de vaccinatie false zal zijn.

Maak ook de nodige getters en setters aan.

Implementeer de interface methoden op de juiste plaats!

Uitleg over de methoden

- **vaccinateAnimal(Disease)** Deze method zal een dier vaccineren voor een bepaalde ziekte. Zet er de boolean dus voor op true.
- **treatAnimal()** zorgt ervoor dat een animal clean gemaakt wordt. Indien je aan het werken bent met een cat, dog of monkey, zorg er dan voor dat de treatAnimal methode ook de hyperactivity, de longnails en de foulbreath problemen oplost. (op false zet)
- **sortAnimals()** sorteert de dieren volgens hun natuurlijke volgorde, dit is volgens hun animalNumber.
- **sortAnimalsByName()** sorteert de dieren op naam.
- **sortAnimalsByAge()** sorteert de dieren op leeftijd.
- **printAnimalsNotVaccinated(Disease)** print alle dieren af die niet gevaccineert zijn voor een opgegeven ziekte.
- **findAnimal(int)** zoek dier op dierennummer
- **findAnimal(String)** zoek dier op diernaam
- **treatAnimal(int)** behandel opgegeven dier
- **treatAnimal(String)** behandel opgegeven dier
- **treatAllAnimals()** behandel alle dieren
- **findOldestAnimal()** geef het oudste dier terug

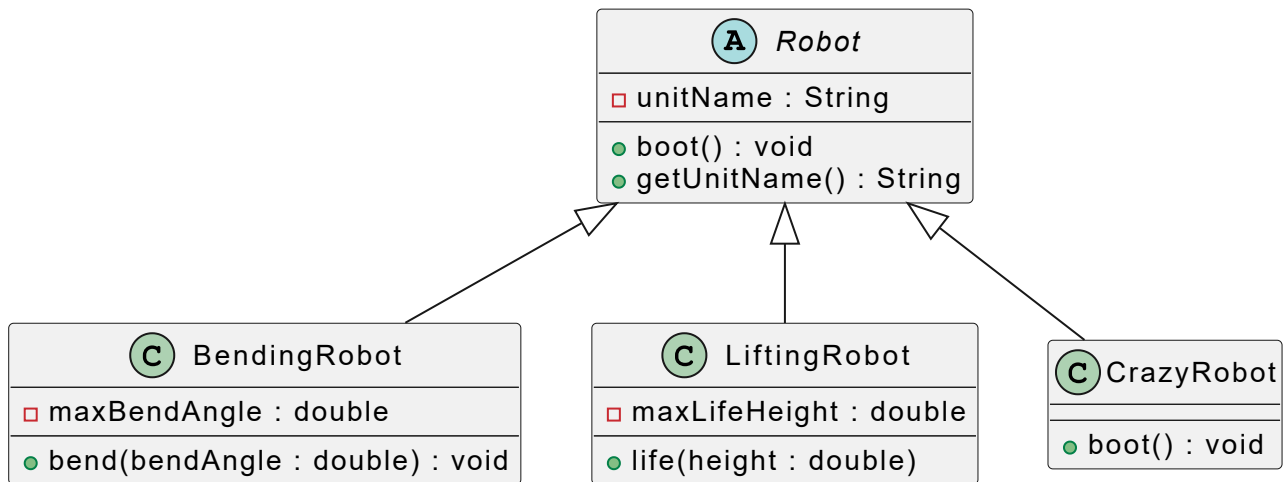
- **countAnimals()** geef het aantal dieren terug
- **addAnimal(Animal animal)** voeg een dier toe aan de lijst van animals. De animalNumber van de animal krijgt de waarde van animalId. Daarna verhoog je de animalId met 1. Zo krijgt elke animal een unieke nummer.

App

Maak een app aan waarin je 10 dieren aanmaakt en toevoegt aan een zelfgemaakte animalshelter. Test alle methoden van de animalshelter klasse.

OPDRACHT 2

UML



Beschouw bovenstaande schema

- **Robot** is een abstracte klasse met 2 constructors, 1 zonder parameters en een met een String als parameter (`unitName`). Bij de constructor zonder parameters geef je als `unitName` "nameless Robot".
- **BendingRobot** heeft een constructor waar je een double `maxBendangle` aan meegeeft en een String `unitName`.
- **LiftingRobot** heeft een constructor waar je een double `maxLiftHeight` aan meegeeft en een String `unitName`.
- **CrazyRobot** heeft een constructor zonder parameters en een constructor waar je een String `unitName` aan meegeeft.

Uitleg bij methoden

- **boot()** in **Robot** zal een bericht afprinten dat de robot aan het opstarten is. Deze boot methode zal tijdens de constructie opgeroepen worden. Ze zal ook de naam van de robot (`Unitname`) afprinten.
- **bend(double)** zal als resultaat een bericht afprinten dat het buigen niet mogelijk is. (indien het argument groter is dan de `maxBendAngle`, hou rekening met het feit dat $360^\circ == 0^\circ$) Ofwel komt er een bericht dat het buigen gelukt is met de gevraagde hoek.
- **lift(double)** zal als resultaat een bericht afprinten dat het buigen niet mogelijk is. (indien het argument groter is dan de `maxLiftHeight`) Ofwel komt er een bericht dat het opheffen gelukt is met de gevraagde hoogte.
- De **crazyRobot** zal zijn eigen naam verbuigen en omdraaien tijdens het boot process. De robot zal dus de boot methode die in **Robot** staat oproepen maar tevens de tekens in zijn eigen naam omdraaien! Print in het bootbericht ook de nieuwe naam af. (bv. Gregory -> yrogerG)
- Zorg er ook voor dat er een `toString` methode bestaat die de nodige gegevens van iedere robot teruggeeft.

App

In je app maak je van elke type robot een instantie en test je 2 keer de lift en bend methoden, telkens 1 keer met een te groot argument en 1 keer met een goed.