OOP Kohiro Sannomiya: Documentation, 2nd Assignment

Teréz A. Várkonyi **2. assignment/9.task** **7th May 2023**

CMDC1H

cmdc1h@inf.elte.hu

Group 4

Task

Hobby animals need several things to preserve their exhilaration. Cathy has some hobby animals: fishes, birds,and dogs. Every animal has a name and their exhilaration level is between 0 and 100 (0 means that the animals die). If their keeper is in a good mood, she takes care of everything to cheer up her animals, and their exhilaration level increases: of the fishes by 1, of the birds by 2, and of the dogs by 3. On an ordinary day, Cathy takes care of only the dogs (their exhilaration level does not change), so the exhilaration level of the rest decreases: of the fishes by 3, of the birds by 1. On a bad day, every animal becomes a bit sadder and their exhilaration level decreases: of the fishes by 5, of the birds by 3, and of the dogs by 10. Cathy's mood improves by one if the exhilaration level of every alive animal is at least 5. Every data is stored in a text file. The first line contains the number of animals. Each of the following lines contain the data of one animal: one character for the type (F – Fish, B – Bird, D – Dog), name of the animal (one word), and the initial level of exhilaration.
In the last line, the daily moods of Cathy are enumerated by a list of characters (g – good, o – ordinary, b – bad). The file is assumed to be correct. Name the animal of the lowest level of exhilaration which is still alive at the end of the simulation. If there are more, name all of them!

Analysis

Independent objects in the task are the animals. They can be divided into 3 different groups: Fishes, Birds, and Dogs. All of them have a name and an exhilaration power that can be got. It can be examined what happens to the exhilaration level of the animals when the carekeeper (in this case, Cathy) is in a good, bad, or in an ordinary mood. The carekeeper's mood effects the animal and its exhilaration level in the following way:

Fish:

Mood	Exhilaration Level
Good	+1
Ordinary	-3
Bad	-5

Bird:

Mood	Exhilaration Level

Good	+2
Ordinary	-1
Bad	-3

Dog:

Mood	Exhilaration Level
Good	+3
Ordinary	—
Bad	-10

Plan

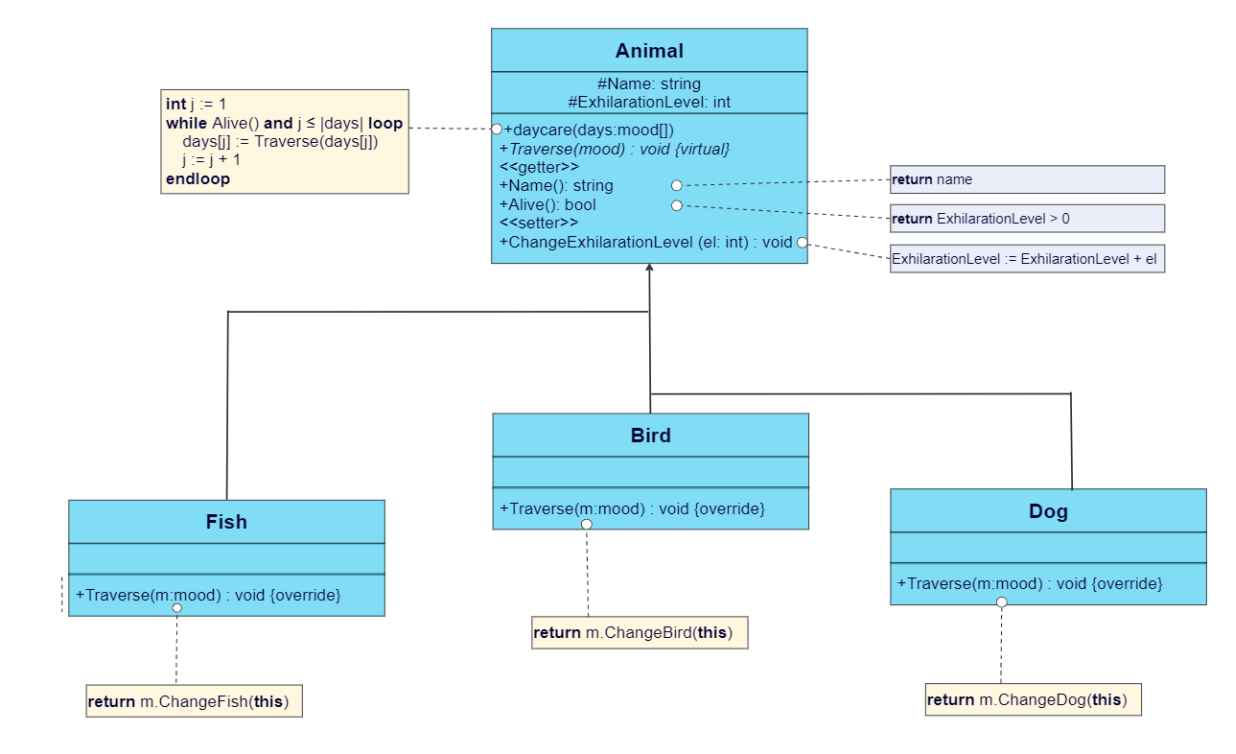
To describe the animals, 4 classes are introduced: base class Animal to describe the general properties and 3 children for the concrete types of Animals: Fish, Bird, and Dog. Regardless of the type of the Animals, they have several common properties, like the name(_name) and the power (_power), the getter of its name (name()), if it is alive (Alive()) and it can be examined what happens to the animal's exhilaration level when the carekeeper's mood changes. This latter operation (Traverse()) modifies the Exhilaration Level of the Animal according to the carekeeper's mood.

Operations alive() and name() may be implemented in the base class already, but Traverse() just on the level of the concrete classes as its effect on the animal's Exhilaration Level depends on the mood of the carekeeper. Therefore, the general class Animal is going to be abstract, as method Traverse() is abstract and we do not wish to instantiate such class.

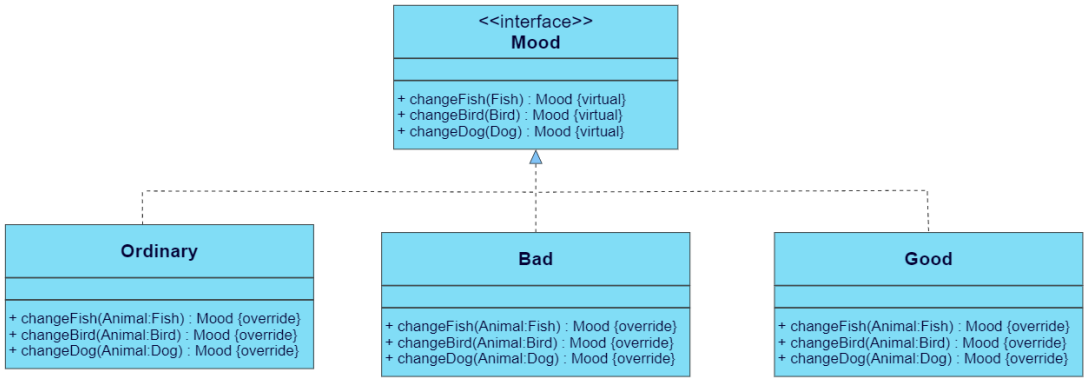
General description of each of the mood is done in the base class Mood from which the concrete moods are inherited: Fish, Bird, and Dog. Every concrete mood has three methods that show how the Exhilaration Level of a Fish, a Bird, or a Dog changes during each of the carekeeper's mood, namely Ordinary, Good and Bad.

The special Animal classes initialize the name and the Exhilaration Level through the constructor of the base class and override the operation Traverse() in a unique way.

Initialization and the override are explained in Section Analysis. According to the tables, in method Traverse(), conditionals could be used in which the type of the mood would be examined. Though, the conditionals would violate the SOLID principle of object-oriented programming and are not effective if the program might be extended by new mood types, as all of the methods Traverse() in all of the concrete creature classes should be modified. To avoid it, the Visitor design pattern is applied where the mood classes are going to have the role of the visitor.



Methods Traverse() of the concrete Animals expect a mood object as an input parameter as a visitor and call the methods which corresponds to the species of the Animal.



All the classes of the moods are realized based on the Singleton design pattern, as it is enough to create one object for each class.

Specification:

$$A = (moods : mood^{days}, animals : Animal^{n}, alive : string^{*})$$

$$Pre = (animals = animals_0 \land moods = moods_0)$$

$$Post = moods = moods_n \land \forall i \in [1..n] : animals[i], moods_i = daycare(animals[i], moods_{i-1}) \quad \land$$

$$minexlevel = \underset{\text{animals[i].ExhilarationLevel}}{\text{MIN}_{i=1..n}} < animals[i].name >$$

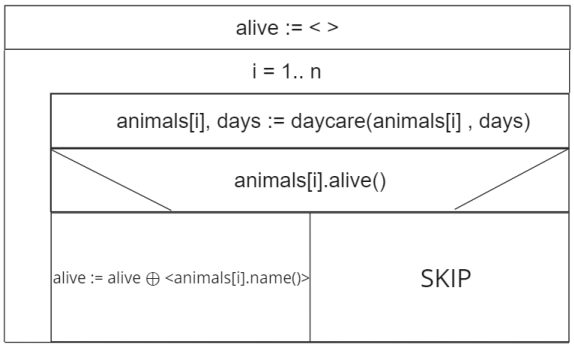
Analogy:

enor(E)	i = 1..n
f(e)	daycare(animals[i] , days)
s	animals
H, +, 0	Animal*, ⊖, animals[i]

enor(E)	i = 1..n
f(e)	daycare(animals[i] , days)
s	days
H, +, 0	mood*, ⊖, days

enor(E)	i = 1..n
f(e)	<animals[i]> if animals[i].Alive()
s	Alive
H, +, 0	Animal*, ⊕, < >

By merging the above to the same loop, the solution is got:



When Cathy’s mood changes with either ordinary,good or bad, the exhilaration level of each animal is changed. So the second task to be solved is below:

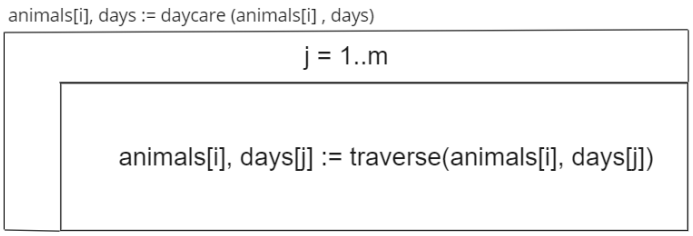
$$\forall j \in [1..m] : animals_j[i], days_i[j] = traverse(animals_{j-1}[i], days_{i-1}[j]) \wedge animals[i] = animals_m[i]$$

Analogy(2nd part):

enor(E)	j = 1..m
f(e)	traverse (animals[i] , days[j])
s	animals[i]
H, +, 0	Animal*, ⊖, animals[i]

enor(E)	j = 1..m
f(e)	traverse (animals[i] , days[j])
s	days[j]
H, +, 0	mood*, ⊕, <>

Merging them into the same loop:



Testing

Testing the number of animals

- 0 animals
- 1 animal
- more than 1 animal

Testing the different exhilaration levels of the animals

- all animals are dead : then it shouldnt output anything
- more than 1 animal with the least exhilaration level