Task

Hobby animals need several things to preserve their exhilaration. Steve has some hobby animals: tarantulas, hamsters, and cats. Every animal has a name and their exhilaration level is between 0 and 70 (0 means that the animals dies). If their keeper is joyful, he takes care of everything to cheer up his animals, and their exhilaration level increases: of the tarantulas by 1, of the hamsters by 2, and of the cats by 3. On a usual day, Steve takes care of only the cats (their exhilaration level increases by 3), so the exhilaration level of the rest decreases: of the tarantulas by 2, and of the hamsters by 3. On a blue day, every animal becomes a bit sadder and their exhilaration level decreases: of the tarantulas by 3, of the hamsters by 5, of the cats by 7.

Steve'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 (T – Tarantula, H – Hamster, C – Cat), name of the animal (one word), and the initial level of exhilaration. In the last line, the daily moods of Steve are enumerated by a list of characters (j – joyful, u – usual, b – blue). The file is assumed to be correct.

List the animals of the highest exhilaration level at the end of each day.

A possible input file:

3

T Webster 20

H Butterscotch 30

C Cat-man-do 50

Uuuujjbjbjuujj

Analysis

Independent objects in the task are the animals. They can be divided into 3 different groups: Tarantulas, Hamsters and Cats. All of them have a name and a exhilarationLevel that can be got. Their exhilarationLevel changes based on mood of Steve like this:

Tarantulas:

Mood	exhilarationLevel
Joyful	+1
Usual	-2
Blue	-3

Hamsters:

Mood	exhilarationLevel
Joyful	+2
Usual	-3
Blue	-5

Cats:

Mood	exhilarationLevel
Joyful	+3
Usual	+3
Blue	-7

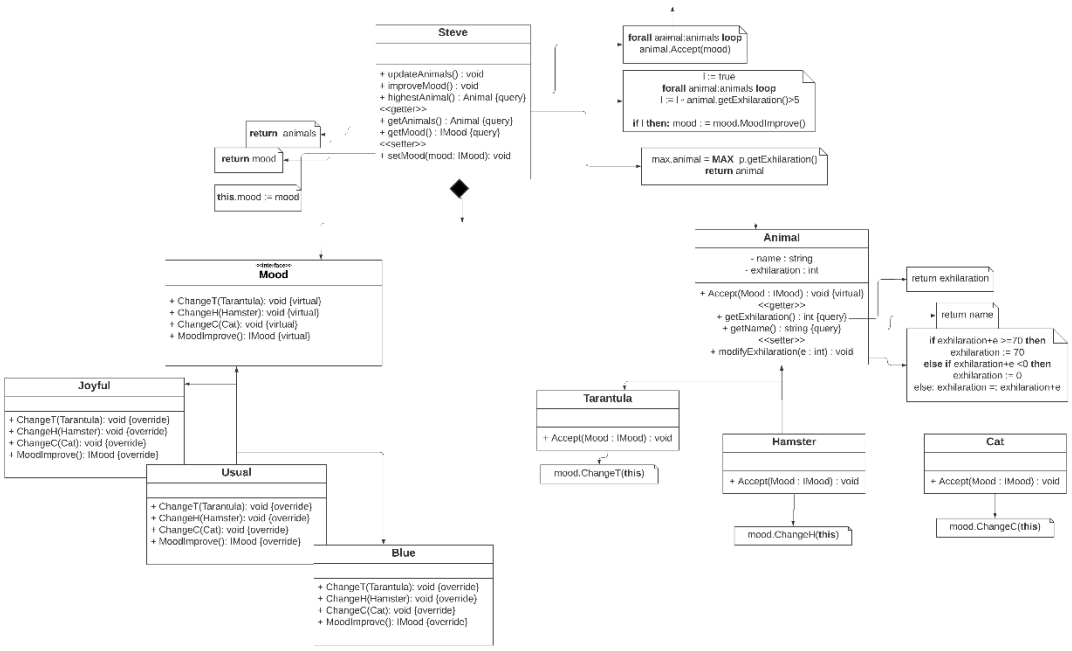
Steve's mood improves by one if the exhilaration level of every alive animal is at least 5. So improved mood of every mood if every animals' exhiliartion level is at least 5 becomes like this:

Mood	improvedMood
Joyful	Joyful
Usual	Joyful
Blue	Usual

Plan

To describe the animals, 4 classes are introduced: base class Animal to describe the general properties and 3 children for the concrete species: Tarantula, Hamster, and Cat. Regardless the type of the creatures, they have several common properties, like the name (_name) and the exhilaration (_exhilaration), the getter of its name (name). The operation (Accept()) modifies the exhilaration of the creature. Operations name() may be implemented in the base class already, but Accept() just on the level of the concrete classes as its effect depends on the species of the creature. Therefore, the general class Animal is going to be abstract, as method Accept() is abstract and we do not wish to instantiate such class. General description of the moods is done the base class Mood from which concrete moods are inherited: Joyful, Usual, and Blue. Every concrete mood has four methods that show how a Tarantula, a Hamster, or a Cat exhilaration level changes and last method for improving current mood . The special animal classes initialize the name and the exhilaration through the constructor of the base class and override the operation Accept() in a unique way. In Steve class, we mood property for Steve's mood, and animals property which represents list of animals Steve have. We have setMood() operation is this class which sets the mood of Steve based on input. In improveMood() method, Steve's current mood changes if all the animal's exhilaration level is at least 5. updateAnimal() method is for updating exhilaration level of each of Steve's animal based on his mood. highestAnimal() method is for finding animal which has highest exhilaration level. Initialization and the override are explained in Section Analysis. According to the

tables, in method Accept(), conditionals could be used in which the type of the mood would 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 Accept() 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.

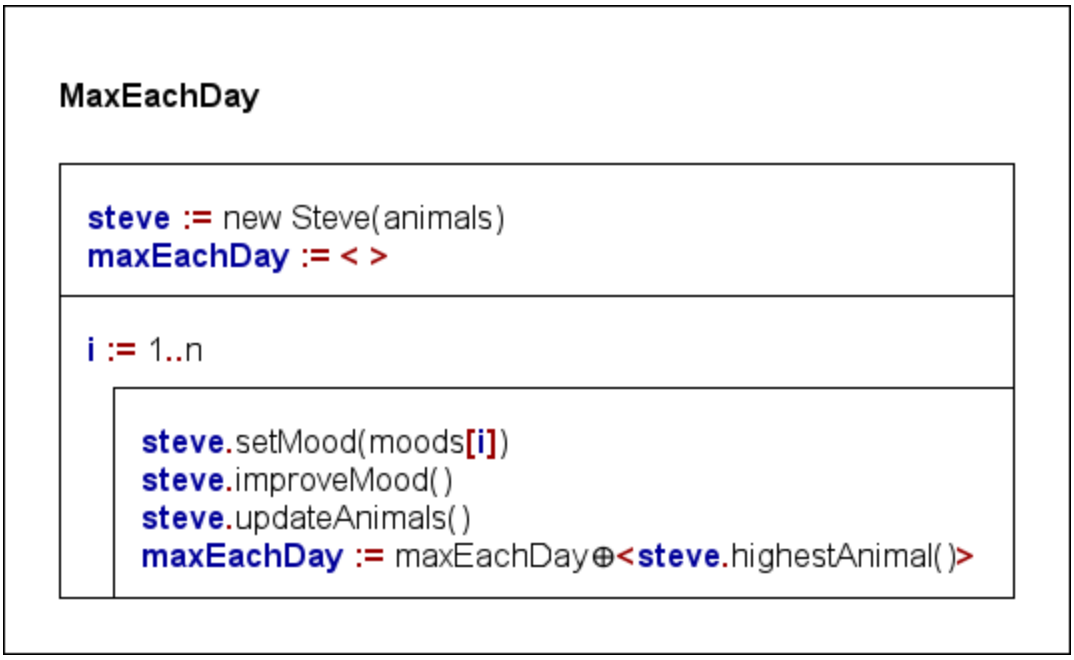


For finding animal with highest exhilaration level each day, we need to use summation like shown below:

A = moods: $IMood^n$, maxEachDay: String*

Pre = moods = moods’ □

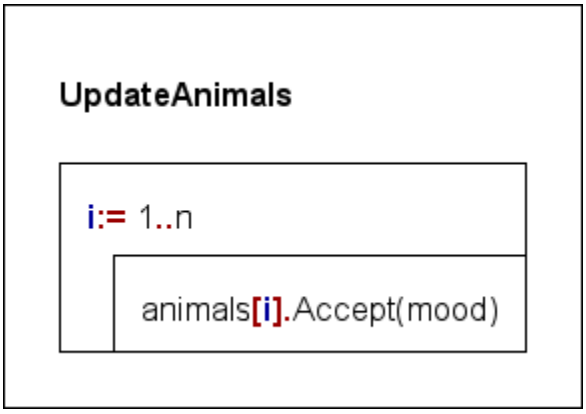
Post = Pre $\wedge$ $\forall i \in [1..n]$: steve.setMood(moods[i]) $\wedge$ steve.improveMood() $\wedge$ steve.updateAnimals() $\wedge$ maxEachDay = $\bigoplus$< steve.highestAnimal()>



A = animals: $Animal^n$, mood: IMood

Pre = animals = animals

Post = Pre $\wedge \forall i \in$ [1..n]: animals[i].Accept(mood)

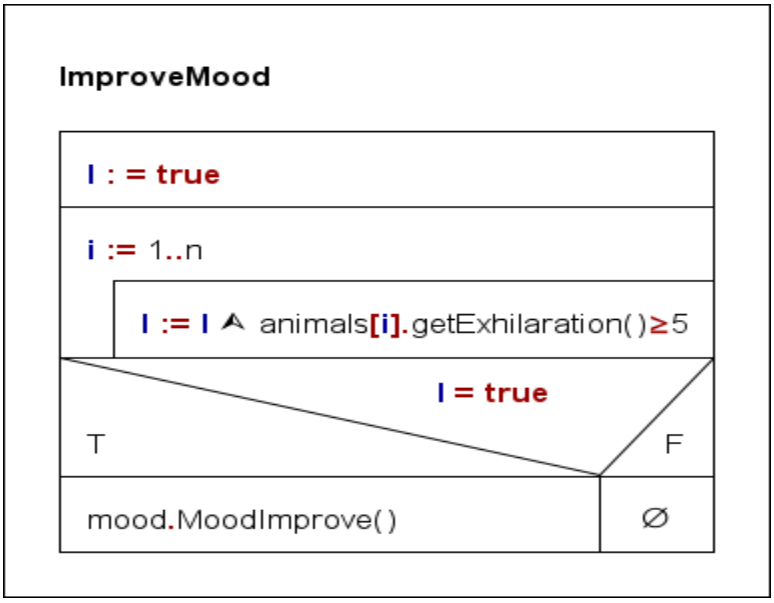


A = animals: $Animal^n$, mood: IMood, l:

Pre = animals = animals'

Post = Pre $\wedge$ l: $\forall$i $\in$1..n]: animals[i].getExhilaration() >= 5

l : mood.MoodImprove()

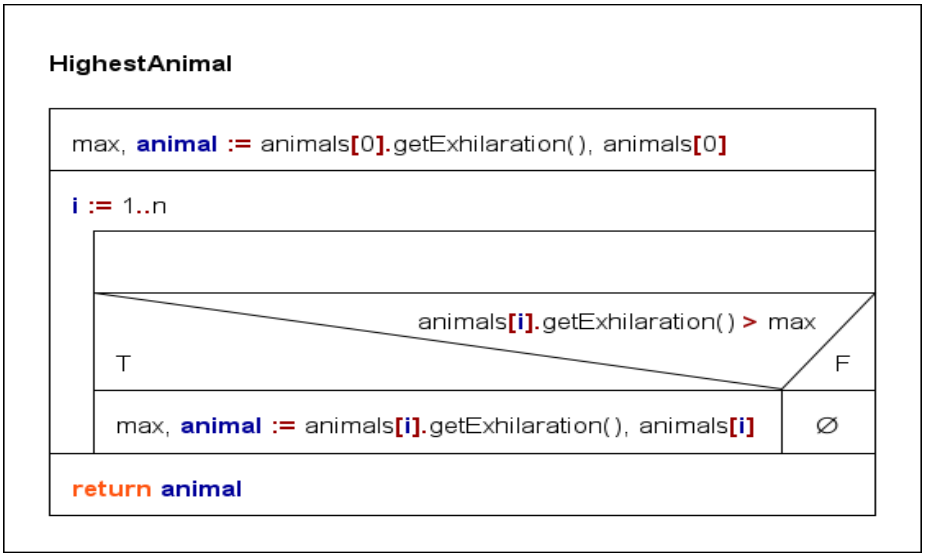


For finding animal with highest exhilaration level, we need to use maximum search pattern like shown below:

A = animals: $Animal^n$, max: Int, animal: Animal

Pre = animals = animals'

Post = Pre $\wedge$ (max,animal) = MAX e.getExhilaration()



Testing

Animal class

1. Checking exception if the animal class is initialized with negative exhilaration
2. Changing exhilaration level
3. Check when exhilaration level reaches max or min

Steve class

1. Updating animals
2. Improving mood
3. Get highest animal