“Object oriented programming”
Documentation to Assignment 2 task 9

Made by: Illia Takhtamyshev

Neptun code: RP0KRP

Group: 06

E-mail: takhtamyshev17@gmail.com

Contents

Task

Hobby animals need several things to preserve their exhilaration. Cathyhas 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 dies). If their keeper is in a goodmood, she takes care of everything to cheer up heranimals, and theirexhilaration 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 leveldoes 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 sadderandtheirexhilaration level decreases: of the fishes by 5, of the birds by 3, and of the dogs by 10.

Cathy'smood 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!

A possible input file:

3

F Nemo 50

B Hedwig 70

D Lassie 20

ooooggbgbgoogg

Plan

```
temp := ""
foreach(Animal a in animals) loop
  temp := temp + a.ToString()
endloop
return temp
```

```
return day
```

```
switch(d):
  case 'g':
    day := GoodDay.Instance()
    break
  case 'o':
    if(everyELevelAtl5()) then
        day := GoodDay.Instance()
        break
    endif
    day := OrdinaryDay.Instance()
    break
  case 'b':
    if(everyELevelAtl5()) then
        day := OrdinaryDay.Instance()
        break
    endif
    day := BadDay.Instance()
    break
endswitch
```

```
below
```

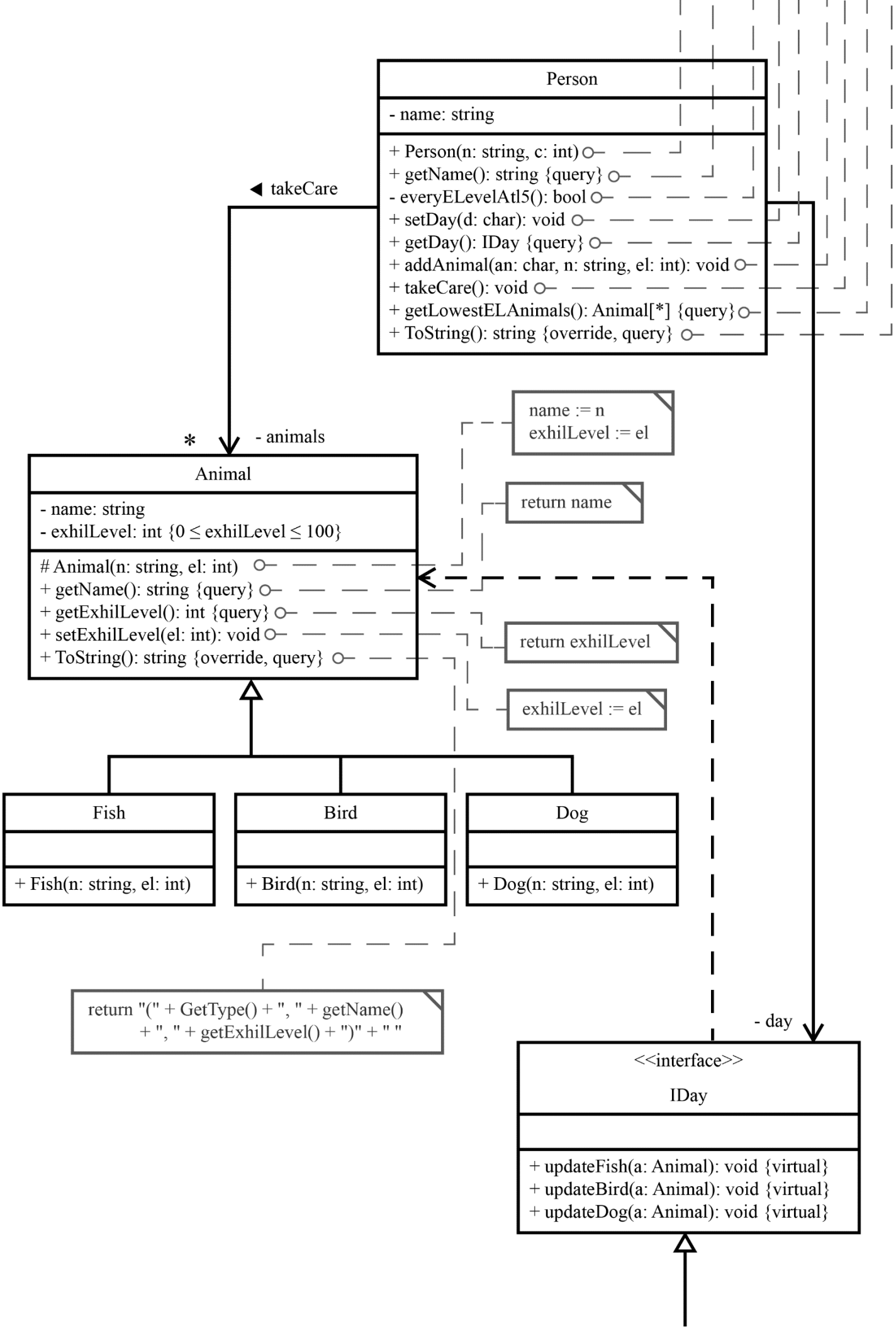
```
foreach(Animal a in animals) loop
  if(a is Fish) then
    day.updateFish(a)
  elseif(a is Bird) then
    day.updateBird(a)
  elseif(a is Dog) then
    day.updateDog(a)
  endif
endloop
```

```
greater := true
for(i = 0 to |animals|) loop
  if(animals[i].getExhilLevel() < 5) then
    greater := false
  endif
endloop
return greater
```

```
switch(an):
  case 'F':
    animals.Add(new Fish(n, el))
    break
  case 'B':
    animals.Add(new Bird(n, el))
    break
  case 'D':
    animals.Add(new Dog(n, el))
    break
endswitch
```

```
return name
```

```
name := n
animals := new List<Animal>(c)
```


Person
- name: string
+ Person(n: string, c: int)
+ getName(): string {query}
- everyELevelAtl5(): bool
+ setDay(d: char): void
+ getDay(): IDay {query}
+ addAnimal(an: char, n: string, el: int): void
+ takeCare(): void
+ getLowestELAnimals(): Animal[*] {query}
+ ToString(): string {override, query}
takeCare
*
- animals
Animal
- name: string
- exhilLevel: int {0 ≤ exhilLevel ≤ 100}
Animal(n: string, el: int)
+ getName(): string {query}
+ getExhilLevel(): int {query}
+ setExhilLevel(el: int): void
+ ToString(): string {override, query}
name := n
exhilLevel := el
return name
return exhilLevel
exhilLevel := el
Fish
+ Fish(n: string, el: int)
Bird
+ Bird(n: string, el: int)
Dog
+ Dog(n: string, el: int)
return "(" + GetType() + ", " + getName()
+ ", " + getExhilLevel() + ")" + " "
- day
<<interface>>
IDay
+ updateFish(a: Animal): void {virtual}
+ updateBird(a: Animal): void {virtual}
+ updateDog(a: Animal): void {virtual}

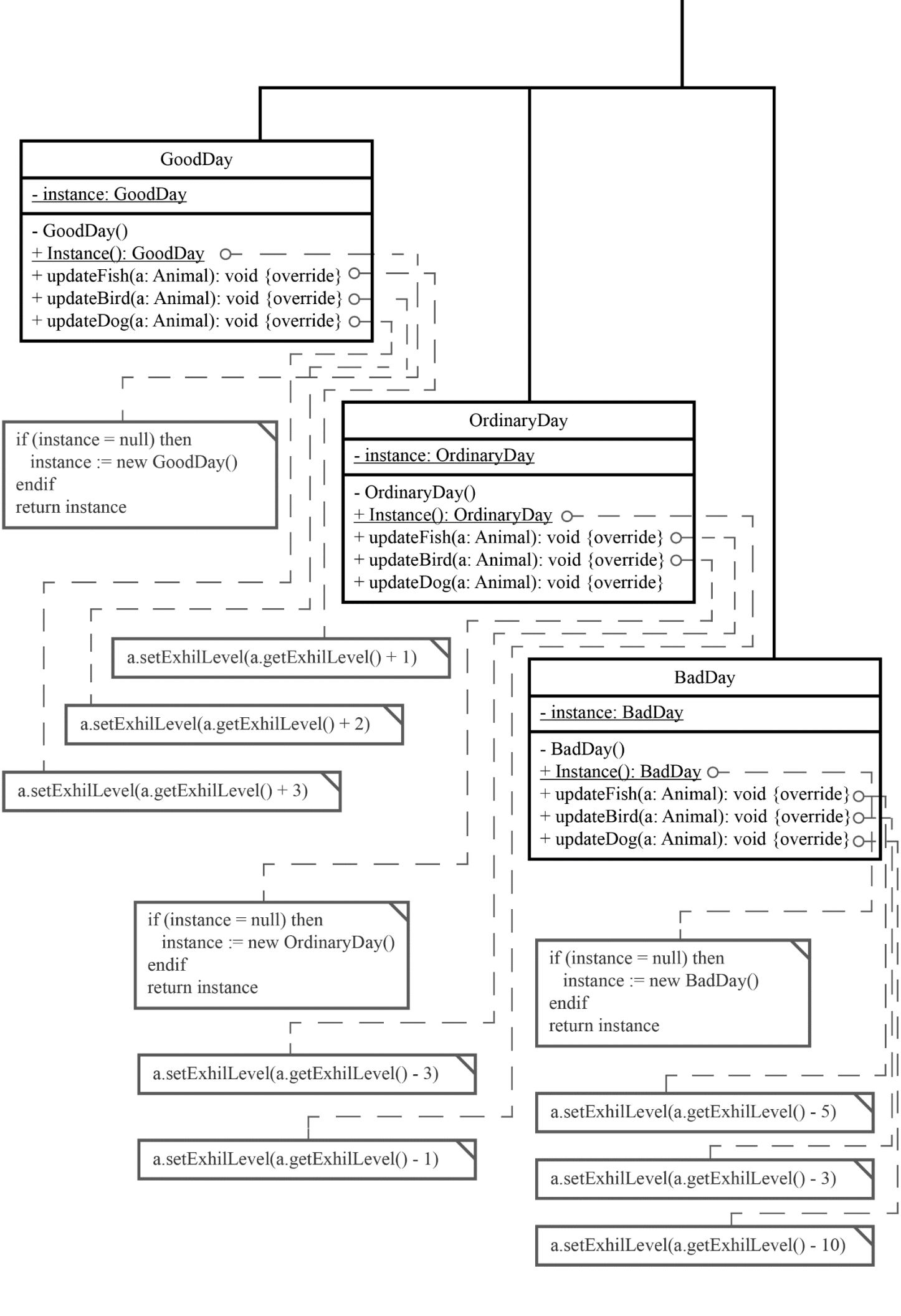

GoodDay
- instance: GoodDay
- GoodDay()
+ Instance(): GoodDay
+ updateFish(a: Animal): void {override}
+ updateBird(a: Animal): void {override}
+ updateDog(a: Animal): void {override}
if (instance = null) then
instance := new GoodDay()
endif
return instance
a.setExhilLevel(a.getExhilLevel() + 1)
a.setExhilLevel(a.getExhilLevel() + 2)
a.setExhilLevel(a.getExhilLevel() + 3)
OrdinaryDay
- instance: OrdinaryDay
- OrdinaryDay()
+ Instance(): OrdinaryDay
+ updateFish(a: Animal): void {override}
+ updateBird(a: Animal): void {override}
+ updateDog(a: Animal): void {override}
if (instance = null) then
instance := new OrdinaryDay()
endif
return instance
a.setExhilLevel(a.getExhilLevel() - 3)
a.setExhilLevel(a.getExhilLevel() - 1)
BadDay
- instance: BadDay
- BadDay()
+ Instance(): BadDay
+ updateFish(a: Animal): void {override}
+ updateBird(a: Animal): void {override}
+ updateDog(a: Animal): void {override}
if (instance = null) then
instance := new BadDay()
endif
return instance
a.setExhilLevel(a.getExhilLevel() - 5)
a.setExhilLevel(a.getExhilLevel() - 3)
a.setExhilLevel(a.getExhilLevel() - 10)

Table animal – day

	Fish	Bird	Dog
GoodDay	+1	+2	+3
OrdinaryDay	-3	-1	0
BadDay	-5	-3	-10

getLowestELAnimals specification

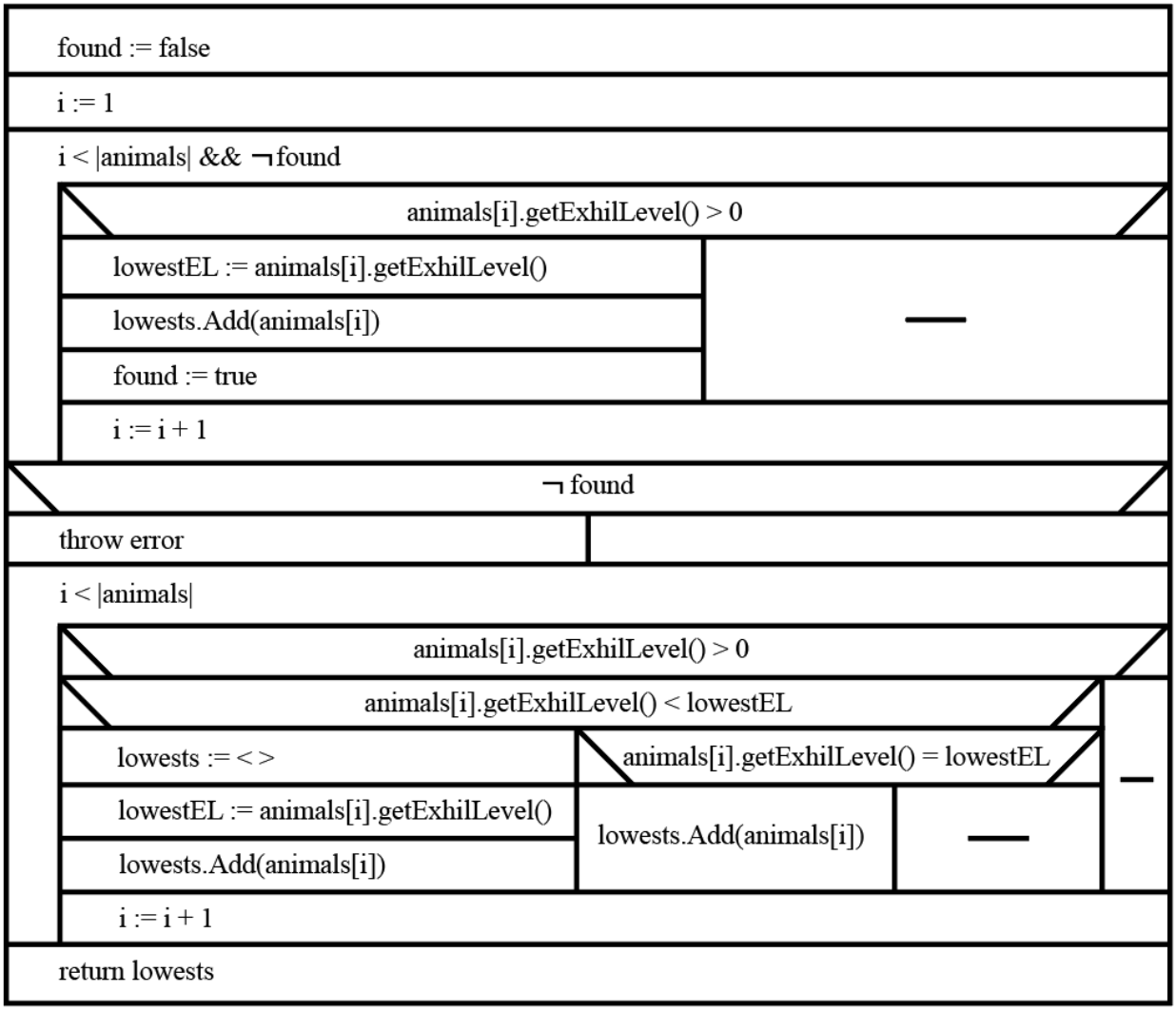
A = (animals : Animal^n, found : $\mathbb{L}$, i : $\mathbb{Z}$, lowestEL : $\mathbb{Z}$, lowests : Animal^n)

Pre = (animals = animals')

Post = (Pre $\wedge$ (found, i, lowestEL, lowests) = $\underset{a \in animals'_{1..n}}{\text{SEARCH}}\ (a.getExhilLevel() > 0)$

$\wedge$ (lowestEL, lowests) = $\underset{\substack{a \in animals'_{i..n} \\ (a.getExhilLevel() > 0)}}{\text{MIN}}\ a.getExhilLevel()$

getLowestELAnimals algorithm



Testing

1) Give an input and check exhilLevels of animals in the output after the simulation

a) Test 1 with input: 4 animals: fish, dog, bird, fish; 3 days: bad, good, ordinary
b) Test 2 with input: 3 animals: bird, fish, bird; 4 days: bad, bad, ordinary, good
c) Test 3 with input: 3 animals: dog, fish, bird; 5 days: bad, good, ordinary, ordinary, ordinary

2) Give an input and check getLowestELAnimals output

a) Test 4 with input: 3 animals: fish, dog, bird; 10 days: ooggbgbgog
b) Test 5 with input: 4 animals: dog, dog, fish, bird; 8 days: ogbgbogg
c) Test 6 with input: 4 animals: bird, fish, dog, fish; 12 days: ggbgooogbgbb
d) Test 7 with input: 3 animals: bird, dog, fish; 7 days: bbbbbbb