Simulation
of
Race: With Creatures

Written and Implemented

By

Kohiro Sannomiya (cmdc1h)

Programming Technology

Eötvös Loránd University

Contents

Task 3
Description of the task 3
UML Diagram 4
Description of the methods 5
Testing 6
White box test cases 6
Black box test cases 6

Task

There is a race for creatures, which takes place on several consecutive days. **Who wins the race? (In other words, which creature can go farthest and remain live?)** At the beginning, each creature has an amount of water, and a distance of 0 from the start. There are three different kind days could occur: sunny, cloudy, rainy. The movement and the water level of a creature are affected by the type of the day and the creature. At first, a creature changes its water level according to the day, and if it is still alive, it moves. A creature dies if it runs out of water (water level drops to 0 or below). A dead creature doesn't move… Properties of creatures: name of the creature (string), water level (integer), maximum water level (integer), living (boolean), distance (integer). The types of creatures on the race are: sandrunner, sponge, walker. The following table contains the properties of the creatures.

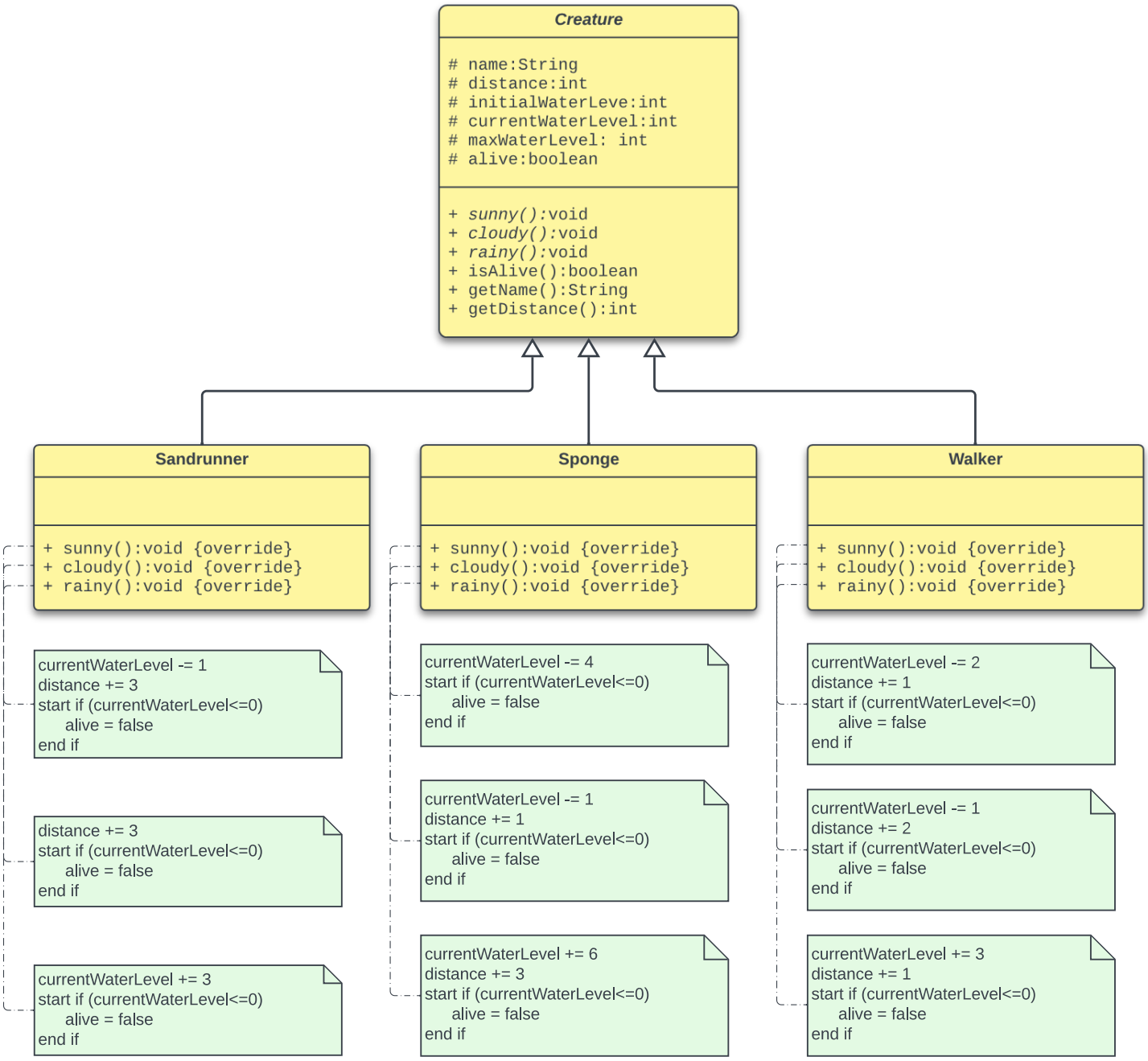
	water change			**distance**			**max. water**
	sunny	cloudy	rainy	sunny	cloudy	rainy	
sandrunner	-1	0	3	3	1	0	8
sponge	-4	-1	6	0	1	3	20
walker	-2	-1	3	1	2	1	12

Creatures cannot have water more than their maximum water level. Read the data of the race from a text file. The first line of the file contains the number of competitors (lets say N). Each of the following N lines contains a competitor: name, type, initial water level. The properties are separated by spaces; and the type is represented with one character: r - sandrunner, s - sponge, w - walker. The last line of the file contains the type of the days on the race: s - sunny, c - cloudy, r - rainy. The program should ask for the name of the file, and it has to print out the name of the winner (we can assume that the file is existing and its format is valid).

Description of the task

Create an abstract Creature class and derive the three kind of creatures. Let this class have a constructor with the parameters of name and initial water level. Introduce three methods for each day (sunny, cloudy, rainy), which updates the water level, checks the life of the creature, and handles the movement. To obtain the result, the following 3 methods are also required: isAlive, getName, getDistance.)

UML Diagram


Creature
name:String
distance:int
initialWaterLeve:int
currentWaterLevel:int
maxWaterLevel: int
alive:boolean
+ sunny():void
+ cloudy():void
+ rainy():void
+ isAlive():boolean
+ getName():String
+ getDistance():int
Sandrunner
+ sunny():void {override}
+ cloudy():void {override}
+ rainy():void {override}
currentWaterLevel -= 1
distance += 3
start if (currentWaterLevel<=0)
alive = false
end if
distance += 3
start if (currentWaterLevel<=0)
alive = false
end if
currentWaterLevel += 3
start if (currentWaterLevel<=0)
alive = false
end if
Sponge
+ sunny():void {override}
+ cloudy():void {override}
+ rainy():void {override}
currentWaterLevel -= 4
start if (currentWaterLevel<=0)
alive = false
end if
currentWaterLevel -= 1
distance += 1
start if (currentWaterLevel<=0)
alive = false
end if
currentWaterLevel += 6
distance += 3
start if (currentWaterLevel<=0)
alive = false
end if
Walker
+ sunny():void {override}
+ cloudy():void {override}
+ rainy():void {override}
currentWaterLevel -= 2
distance += 1
start if (currentWaterLevel<=0)
alive = false
end if
currentWaterLevel -= 1
distance += 2
start if (currentWaterLevel<=0)
alive = false
end if
currentWaterLevel += 3
distance += 1
start if (currentWaterLevel<=0)
alive = false
end if

Description of the methods

Creature

Creature(String name, int initialWaterLevel): Constructor. Created with the name and initial water level.

sunny(): It keeps tab of the creature's activity during the sunny days. Abstract method.

cloudy(): It keeps tab of the creature's activity during the cloudy days. Abstract method.

rainy(): It keeps tab of the creature's activity during the rainy days. Abstract method.

isAlive(): Checks whether the creature is alive or not. Returns true or false.

getName(): Returns the name of the creature.

getDistance(): Returns the distance of the creature.

Sandrunner

Sandrunner(String name, int initialWaterLevel): Constructor. Along with the name and initial water level, the maxWaterLevel is set to 8.

sunny(): It keeps tab of the creature's activity during the sunny days. Decreases the water level by 1 and increases the distance by 3. If the water level is below 0, the creature is not alive.

cloudy(): It keeps tab of the creature's activity during the cloudy days. Increases the distance by 1. If the water level is below 0, the creature is not alive.

rainy(): It keeps tab of the creature's activity during the rainy days. Increases the water level by 3. If the water level is below 0, the creature is not alive.

Sponge

Sponge(String name, int initialWaterLevel): Constructor. Along with the name and initial water level, the maxWaterLevel is set to 20.

sunny(): It keeps tab of the creature's activity during the sunny days. Decreases the water level by 4. If the water level is below 0, the creature is not alive.

cloudy(): It keeps tab of the creature's activity during the cloudy days. Decreases the water level by 1 and increases the distance by 1. If the water level is below 0, the creature is not alive.

rainy(): It keeps tab of the creature's activity during the sunny days. Increases the water level by 6 and increases the distance by 3. If the water level is below 0, the creature is not alive.

Walker

Walker(String name, int initialWaterLevel): Constructor. Along with the name and initial water level, the maxWaterLevel is set to 12.

sunny(): It keeps tab of the creature's activity during the sunny days. Decreases the water level by 2 and increases the distance by 1. If the water level is below 0, the creature is not alive.

cloudy(): It keeps tab of the creature's activity during the cloudy days. Decreases the water level by 1 and increases the distance by 2. If the water level is below 0, the creature is not alive.

rainy(): It keeps tab of the creature's activity during the rainy days. Increases the water level by 3 and increases the distance by 1. If the water level is below 0, the creature is not alive.

Race

main(String[] args): The main (driving) class for simulation of a race involving the creatures. It reads the weather conditions and creatures from a file and picks out the winner.

newRacer(String name, char type, int initialWaterLevel): Creates and initializes a new creature with the name and initial water level.

Testing

Black box test cases

Type of test	**Input**	**Expected Output**
Default Input	input.txt	"Winner: slider"
Empty File Content	input2.txt	"The file is empty."
Wrong Creature Type	input3.txt	"No such Creature exists. No one won."
No Creatures	input4.txt	"No one won."
All Creatures Dead	input5.txt	"No one won."

- input.txt contains the default values which should return slider as an output.
- input2.txt is an empty file so the program should throw an exception with the message written above.
- input3.txt contains a type of creature asides from w, r, or s, so the program should throw an exception with the message written above.
- input4.txt contains 0 creatures so the expected output is "no one won".
- Input5.txt is a file which contains the case where none of the animals make it because their water level reaches below 0 thus the message "no one won" should be outputted.

White box test cases

Test	Input	Expected Output
sunny()	Sandrunner(“Sand”,5)	sandrunner.currentWaterLevel == 4 && sandrunner.distance == 3
cloudy()	Sponge("Spon", 10)	sponge.currentWaterLevel == 9 && sponge.distance == 1
rainy()	Walker(“Walk”, 15)	walker.currentWaterLevel == 18 && walker.distance == 1

- Testing the sunny() method by giving a creature with the type Sandrunner with an initial water level of 5.
- Testing the cloudy() method by giving a creature with the type Sponge with an initial water level of 10.
- Testing the rainy() method by giving a creature with the type Walker with an initial water level of 15.