Sarun Kumar **Assignment 1 / Task 2** **07-10-2022**
IWCYYD
IWCYYD@INF.ELTE.HU

Task

There is a planet, where different kind of plants are living. All the plants are using nutrients to live. If a plant runs out of its nutrients, it dies. Each day one radiation type can occur from the followings: alpha, delta, or no radiation. Radiations affect the plants differently based on their types. The reaction of a plant to a given radiation consists of the following: it changes its nutrient level, and affects the radiation of the next day. The radiation of the next day:

1. alpha, if the need for alpha radiation is 3 or more greater than for the delta radiation
2. delta, if the need for delta radiation is 3 or more greater than for the alpha radiation
3. no radiation, otherwise

There is no radiation on the first day...

Simulate the behaviors of the plants, and print out the radiation of the day and the properties of the plants on each day.

Properties of the plants: name (string), nutrients (integer), living (boolean). The types of the plants in the simulation: puffs, deltatree, parabush.
On a day of the the simulation the living plant first changes its nutrients, then if it is still alive, it can affect the radiation of the next day.

	nutrients (N)			radiation need on next day			dies
	alpha	delta	no radiation	alpha	delta	no radiation	
Puffs	+2	-2	-1	10-N			10<N
Deltatree	-3	+4	-1		+4, if N < 5 +1, if 5 ≤ N ≤ 10		
Parabush	+1	+1	-1				

Read the data of the simulation from a text file. The first line contains the number (n) of the plants. The following n lines contain the information about the plants: name, type, initial nutrient level. Type is represented by one character: p - Puffs, d - Deltratree, b - Parabush. The last line of the file defines the number of the days you have to simulate.

The program should ask for the name of the file, and it has to print out the name of the survivors (we can assume that the file is existing and its format is valid).

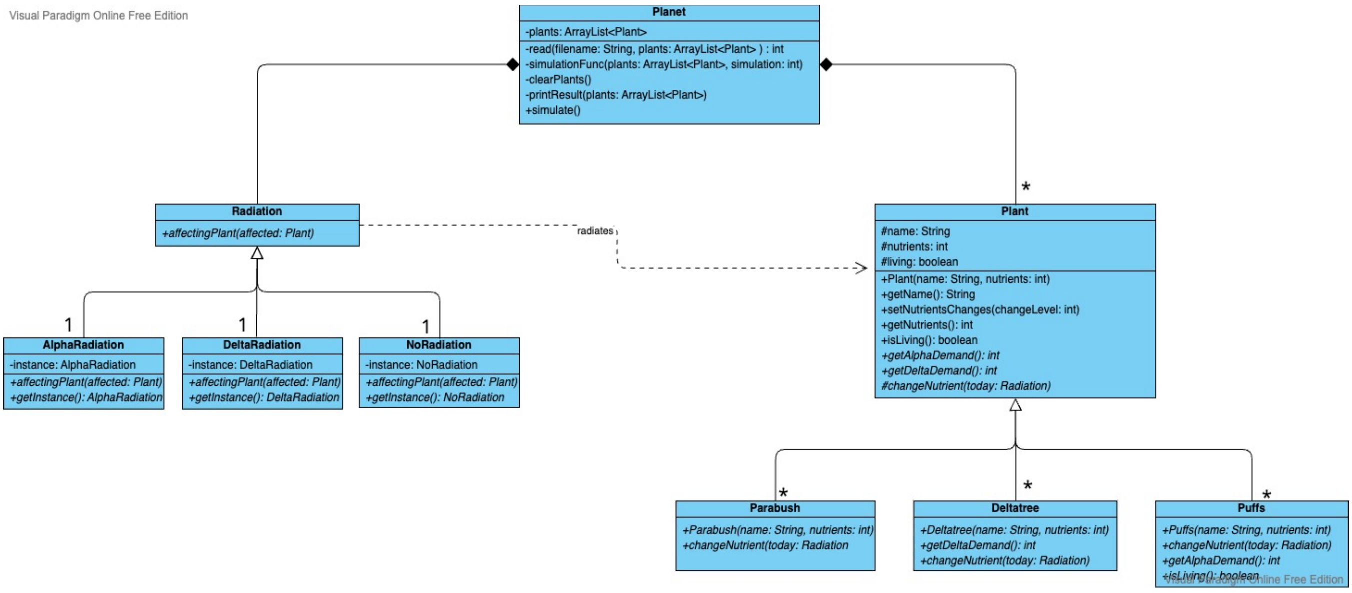
A sample file content:

```
  4
 Piggy  p 7
Slender d 5
Dumpy b 4
Willowy d 3
10
```

Plan

The strategy is to use visitor design pattern to and change the nutrient level according to radiation type which is determined by collecting demand of radiation by each plant. Furthermore, by summing the demand for different radiation we can decide which radiation will occur following day. We can simulate the plants simultaneously calculating the radiation for following day(only after the plant is already affected by today's radiation).

Class Diagram


Visual Paradigm Online Free Edition
Planet
-plants: ArrayList<Plant>
-read(filename: String, plants: ArrayList<Plant>) : int
-simulationFunc(plants: ArrayList<Plant>, simulation: int)
-clearPlants()
-printResult(plants: ArrayList<Plant>)
+simulate()
Radiation
+affectingPlant(affected: Plant)
radiates
1
1
1
AlphaRadiation
-instance: AlphaRadiation
+affectingPlant(affected: Plant)
+getInstance(): AlphaRadiation
DeltaRadiation
-instance: DeltaRadiation
+affectingPlant(affected: Plant)
+getInstance(): DeltaRadiation
NoRadiation
-instance: NoRadiation
+affectingPlant(affected: Plant)
+getInstance(): NoRadiation
*
Plant
#name: String
#nutrients: int
#living: boolean
+Plant(name: String, nutrients: int)
+getName(): String
+setNutrientsChanges(changeLevel: int)
+getNutrients(): int
+isLiving(): boolean
+getAlphaDemand(): int
+getDeltaDemand(): int
#changeNutrient(today: Radiation)
*
*
*
Parabush
+Parabush(name: String, nutrients: int)
+changeNutrient(today: Radiation
Deltatree
+Deltatree(name: String, nutrients: int)
+getDeltaDemand(): int
+changeNutrient(today: Radiation)
Puffs
+Puffs(name: String, nutrients: int)
+changeNutrient(today: Radiation)
+getAlphaDemand(): int
+isLiving(): boolean
Visual Paradigm Online Free Edition

Some important method description:

1. -read(filename: String, plants ArrayList<Plant>): int
   - This function uses Scanner class to read the file and store the data from it.
   - It starts with reading number of plants from the file.
   - Then it reads next 'n' number of line in a loop('n' is number of plants).
   - The loop stores and creates new plant from the details received from file.
   - At last the function returns an integer which is the last element read from the file(if the file input is corrected/in expected pattern
   - The integer returned is number of days simulation should happen

   EXCEPTION THROWN: FileNotFoundException, InvalidInputException

2. -simulationFunc(plants: ArrayList<Plant>, simulationDaysCount: int)
   - This function takes the integer value returned from read method and takes the plants
   - The function checks if the simulationDaysCount is more than 0
   - Then it proceeds with Radiation array and sets first radiation day as NoRadiation(as per task)
   - Proceeding in a loop going for simulationDaysCount times where first the plant gets affected by radiation(using changeNutrient method(Radiation))
   - The next day's radiation is decided in this function after comparing total demand of alpha and delta radiation required by all plants together.

   EXCEPTION THROWN: InvalidInputException

3. -printResult(plants: ArrayList<Plant>)
   - Simply prints the number of plants alive after the simulation is done

   EXCEPTION THROW: None

4. +simulate()
   - All the other function in the planet class are private
   - So, this function calls all the other function in appropriate order
   - Leading to the result of the program

   EXCEPTION THROWN: FileNotFoundException, InvalidInputException

Testing

Grey box test cases:

Outer loop (Summation)

1. length-based:
   - zero plant (inp.txt)
   - one plant(inp1p.txt
   - more plants(inp5.txt)
2. first and last:
   - first plants survives or not after facing the radiation for n days
   - last plants survives or not after facing the radiation for n days

Inner loop (Summation)

1. length-based:
   - one plant under radiation for zero-day(inp3.txt)
   - one plant under radiation for one day(inp2.txt)
   - one plant under radiation for more than one day(inp1p2.txt)
   - more plant under radiation for more than one day(inp4.txt)
2. first and last:
   - first radiation of getting influenced properly depending on the demand of radiation by the species of the plant
   - last radiation of getting influenced properly depending on the demand of radiation by the species of the plant
   -

Examination of function simulate()

Different cases checking each radiation of n days.

Some cases for checking if the plant is alive.