The assignment is made by **Violetta Firyaridi**

Faculty of Informatics, Computer Science

Neptun code: CHWMXR

The description of the exercise

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: a. alpha, if the need for alpha radiation is 3 or more greater than for the delta radiation b. delta, if the need for delta radiation is 3 or more greater than for the alpha radiation c. no radiation, otherwise There is no radiation on the first day…

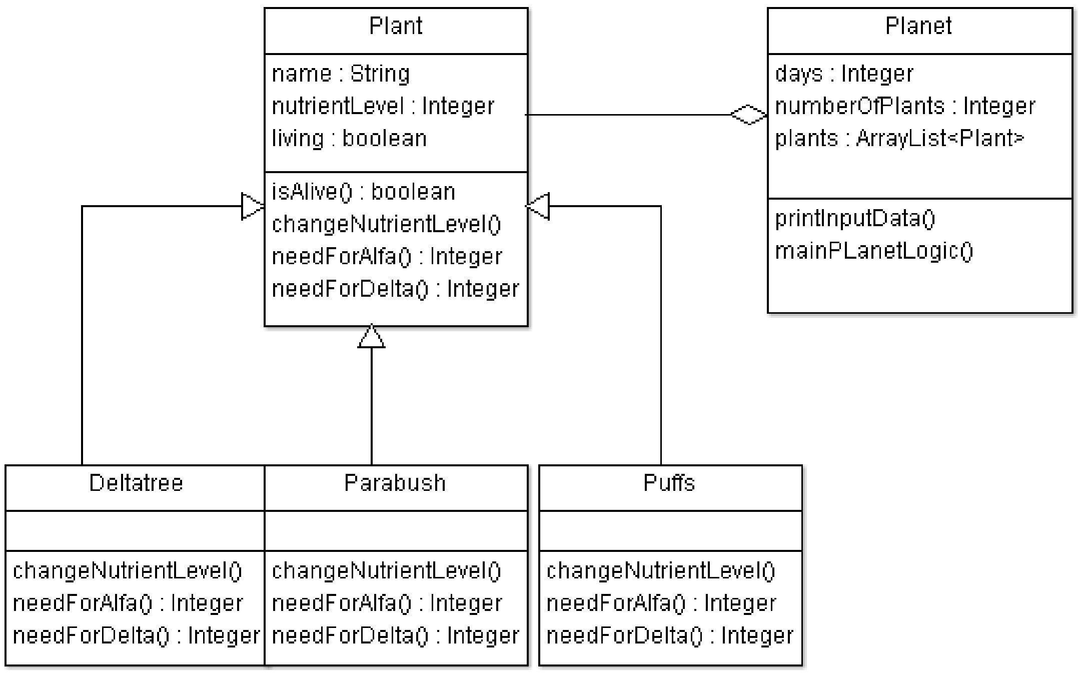
Simulate the behaviors of the plants, and print out the radiation of the day and the properties of the plants on each day.

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
	alfa	delta	no radiation	alfa	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).

The class diagram



The short description of each method

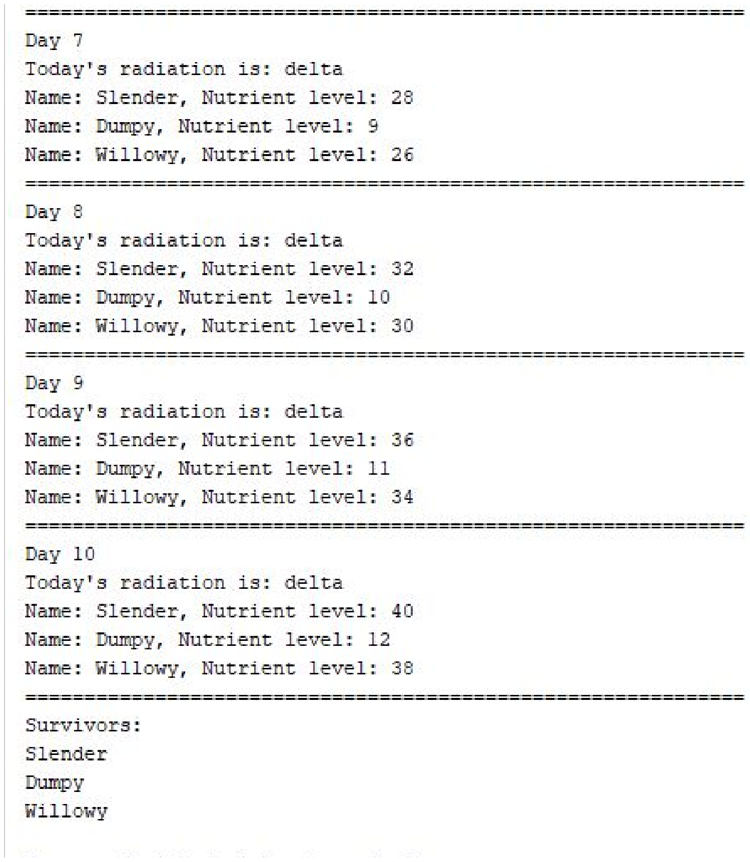
Method **readFileInfo** parse the data from the file and writes into class instance Planet.
Every class contains method **changeNutrientLevel** that in accordance with the type of radiation of the day and the type of the plant, changes nutrient level according to the task.
Class Plant has a method called **isAlive**, that check whether the plant runs out of the nutrients or not.
Method **mainPlanetLogic** contains the simulation of each day for every plant. The algorithm is like this: we are going through each day, for each plant we change its nutrient level.
Methods **needForAlfa** and **needForDelta** changes the need for the radiation of the next day according to the type of the plant and the case of the task.

User guide

In order to run the program you need to input the path to the *.txt file in console. The format of the file is: 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.

Then every step of the simulation will be written in a console.

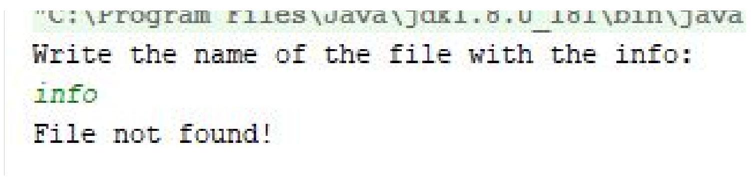
Example of the output of the last 4 days and result:



```
============================================================
Day 7
Today's radiation is: delta
Name: Slender, Nutrient level: 28
Name: Dumpy, Nutrient level: 9
Name: Willowy, Nutrient level: 26
============================================================
Day 8
Today's radiation is: delta
Name: Slender, Nutrient level: 32
Name: Dumpy, Nutrient level: 10
Name: Willowy, Nutrient level: 30
============================================================
Day 9
Today's radiation is: delta
Name: Slender, Nutrient level: 36
Name: Dumpy, Nutrient level: 11
Name: Willowy, Nutrient level: 34
============================================================
Day 10
Today's radiation is: delta
Name: Slender, Nutrient level: 40
Name: Dumpy, Nutrient level: 12
Name: Willowy, Nutrient level: 38
============================================================
Survivors:
Slender
Dumpy
Willowy
```

Testing

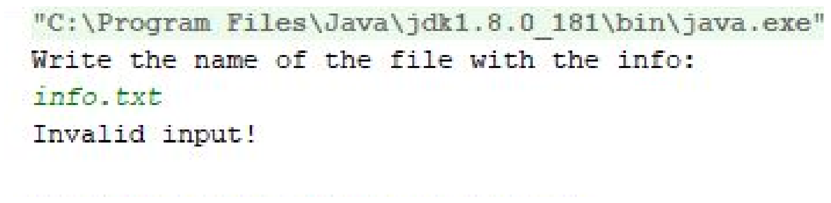
The first and the most obvious problem can occur in case of the wrong path to the file. To prevent this FileNotFoundException is used



```
"C:\Program Files\Java\jdk1.8.0_181\bin\java
Write the name of the file with the info:
info
File not found!
```

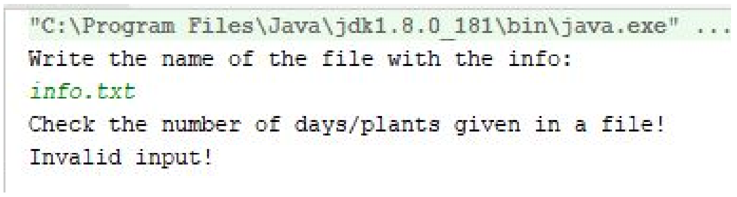
The second problem is having invalid data inside the *.txt file. For instance,

- the number of days is negative or zero
- the number of plants is negative or zero
- the nutrient level of the given plant is negative or zero
- there is no name of the plant given
- invalid category of the plant



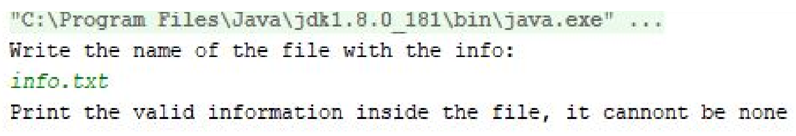
```
"C:\Program Files\Java\jdk1.8.0_181\bin\java.exe"
Write the name of the file with the info:
info.txt
Invalid input!
```

The next problem can occur when the number of plants given is not the same as the described number of plants in a file.



```
"C:\Program Files\Java\jdk1.8.0_181\bin\java.exe" ...
Write the name of the file with the info:
info.txt
Check the number of days/plants given in a file!
Invalid input!
```

The NullPointerException is used to catch the occurrence, when there is no information inside the file.



```
"C:\Program Files\Java\jdk1.8.0_181\bin\java.exe" ...
Write the name of the file with the info:
info.txt
Print the valid information inside the file, it cannont be none
```

The IOexception is used in order to catch exception when the connection with the file was lost.