Question 1 [4 points]

Implement a new ***Desqueing Strategy*** called **`LessContaminationStrategy`** that gives priority to less contaminating vehicles (according to their contamination class) as follows. This strategy has a parameter **`"limit"`** that indicates the maximum number of vehicles that can be advanced in a tick, and it advances the first **`"limit"`** less contaminating vehicles, respecting the order in the queue when two vehicles have the same contamination class. For example, if in the queue we have the vehicles **`[v1,v2,v3,v4,v5]`** (in this order) with corresponding contamination classes **`[7,2,3,1,2]`** and **`"limit"`** is **`2`**, then **`[v4,v2]`** will advance (in this order). Note that if there are less than **`"limit"`** vehicles, then obviously they will all advance but the order is important, for example, if **`"limit"`** is **`10`** in the above example then **`[v4,v2,v5,v3,v1]`** will advance (in this order).

The **`JSON`** input corresponding to this strategy has the following syntax:

```
{
  "type" : "less_cont_dqs",
  "data" : { "limit" : 1 }
}
```

where the value of **`"limit"`** is a positive integer, and it is optional with default value 1.

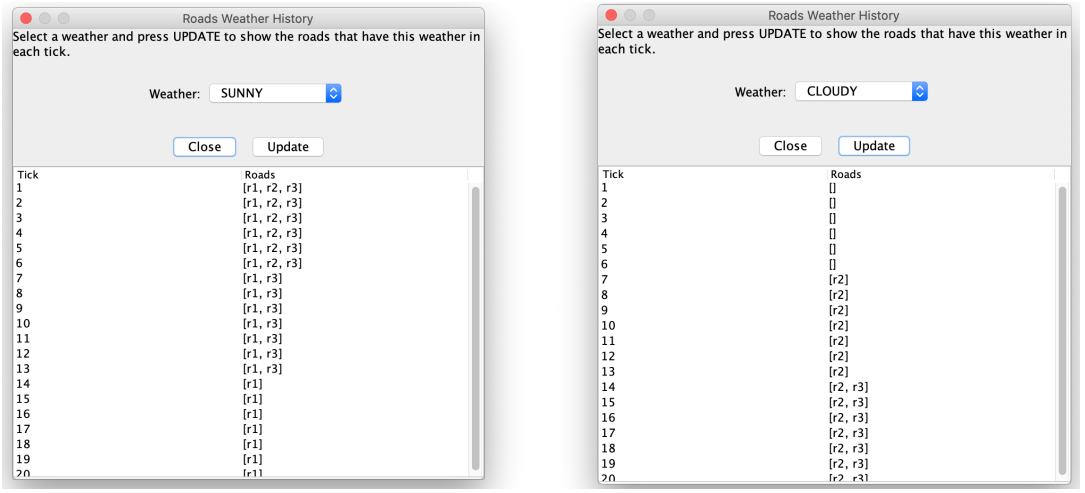
Recall that the sorting algorithms of **`Java`** guarantee that equal elements will not be reordered as a result of the sort.

Together with this **`PDF`**, you have a **`JSON`** file that you can use to test your implementation (it corresponds to the example above).

In your solution, you can add new classes but you are not allowed to modify existing ones except the `Main` class. A solution that does not respect this requirement will be given 0 points.

Question 2 [6 points]

Add a new button to the **`ControlPanel`**, such that when clicked it opens a Dialog with the following aspect:



where the user can select a **Weather**, and when **`UPDATE`** is clicked it shows in each row of the table: (1) a tick, i.e., a time; and (2) the identifiers of roads that had this weather at that time (at the end of the corresponding simulation step). For example, suppose we have executed the simulation for `20` ticks (using the **`JSON`** file provided with this question):

1. If we select **`SUNNY`** and click on **`UPDATE`**, the table will be as the one shown on the left, which indicates that: during the first `6` ticks roads `r1`, `r2` and `r3` had **`SUNNY`** weather; from tick `7` to `13`, only roads `r1` and `r3` had **`SUNNY`** weather; and from tick `14` to `20`, `r1` was the only road with **`SUNNY`** weather.
2. If we select **`CLOUDY`** and click on **`UPDATE`**, the table will be the one the right.

When the dialog window opens, the table can be empty or already filled with information for the selected weather (it depends much on how you decided to implement this question, do not worry about this, both solutions are valid).

The new button in the **`ControlPanel`** should be disabled during the simulation and enabled again when the simulation is over or stopped. Together with this **`PDF`**, you have an icon that should be used for the new button, and a **`JSON`** file that to try your implementation (it corresponds to the tables above).

- **When you implement this question, think well of the information you need to store in order to generate the table. Be careful, as if you do not think of this well you will end up showing the information of the last tick in all rows.**
- **You are not allowed to modify or add any class the model, you can only modify or add classes to the view. A solution that does not respect this requirement will be given 0 points.**