Assignment 1
DEV 2 - Year 2015-2016

The Dev TEAM

September 28, 2016

1 Goal and description

The goal is to improve your design and implementation skills on data structures. For this purpose we created an *incomplete* simulation of a city, where only the city and it's roads are rendered. **Your task is to design and implement cars that are able to move randomly through the city**.

2 Software requirements

To work with the simulation you need `PyGame 3.4` and `Python 3.4`. You can download the `PyGame 3.4 x86` here. PyGame is a set of Python modules designed for writing games. The simulation comes with a *template project*. The template is available on N@school and GitHub under the voice `Assignment 1`.

3 Details

Classes As you will see in the template we have implemented some classes for you: a `Node` and a `Tile` data structure, accordingly available in `Node.py` and `Tile.py`. We recommend you to read them carefully and to understand their attributes.

The class `Tile` has a `Properties` attribute. Elements in `Properties` gives you information about the current node. For example a property could be `NotTraversable`, which means that this node is not traversable; or `Parking`, which means that this node is a parking place; etc. You can make you own properties if necessary.
NB. You need to study those structures and codes before you start with your implementation.

Game.py We also we provide you a main loop in `Game.py`. The `Main` function is the entry point of the game. Precisely the in the `Main` you find the a block of code which runs indefinitely the game. Within the block of code we call the functions `Update` and `Draw` to update the scene logic and display the scene elements accordingly.
Inside `Game.py` search for the function `Update`. Update takes as parameters the lists of cars to update and returns a new collection of cars (note a car might get filtered in case it enters a parking tile)

4 Tasks

Task 1 *Design* and the `Car` data structure that should at least provide the following attributes:

- A position, which references the node the car is in

Task 2 In `Update` *implement* the behavior of your cars.

- **Move** your cars randomly through the city (based on the current node of the car) and avoid non traversable nodes
- **Add** new cars after a condition is met. For example add a new car every 5 seconds (check `speed`).
- **Remove** a car from `car_list` if it enters a parking place

Task 3 Draw all cars. We provide you a function `Draw` that takes as input a list of cars. Use the hint we provide you inside the `Draw` function.

5 Submission and deadline

Contribution: *Groups of 2 students is allowed with individual responsibility*
What: *One PDF per student for all code + comments (comments: explain your code)*
When: *The Friday of week 6*
Where: *On N@school*

GOOD LUCK!!! The Dev TEAM ☺