VG101 — Introduction to Computer and Programming

Project 1

Manuel — UM-JI (Summer 2018)

- Include simple comments in the code
- If applicable, split the code over several functions
- Extensively test your code and impove it
- Write a single README file per assignment
- Zip all the files and upload the archive on Canvas

Ex. 1 — *Cross road traffic game*

At a cross road cars are either moving if the light is green or waiting if it is red. A road is composed of two lanes, for cars going in opposite directions while the light is located on the right hand side of the cars, before the cross. Moving cars are all going at a same constant speed of one unit every second.

A snapshot of the scenario at a time t is shown on Figure 1. On the picture three cars are going from top to bottom but are waiting as the traffic light is red. In the meantime five cars going from right to left are crossing, while two other cars are coming from the opposite direction. The two last cars can be seen coming from the bottom but have not yet arrived at the cross road.

Requirement for the above scenario:

- The user inputs n, the total number of cars, w the width of the crossroad, p the probability that a cars does not stop at the red light, and t the probability for a car to turn left of right;
- The number of cars on each line is randomly decided with respect to n;
- The color cycle of a traffic light is green for g s, orange for o s, and red for r s; The user sets the value of the variables g, o, and r;
- Lights in the diagonal of the cross road have a same color;
- Each time a light changes from green to orange the cars before the light stop and the ones which have already passed it finish crossing; e.g. on Fig. 1 if the green light becomes orange then the two cars in the middle of the cross should reach the other side before the red light turns green;
- Cars should appear at the beginning of the road with respect to where they come from; e.g. a car coming from the right should start from the right most edge of the figure;
- Every second all the non-waiting cars move by one unit;

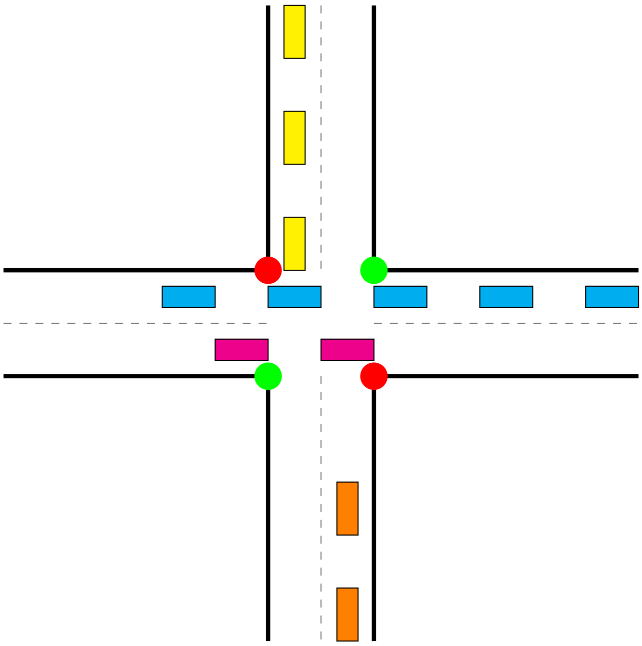

Figure 1: Traffic at a cross road.

The goal is to animate the displacement of the cars with respect to the traffic lights. The choice of the user will impact the risk for a car crash to happen. The animation stops as soon a crash occurs and the player loses. If all the cars have gone through without any problem the player wins.

1. Write a simple function that initialises all the parameters, i.e. prompts the user for the number of cars, the time for each light color, randomly generates how many cars go in each direction, etc..
2. Write a function that draws the cross road.
3. Write a clear and precise algorithm describing the motion of the cars as well as the change of color of the traffic lights. Justify your choices. The algorithm should describe the whole animation of the cross road.
4. Write several functions to plot the cars and animates the whole scenario of the game.

Each of the following questions is optional and will bring a bonus on the project grade if properly solved.

5. Design more realistic cars of various sizes.
6. Allow cars to adjust their speed within a range specified by the user. They can in turn use this information to prevent a crash.
7. Allow the player to choose a difficulty level and automatically adjust all the parameters accordingly, e.g. number of cars, width of the crossroad, right and left turns allowed, etc..

Note: other adjustments are welcome and encouraged, however first discuss them with the teaching team to check their validity and suitability for the project.

Ex. 2 — *Registration plates*

Each car features a registration plate composed one letter followed by a mix of five letters and digits, e.g. B $34HB0$. We want to generate a random plate for each of the car and return the plates of all the cars which do not stop at the cross road.

1. Random permutations.
    a) Write a recursive algorithm that takes as input a set of elements and returns all their permutations.
    b) Adjust the previous algorithm to return a random subset of all the permutations containing n elements.
    Hint: generating all the permutations and selecting n of them afterwards in not a good idea.
2. Implement the generation of the registration plates in MATLAB, and assign a unique plate to each car.
3. At the end of the game display the plate of all the cars that did not stop at the traffic light.