Design and solution description “The Boss”

Task description

The setting is present-day Latvia. Most people own a car or even multiple cars. Some cars are very nice, expensive and powerful. These cars need to be displayed on the streets as much as possible – nobody should think that the owners are miserly and cheap. Riga is the best place to display them because of the density of the viewers.

Nevertheless, there are downsides to this showing off – as it becomes ever harder to get around. In the early days, the drivers in Riga enjoyed their rapid pace, but now it has slowed down to a crawl. And no crisis would change this.

A car owner – affectionately named the Boss – also had to wait in the gridlock. He decided to restore the order in the traffic of Riga, in particular, via creating a system allowing him to drive faster than the others.

The Boss started with the minimum task – to ensure his travel in Riga. The Boss was convinced about his success since he had the needed connections and full support from the Riga City Council and the government. If a system could speed up the travel for certain high-ranking people, then he could count on their support.

It was decided that public services would gather the data about the traffic in Riga – and forward it to the car of the Boss – so that he can have computer-assisted route planning all the time. We expect that the Boss has excellent leadership capacity – and we do not care how he ensures the collaboration from the public services in the city.

In order to control the costs, the implementation was entrusted to the 2nd year students in the Faculty of Computing, University of Latvia. They had a reputation as competent professionals that do not demand outrageous money for their work yet.

The students had to do the initial analysis and design. The formal completion of the project was delegated to the company Insatiable Ltd - who were experts finding public grants for any noble cause.

Describe a more detailed set of requirements; the design with your analysis and explanations that will be passed to another programmer for coding. The map (actually, a simplified street plan) is read from an input file. The public services have divided the streets into segments stretching from one intersection to another. They can provide the following information – either by schedule or on-demand:

1. The maximum number of cars C_i (Cars), that can simultaneously drive through the street segment S_i (Segment) without experiencing any delay (a small number of cars ensures that there is no need to synchronize your car with the other cars; so there is nothing that can hinder your movement). The value $C_i=0$ means that the segment is currently closed; even the most important people cannot drive through it. The time required for the segment S_i is denoted by T_i (Time) – it is the minimum time when there are no obstacles.
2. The number of cars N_i (Number) currently driving through the segment S_i.
3. The maximum number of cars M_i (Maximum) that can be squeezed on the street segment S_i during the peak traffic jam.

4. The delay time D_i (Delay) that is additionally required on the segment S_i for every car that is above the number C_i. This lets us compute the actual time required for the segment S_i.

Assume that no time is spent in the intersections since it is already included in the values T_i and D_i.

The Boss has a car with a GPS system. It regularly sends the current coordinates to the central server. There is a program that can receive a few coordinate values, and it returns the segment S_i and the direction (the move from one intersection to the next one). This service does not need to be implemented, but can be used as needed.

At every moment, the Boss can tell us, what is his destination segment S_i. The segments are enumerated, and the central server returns the segment numbers in the order that would allow reaching the destination as early as possible. The street names will be added to these segment numbers by the final contractor, Insatiable Ltd. As the situation on the streets can change – either because of other cars or because of the road condition, the values C_i, T_i, D_i, N_i, can change at any time. The central server should be able to supply new values. The Boss can ask to recompute the chosen route at any time if he thinks that the speed is insufficient.

It was agreed that every student could define the remaining details depending on her/his interests and abilities. The main focus is not the design documentation, but the data structures and the algorithmic thinking.

Let us help the Boss and his team to save money when completing this worthy goal!

Suggestions for Making a Project Description

We recommend that the design or the description of a problem contains the following parts:

- Title Page. Specify the title of the project, your full name, Student ID, and the current year. Other essential information can be added.
- Requirements. List the requirements (remove all the redundant stuff from the original description) and make your assumptions explicit. Formalize all the important aspects of your problem.
- Input file. Define the structure of an input file and provide examples.
- Output file. Describe the structure of an output file and provide examples.
- Solution alternatives and theoretical justification. Sketch the ideas for all approaches solving the proposed problem known to you. Justify why they are valid. Mention the advantages and disadvantages of each alternative. List the paradigms of these algorithms when applicable.
- The selected approach. Define the solution alternative you have chosen. Describe the algorithm informally if this is not already done in the previous section. The key parts of the algorithm have to be described formally. Define the data structures used by your algorithm.

- Estimates of Complexity. Specify the complexity for the major parts of your solution and also the entire algorithm – both time and space complexity. Justify your estimates.
- Initial assessment of the implementation. Assess the possible implementation: What could be challenging for another programmer? How easy or how fast can it be done? What could be the hardware requirements for your program and its real-world response time?
- Test samples. Samples for your program: input and output files. This set can be implemented as a regular suite of acceptance tests. Provide hints on how to make relevant tests with large data volume and how to check that the code passes them.
- Conclusions. Conclusions and various remarks about this project.
- Bibliography. Specify different references and sources of inspiration that were helpful for this task, and what have you learned. References can include books, articles, Web resources, people you consulted, and so on.

You may also use a modified structure and add more sections describing your problem and its solution. There are no rigid standards. The abovementioned parts list the items that your instructor would like to see in one form or another.

The interface of your program (such as the data formats in input and output files) should be easy to use.

We recommend using charts, diagrams, and images to describe the data structures and the operation of your algorithm. This usually improves the readability of your description and also your grade.