# (Final) Homework - 4

**Q1)** Seeing how everybody is using graphs one way or another, you have decided to create a company ("BulutÇizge A.Ş.") that will provide graph oriented services to the public.

In more detail, clients will contact your server, send a graph object to it (+ some parameters), and ask for some computation with it, that your server will realize and return its result(s) to the client.

For some unknown reason you have chosen **java RMI** to implement the communication layer between the server and the clients.

Your server provides two types of services:
a) Given a graph, and two nodes A and B, it returns the shortest path between A and B.
b) Given a graph it returns the shortest Hamiltonian cycle (i.e. travelling salesman problem) (assume that the graph always contains such a cycle if this method is called)

What is required of you, are the following:
1) Design and implement the graph data structure, graph algorithms and nodes. Remember that they can represent anything, e.g. cities, people, etc. You want your service to work with any kind of data. (**30 points**)

2) The client, who has some data, will construct her graph using your interface, (so that it'll be compatible with your server), and call the corresponding method from the remote server, and wait for the output, in order to show it on screen (**15 points**) (a GUI is optional, **+20 points**).

3) The server must be waiting at all times for a new method invocation, and have the necessary graph processing methods ready to do the job. Whenever a new request comes it should print on screen which method is called at what time, and how long it took to complete it. (**30 points**)

4) Free trials are over. Now you are going pro, woohoo!! From now on, any client that wants to use your services must pay for it. In order to achieve this, assume that every client has an account on your server containing a credit amount that allows them to make a certain number of remote method calls. For instance client X has a credit of 100, meaning that she can make 100 remote calls (Does every method cost the same? That's up to you.). If her credit is zero, she'll receive an error. Make sure clients authenticate themselves with every remote method call, and decrease their credit accordingly. (**40 points**) (Optional: implement account creation and credit loading also through RMI, (for instance by providing their credit card information) **+20 points**). Make sure your server is thread-safe when handling credits, as tens (hundreds?) of remote method calls may happen at the same time.

5) Make sure you test both services (if possible, on different systems) and provide screenshots that they work.

6) Draw the class diagram of your solution. (**5 points**)

Have fun.

Note: assume your graphs are directed and weighted, and you are free to implement services (a) and (b) with any algorithm you like, except for brute force. It is up to you to model these graphs. Be careful when producing stubs and skeletons, as the exact procedure depends on your jdk version.