Project 1

The theme of this project is to implement the basic network design model that is presented in the lecture note entitled "An Application to Network Design," and experiment with it.

Specific Tasks:

1. Create a program that is capable of doing the following:

   - As input, it receives the number of nodes (N), the traffic demand values (b_{ij}) between pairs of nodes, and the unit cost values for the potential links (a_{ij}).
   - As output, the program generates a network topology (directed graph), with capacities assigned to the links (directed edges), according to the studied model, **using the shortest path based fast solution method** (see at the end of the referred lecture note). The program also computes the total cost of the designed network.

   Important notes:

   - Any programming language and operating system can be used, it is your choice.
   - For the shortest path algorithm you may download and utilize any existing software module from the Internet. If you use this opportunity, then include in your documentation a precise reference that tells where the module comes from.

2. Clearly explain how your program works. It is helpful to use flowcharts for visualizing the explanation.

3. Run your program on examples that are generated as explained below.

   - Let the number of nodes be $N = 21$ in each example.
   - For each example, generate the a_{ij}, b_{ij} values according to the rules described below. In these rules k is a parameter that will change in the experiments.

– For generating the b_{ij} values, take your 10-digit student ID, and repeat it 2 times, and append the first digit again at the end, to obtain a 21-digit number. For example, if the ID is 0123456789, then after repetition it becomes 012345678901234567890. Let $d_1, d_2, \ldots, d_{21}$ denote the individual digits in this 21-digit number. Then the value of b_{ij} is computed by the formula

$$b_{ij} = |d_i - d_j|.$$

For example, using the above sample ID, the value of $b_{3,7}$ will be $b_{3,7} = |d_3 - d_7| = |2 - 6| = 4$.

– For generating the a_{ij} values, do the following. For any given i, pick k random indices $j_1, j_2, \ldots, j_k$, all different from each other and also from i. Then set

$$a_{ij_1} = a_{ij_2} = \ldots = a_{ij_k} = 1,$$

and set $a_{ij} = 100$, whenever $j \neq j_1, \ldots, j_k$. Carry out this independently for every i.
Remark: The effect of this is that for every node i there will be k low cost links going out of the node, the others will have large cost. The shortest path algorithm will try to avoid the high cost links, if possible, so it effectively means that we limit the number of links that go out of the node, thus limiting the network density.

- Run your program with $k = 3, 4, 5 \ldots, 14$. For each run generate new random a_{ij} parameters independently.

4. Show graphically in diagrams the following:

- How does the total cost of the network depends on k?
- How does the density of the obtained network depends on k? Here the density is defined as the number of directed edges that are assigned nonzero capacity, divided by the total possible number of directed edges, which is $N(N-1)$.
- Show some of the obtained network topologies graphically. Specifically, draw three of them: one with $k = 3$, one with $k = 8$, and one with $k = 14$.

5. Structure of the program: your entire program should contain three well separated modules:

   - *Module 1:* generates the parameters of the random examples, These are passed on to Module 2.
   - *Module 2:* carries out the main algorithm (see Task 1.), and passes on the result to Module 3.
   - *Module 3:* creates the required presentation of the results (diagrams, figures, see Task 4.).

6. Provide a brief (1-2 paragraph) verbal justification that explains why the obtained diagrams look the way they do. In other words, try to convince a reader that what your diagrams show is indeed the "right" behavior, that is, your program that carries out the network design is likely correct.

7. Also include a section in the project document that is often referred to in a software package as "ReadMe file." The ReadMe file (or section) provides instructions on how to run the program.

Submission guidelines:

There will be a separate posting about submission guidelines and formatting requirements.