Introduction to Programming II
Assignment 1

Deadline: Week 8 (12.03.19 23:59)
Output: ZIP-file with the code has to be uploaded to Moodle, input and output files are NOT needed. Name of the file should be like this *NameSurname.zip* (For example, *IvanIvanov.zip*). No other symbols allowed
Programming language: C (C11 standard)
Requests:

- The program must work, the code should be readable, well-structured and should contain English comments
- It is allowed to use only standard C libraries
- NO extension of a deadline. Works sent after the deadline will NOT be evaluated
- Assignment is individual
- We will be using MOSS (Measure of Software Similarity) as a test for plagiarism. Be reminded that a score of 0 will be assigned to any submissions suspected of plagiarism pending a full investigation as per IU policies.

Evaluation criteria: 80% for the code correctness, 20% for readability of code and comments
Task:
Using Dijkstra's algorithm write a program that will find the shortest path(s) between cities. The number of cities can be in the range [5; 50]. The distance between cities has to be in the range [1;20]. Every city is assumed to be 0 distance from itself. Consider that some cities can be unachievable from other cities (distance between such cities should be marked with * symbol). The distance between cities is not guaranteed to be the same in both directions because of using a directed graph. Cities have to be numbered from 0 to *N*-1, where *N* is the number of cities.
Inputs: Input file called *input.txt* should have the next structure:

```
5 1 2

0 5 7 4 5
4 0 9 8 1
7 9 0 4 7
6 9 4 0 11
5 1 9 5 0
```

The first line should contain 3 numbers:

- The first number is the number of cities
- The second number is the initial city
- The third number is the destination city

The second line is empty. Third and all next lines represent a 2-dimensional square matrix of distances between cities. Initial city is defined by columns and destination city is defined by rows. Each number in each string is separated by the only space, NOT tabulation or anything else. Last symbol of correct file is the last number of the matrix. Only Unix style line separators are allowed. Insignificant zeros in the beginning of numbers are not allowed

Outputs: Output file called like *NameSurnameOutput.txt* (For example, *IvanIvanovOutput.txt*) should have the next structure:

The shortest path is 8.
The number of shortest paths is 1:
1. 1 -> 4 -> 2

Only Unix style line separators are allowed.
Errors: Some of the input tests can contain mistakes. Your algorithm should detect them and print one error message in the output file instead of normal output. Even if there are more than 1 error in the input, output has to contain message only for the first found error. There is no priority for errors, the first found error has to be in output file. Possible error messages:

1. Number of cities is out of range
2. Chosen initial city does not exist
3. Chosen destination city does not exist
4. Matrix size does not suit to the number of cities
5. The distance between some cities is out of range
6. Initial and destination cities are not connected
7. Structure of the input is invalid // in case of other problems

Tests:

1. **Test 1**
   Input:
   5 3 4

   0 5 7 4 5
   5 0 9 8 1
   7 9 0 4 7
   4 8 4 0 11
   5 1 7 11 0

   Output:
   The shortest path is 9.
   Number of shortest paths is 2:
   1. 3 -> 0 -> 4
   2. 3 -> 1 -> 4

2. **Test 2**
   Input:
   6 6 4   // max is 5 because the first city is 0

   0 5 7 4 5 3
   5 0 9 8 1 3
   7 9 0 4 7 6
   4 8 4 0 11 4
   5 1 7 11 0 9
   3 3 6 4 9 0

   Output: *Chosen initial city does not exist*

3. **Test 3**
   Input:
   4 3 0   // min is 5 according to assignment constraints

   0 5 7 4
   5 0 9 8
   7 9 0 4
   4 8 4 0

   Output: *Number of cities is out of range*

4. **Test 4**
   Input:
   5 3 4

   0 5 7 4 5 4   // excess symbol, has to be a new line
   5 0 9 8 1 3
   7 9 0 4 7 6
   4 8 4 0 11 4
   5 1 7 11 0 9
   3 3 6 4 9 0

   Output: *Matrix size does not suit to the number of cities*

5. **Test 5**

   Input:

   6 5 4

   0 5 7 4 5 3
   5 0 9 8 1 31   *// max is 20 according to assignment constraints*
   7 9 0 4 7 6
   4 8 4 0 11 4
   5 1 7 11 0 9
   3 3 6 4 9 0

   Output: *The distance between some cities is out of range*

6. **Test 6**

   Input:

   5 4 1

   *0 5 7 4 **
   *5 0 9 8 **
   *7 9 0 4 **
   *4 8 4 0 **
   ** * * * 0*

   Output: *Initial and destination cities are not connected*

7. **Test 7**

   Input:

   s;dlkfmk   *// has to be a number*
   5 t 3

   Output: *Structure of the input is invalid*