Assignment 3

CmpE 250, Data Structures and Algorithms, Fall 2017

Instructors: H. L. Akın, A. T. Cemgil
TA: Alper Alimoğlu, Özlem Şimşek
SA: Yusuf Hakan Kalaycı, Abdullatif Köksal

Due: 7 Dec. 2017, 23:59 Sharp

1 Statement

Ali Baba has become a trader - head of caravan with large amount of camels. Currently he is returning to his hometown Tehran. The way home leads through a land full of thieves. Fortunately the residents of the towns, forced to fight the thieves for centuries, have mastered preparing special coins to persuade the thieves to avoid robbery. They are now capable of avoiding the thieves via coins. Whenever a group of thieves see their favourite coin, they can let Ali Baba pass.

The land of Middle East is quite confusing: there are many towns, and many roads connect them. These roads do not cross outside the towns. Ali Baba has gathered all practical information about the land. He knows what kind of thieves he may come across on each of the roads and how much time he needs to walk it down. He also knows in which towns there are jeweler and what kinds of thieves can be avoided with the coins that they have.

But Ali Baba does not have any special coins to avoid thieves at the moment. He wants to get back to Tehran as soon as possible. As a trader he is quite ashamed that he does not know the best route, and that he has no special coins. Help him find the shortest path to Tehran such that whenever he could meet some group of thieves, previously he would have a chance to get an appropriate coin to avoid the group of thieves. Notice taht he can store unlimited number of special coins.

2 Input/Output Format

2.1 Input Format

The first line of the input file holds four integers: n, m, p and k, separated by single spaces.

1. n: the number of towns ($1 \leq n \leq 2,000$)
2. m: the number of roads connecting the towns ($0 \leq m \leq 3,000$)
3. p: the number of different kinds of thief groups ($1 \leq p \leq 13$)
4. k: the number of jewelers who have some special coins ($0 \leq k \leq n$)

The towns are numbered from 1 to n in such a way that n is Tehran's number and 1 is the number of the town which Ali Baba starts in. The different kinds of thief groups are numbered from 1 to p.

In the following k lines, the profiles of successive jewelers are given, one per line. The $(i+1)$-st line holds the integers w_i, q_i, $r_{i,1} < r_{i,2} < ... < r_{i,q_i}$, separated by single spaces. Note that a town may have more than one jeweler.

1. w_i: the town id in which the jeweler lives ($1 \leq w_i \leq n$)
2. q_i: the number of different kinds of coins that the jeweler can provide ($1 \leq q_i \leq p$)
3. $r_{i,1} < r_{i,2} < ... < r_{i,q_i}$: the kinds of coins themselves in increasing order ($1 \leq r_{i,j} \leq p$)

The coin id corresponds the thief group id which it is produced for. For example, Ali Baba needs to have coin type 3 in order to pass by thief group 3.

After that, there are m lines with road descriptions. The $(k+i+1)$-th line holds the integers v_i, w_i, t_i, s_i, $u_{i,1} < u_{i,2} < ... < u_{i,s_i}$, separated by single spaces. No two roads connect the same pair of towns.

1. v_i: the town in the beginning of the road ($1 \leq v_i < w_i \leq n$)
2. w_i: the town in the end of the road ($1 \leq v_i < w_i \leq n$)
3. t_i: the time needed to walk down the road (same in both directions) ($1 \leq t_i \leq 500$)

4. s_i: the number of different kinds of group of thieves that may appear on that road ($0 \leq s_i \leq p$)

5. $u_{i,1} < u_{i,2} < ... < u_{i,s_i}$: the kinds of thieves groups themselves in increasing order ($1 \leq u_{i,j} \leq p$)

2.2 Output Format

Your program should print out the path to the output file. The path is a sequence of town ids in order of travel from starting town (1) to Tehran (n). The optimum path takes the minimum transfer time to reach Tehran.

If reaching Tehran is impossible, the output should be -1.

3 Examples

For the input data:

```
6 7 4 2
2 1 2
3 2 1 3
1 2 2 0
2 3 9 0
1 4 2 1 2
2 5 3 0
4 5 5 2 2 3
4 6 18 0
5 6 3 2 1 2
```

The correct result is:
1 2 1 4 6

The minimum transfer time is 24 and there is no any better path then this one. For better understanding you may see the Figure 1.

And for the following data set:

```
2 1 1 1
2 1 1
1 2 1 1 1
```

the correct result is: -1

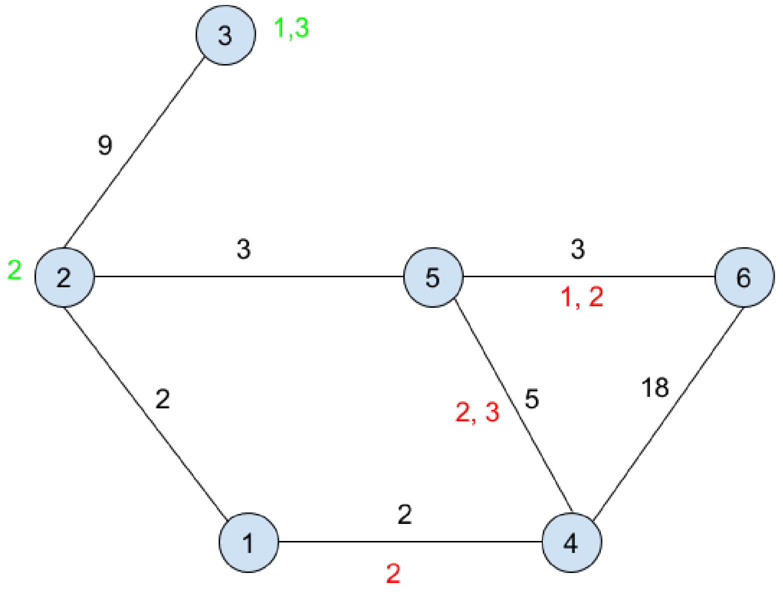

Figure 1: Visualization of map of Middle East. Black values represent the time needed to walk down that road, red values represent the kinds of thieves that may appear on that road and green values represent the kinds of thieves against which that jeweler's coins can avoid.

4 Grading

Grading of this project is based on three cases for any test case.

1. If the printed path is a valid path that provide Ali Baba going from start town to end town without any robbery, you will get 50% points of that test case.

2. If the printed path is an optimum path with minimum transfer time then you will get 100% points of that test case.

3. If there is no valid path then you must print -1 in order to get point from that test case.

Your score will be the sum of collected points from each test case. Maximum score can be up to 100.

5 Implementation Details

1. Your program will be compiled with **cmake CMakeLists.txt && make** command. Therefore, if you add new files, you have to check CMakeLists.txt is updated accordingly so that your code auto-compiles. Note that it is also fine to code in a single file in this project (Because I am assuming you already know what .h and .cpp files are and how to use those)

2. I will execute your program with **./project3 inputFile outputFile** command. So, use command line arguments in your main function accordingly.

6 Project Guidelines

1. Warning: All source codes are checked automatically for similarity with other submissions and also the submissions from previous years. Make sure you write and submit your own code.

2. Your program will be graded based the correctness of your output and the clarity of the source code. Correctness of your output will be tested automatically so make sure you stick with the format described above.

3. There are several issues that makes a code piece 'quality'. In our case, you are expected to use C++ as powerful, steady and flexible as possible. Use mechanisms that affects these issues positively.

4. Make sure you document your code with necessary inline comments, and use meaningful variable names. Do not over-comment, or make your variable names unnecessarily long.

5. Try to write as efficient (both in terms of space and time) as possible. Informally speaking, try to make sure that your program completes in meaningful amount of time.