HACETTEPE UNIVERSITY
DEPARTMENT OF COMPUTER ENGINEERING
BBM204 SOFTWARE LAB. II
ASSIGNMENT #5

Subject: Shortest path.
Submission Date: April 28,2016
Deadline: May 12,2016
Programming Language: Java
Advisors: TAs.Levent Karacan

1 Constrained Shortest Path

Shortest path problem is defined as to find shortest path between a starter node and target node in a graph data structure and plays an important role for many practical applications for example, network routing to send internet packets through shortest path between routers, chip design to produce most effective circuit, shipping to transmit shipment in a shortest time and et cetera. There are various shortest path algorithms, such as Breadth First Search, Dijkstra and A^*, which find the least cost path to reach a target node from starter node. However, in the real problems, it is possible to encounter different constraints while finding shortest path. Therefore, this shortest path algorithms must be modified according to the given constraint. Suppose a navigation system which offers shortest or least cost path between starter and target locations defined by a user. In this case, we can use standard shortest path algorithms, however, if we want to pass certain paths or locations, we must make some changes on algorithms or develop a new strategy to provide the given constraints.

In this experiment, you are expected to develop your strategy for such a constraint to find shortest path by providing it. You can modify any algorithm, develop your strategy or even build your own algorithm. All details are given below.

2 Problem

Given a weighted graph representing locations as nodes and roads as edges with related distances and *mustpass* nodes as constraints , you will develop or search a solution to find shortest path by providing your shortest path must contain given *mustpass* nodes.

Input: A map with locations and roads which are described by a text file with starter and destination locations and *mustpass* label.

Output: Shortest path providing given constraint

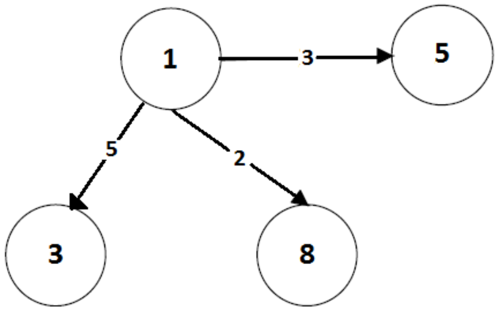
You are given an input file in the following format:

S:5,D:13
1. 5(3),8(2),3(5)
2. 3(10),4(4) mustpass
3. 6(2),3(5)

⋮

N. . . . ,. . . ,. . .

,where first line indicates ID numbers of starter and destination vertices and the following lines specify graph nodes with ID numbers(1,2,3 . . . N). After the ID number for each line , connected locations(5(3) 8(2) 3(5)) are indicated with distances in the brackets. For example the first line draws a graph structure as:

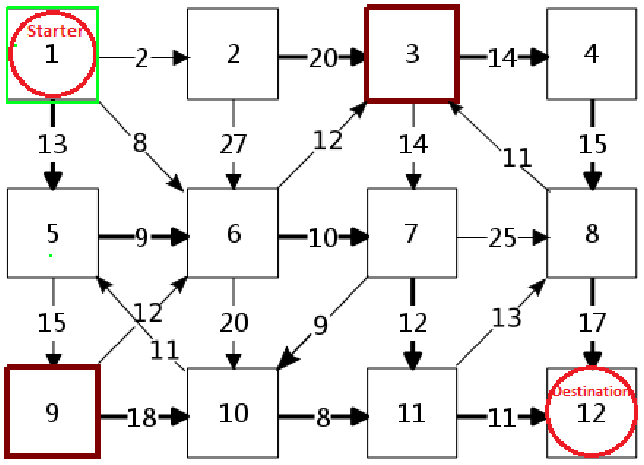


If there is a *mustpass* label for a graph node as in the second line given example above, shortest path must contain second node. For better understanding, consider the graph given in Figure 1-a with 12 locations connected to each other with roads indicated by arrows with distances. Red circles indicate starter and target nodes and red squares indicates *mustpass* nodes. If there were not any constrained, shortest path would be like in Figure 1-b and if we provide given mustpass constrained, shortest will be like in Figure 1-c.

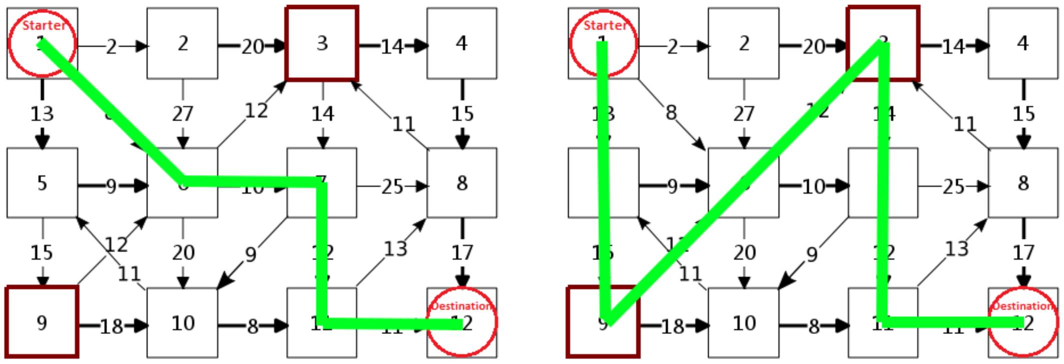
You can develop your approach or modify existing methods to find shortest path for a given such a graph structure. You must shortly explain your idea in README.txt(maximum a half page) file. Your are also free to use idea of someone else, but you must code the idea yourself and make an explanation of the solution and refer where you find the solution in a README.txt file with your own sentences(maximum a half page). If you violate this, you will be punished according to department discipline rules. In short, **you can not use the code of someone else** but idea provided that you explain the solution and refer the sources of which you make use. Moreover, if your explanation is not enough to give the idea, your grades will be lowered according to your explanation. (You can give me examples).

3 Steps

- Read vertices from input file
- Create a graph structure
- Implement a function takes starter and destination vertices, list of *mustpass* nodes and outputs shortest path



(a) Example directed graph with 12 vertices



(b) Shortest path

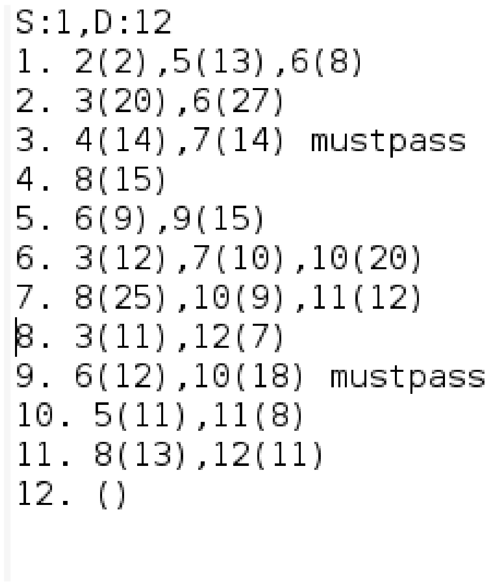
(c) Shortest path with constraint

Figure 1: Illustration of shortest paths with or without mustpass nodes.

- Print shortest path to a output file with distance. (Look at example output file)

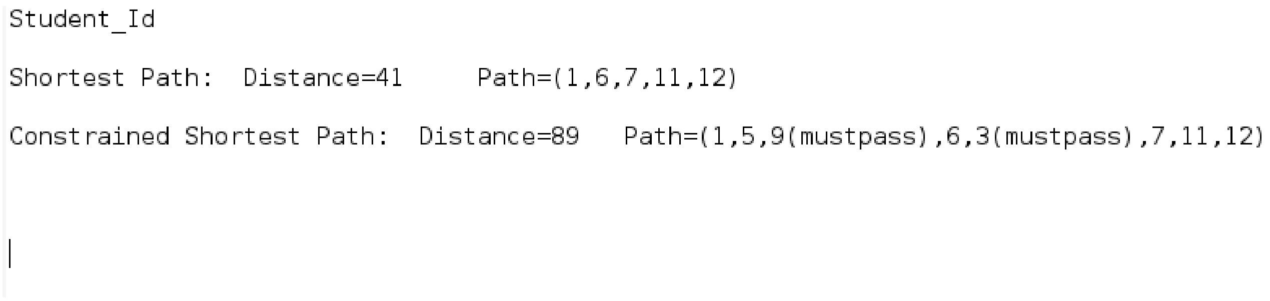
4 Input and Output

Input file format will be as in Figure 4 and your program must print results to output file as shown in Figure 5 for such a graph illustrated in Figure 1. You will be given example input and output files.



```
S:1,D:12
1. 2(2),5(13),6(8)
2. 3(20),6(27)
3. 4(14),7(14) mustpass
4. 8(15)
5. 6(9),9(15)
6. 3(12),7(10),10(20)
7. 8(25),10(9),11(12)
8. 3(11),12(7)
9. 6(12),10(18) mustpass
10. 5(11),11(8)
11. 8(13),12(11)
12. ()
```

Figure 2: input.txt



```
Student_Id

Shortest Path:  Distance=41     Path=(1,6,7,11,12)

Constrained Shortest Path:  Distance=89   Path=(1,5,9(mustpass),6,3(mustpass),7,11,12)
```

Figure 3: output.txt

Notes

Your experiments will be executed in Dev machine, please make sure whether it works properly on Dev Machine before or not. You should use comment lines to explain your code. Save all your work until the assignment is graded.You can ask your questions about the experiment on Piazza.

You have to give your experiment files in the given format below;
Input file must be taken as argument from command line. (java Main input.txt output.txt)

```
[Student id]
        *.java
        Main.java
        [README.txt]
```

Your assignment will not be evaluated If you do not build a valid code.

Policy

All work on assignments must be done individually unless stated otherwise. You are encouraged to discuss with your classmates about the given assignments, but these discussions should be carried out in an abstract way. That is, discussions related to a particular solution to a specific problem (either in actual code or in the pseudo code) will not be tolerated. In short, turning in someone else's work(from internet), in whole or in part, as your own will be considered as a violation of academic integrity. Please note that the former condition also holds for the material found on the web as everything on the web has been written by someone else.