National Yang Ming Chiao Tung University
Department of Electrical and Computer Engineering
Data Structures, Fall 2023

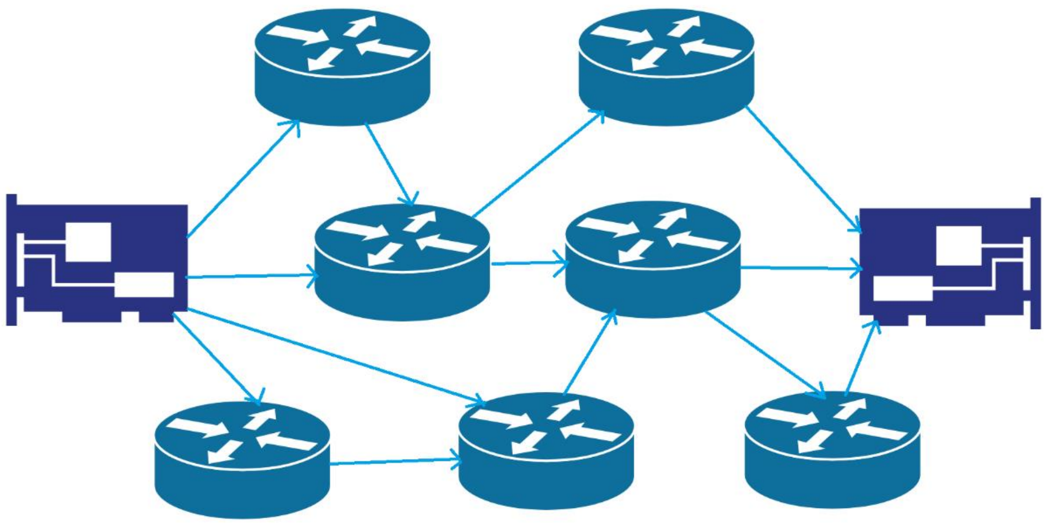
December 19, 2023
Instructor: H.-M. Chen
TA: S.-P. Wang

Programming Assignment #6

Graph & Shortest Path

1. Problem Description

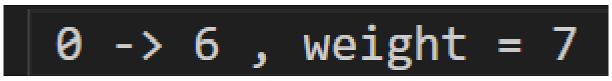
In this assignment, you're asked to solve a **Packet-Routing problem** between two PC. In order to achieve the highest package-tranfer speed, you have to evaluate the network between the source PC and the target PC, then determine the shortest routing path. Given a **Weighted Directed Graph**, determine the shortest path of the graph between the output ports of the source PC (only have output edges) and the input ports of the target PC (only have input edges). You are allowed to use **STL container** in this assignment.



2. Input Format

All input data comes from an **input file** addressed by **argv[1]**.

Each line in the input file represent an **direct route (edge)** between two connection points (node) in the network (graph). **Weight** represent the **timing cost** for this route). (Notice that there's a space character at the beginning of each line)
Ex.



```
 0 -> 6 , weight = 7
```

There will be at most **70 connection point**s in each input case.

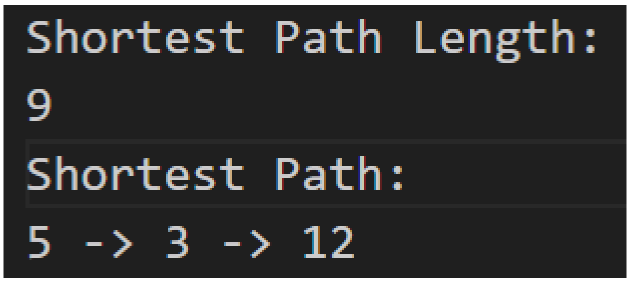
Notice that it is possible to have **inconsecutive naming index** of each node**.**
Which means that there may be missing some indexes (you can just ignore them since it means that they are not connected to this network)
The given graph **will not contain any cycles**.

3. Output Format

All output data should go to an **output file** addressed by **argv[2]**.

First you have to output the **shortest path length** of this network from an output port of the source PC to an input port of the target PC.
Then output the **shortest path** itself.
(You need to output only **one** shortest path.)

Ex.

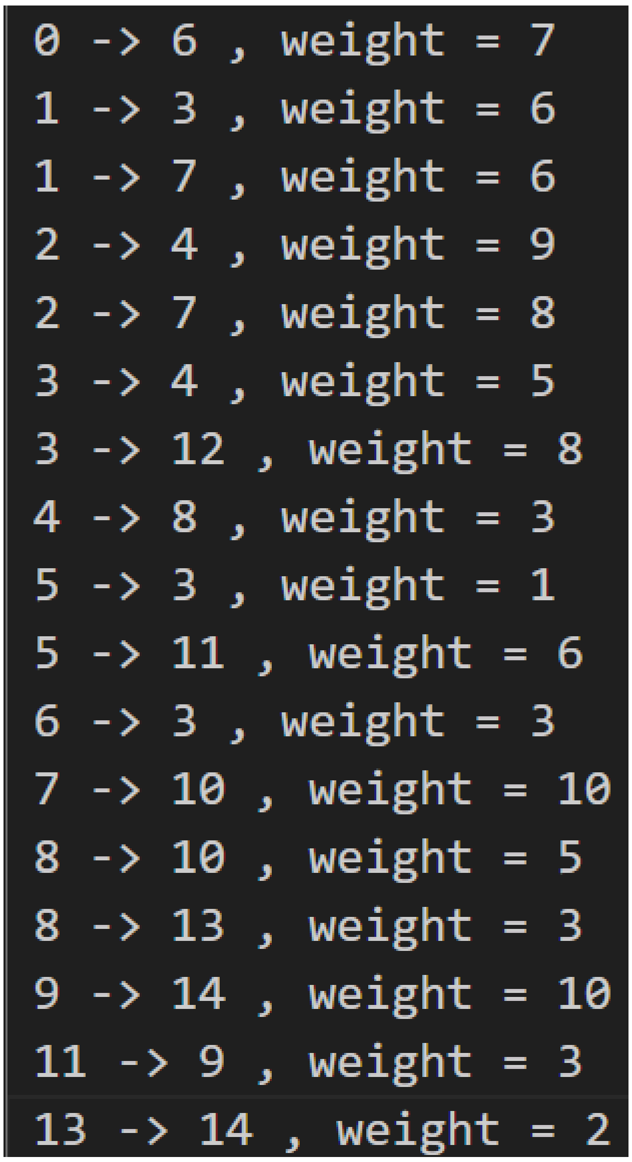


```
Shortest Path Length:
9
Shortest Path:
5 -> 3 -> 12
```

Notice that the shortest path must start from an **output node** (only have input edges), then ends with an **input node** (only have output edges).
The given input cases will only have **single output-input node pair** that has shortest path, however it is possible that the intermediate points may be different from the golden, as long as the path length is minimum, your answer would be viewed as correct by TA's judge program. (But please check your answer carefully).

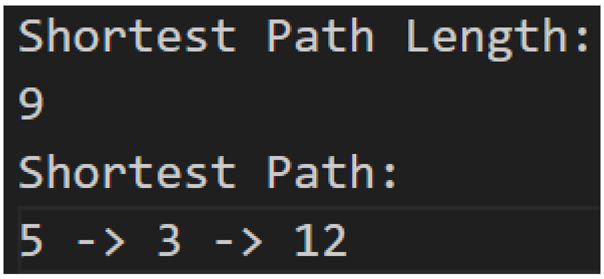
4. Sample Input / Output

Sample Input 1



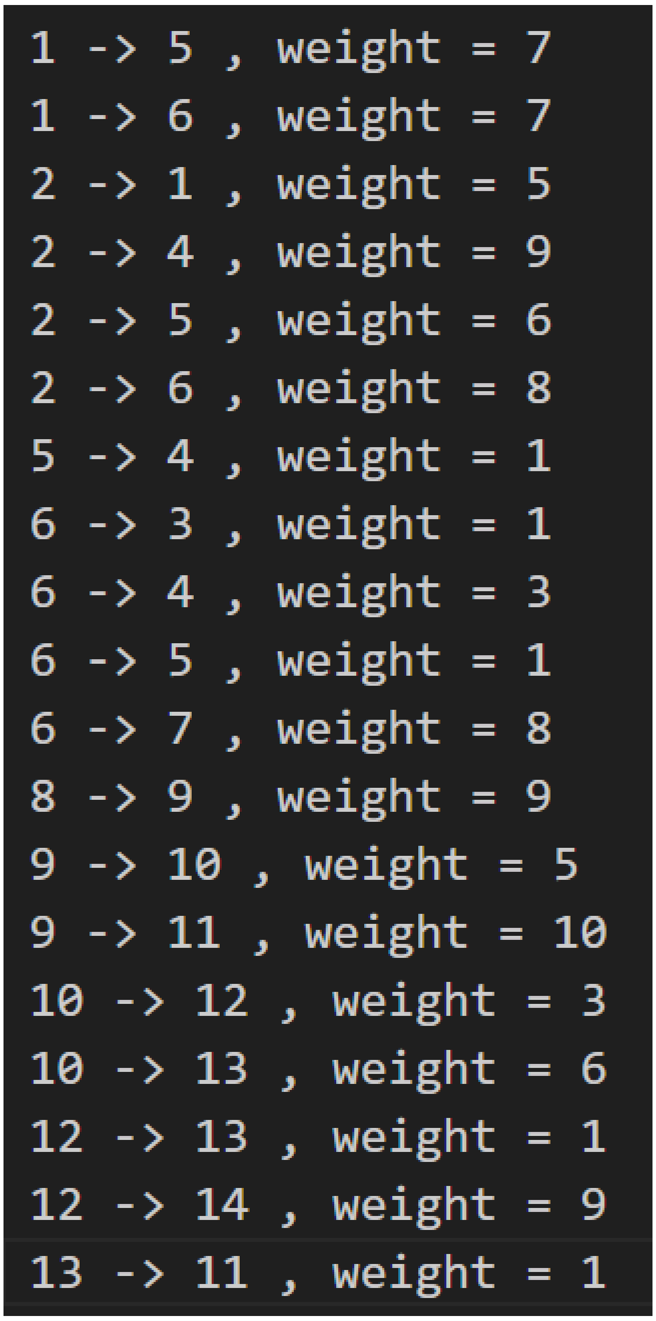
```
0 -> 6 , weight = 7
1 -> 3 , weight = 6
1 -> 7 , weight = 6
2 -> 4 , weight = 9
2 -> 7 , weight = 8
3 -> 4 , weight = 5
3 -> 12 , weight = 8
4 -> 8 , weight = 3
5 -> 3 , weight = 1
5 -> 11 , weight = 6
6 -> 3 , weight = 3
7 -> 10 , weight = 10
8 -> 10 , weight = 5
8 -> 13 , weight = 3
9 -> 14 , weight = 10
11 -> 9 , weight = 3
13 -> 14 , weight = 2
```

Sample Output 1



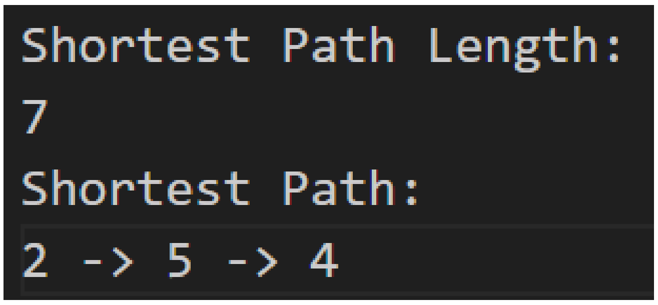
```
Shortest Path Length:
9
Shortest Path:
5 -> 3 -> 12
```

Sample Input 2



```
1 -> 5 , weight = 7
1 -> 6 , weight = 7
2 -> 1 , weight = 5
2 -> 4 , weight = 9
2 -> 5 , weight = 6
2 -> 6 , weight = 8
5 -> 4 , weight = 1
6 -> 3 , weight = 1
6 -> 4 , weight = 3
6 -> 5 , weight = 1
6 -> 7 , weight = 8
8 -> 9 , weight = 9
9 -> 10 , weight = 5
9 -> 11 , weight = 10
10 -> 12 , weight = 3
10 -> 13 , weight = 6
12 -> 13 , weight = 1
12 -> 14 , weight = 9
13 -> 11 , weight = 1
```

Sample Output 2

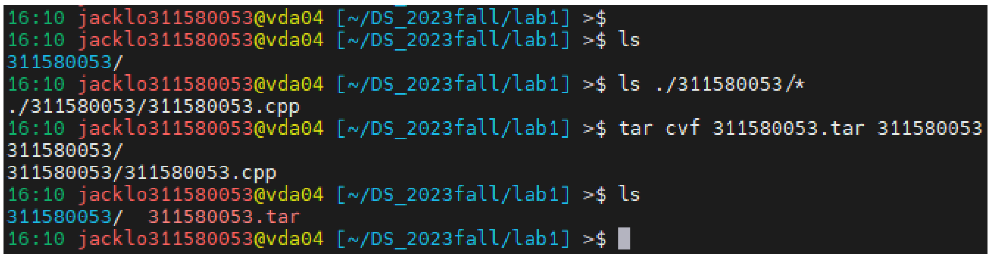


```
Shortest Path Length:
7
Shortest Path:
2 -> 5 -> 4
```

5. Submission Information

1. Your program must be written in C/C++ language and can be compiled on the Linux platform.
2. Please put the required files in a folder named with your Student_ID and the required files should also be named with your Student_ID (.cpp, .c).
3. To submit your program, please use the command below to compress the folder named with "[Student_ID].tar" in the Linux environment and upload it to E3.

   `tar cvf Student_ID.tar Student_ID`



```
16:10 jacklo311580053@vda04 [~/DS_2023fall/lab1] >$
16:10 jacklo311580053@vda04 [~/DS_2023fall/lab1] >$ ls
311580053/
16:10 jacklo311580053@vda04 [~/DS_2023fall/lab1] >$ ls ./311580053/*
./311580053/311580053.cpp
16:10 jacklo311580053@vda04 [~/DS_2023fall/lab1] >$ tar cvf 311580053.tar 311580053
311580053/
311580053/311580053.cpp
16:10 jacklo311580053@vda04 [~/DS_2023fall/lab1] >$ ls
311580053/  311580053.tar
16:10 jacklo311580053@vda04 [~/DS_2023fall/lab1] >$ 
```

File hierarchy example:

```
311555008.tar
│ 311555008              # folder
│ │ 311555008.cpp        # source file
```

6. Due Date

Be sure to upload the tar file by "January 7, 2024". There will be a 10% penalty per day for the first four days (weekend included) and will not be accepted afterwards.

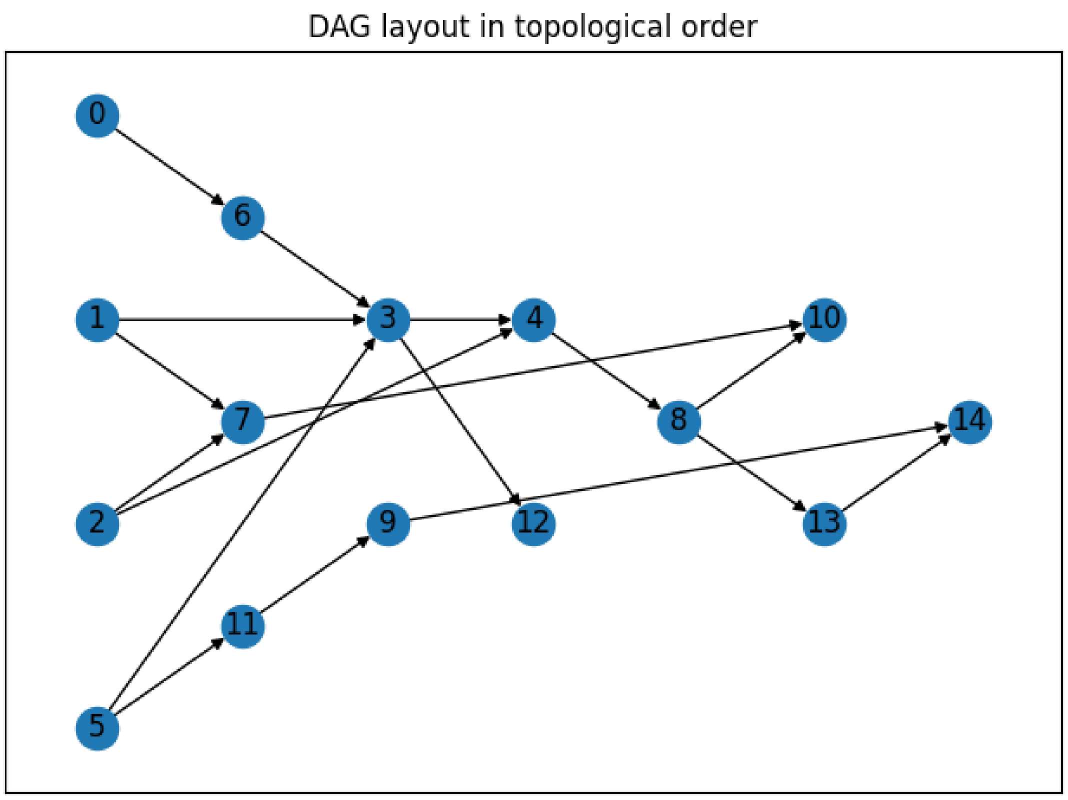
7. Grading Policy

The programming assignment will be graded based on the following rules:

- Pass the open cases with compilable source code (60%)
- Pass the hidden cases with compilable source code (40%)
- -10% Output more than one shortest path
- -10% of your total score if any file occurs naming error or not compress, or include irrelevant files or folder in the .zip file
- -10% for typo
- -50% for didn't use file I/O (ex. Use cin or cout for I/O)
- ***No credits for plagiarism or hard-coded program***

8. Hint

You can refer to the .png files given along with the open cases, remember to take **weights** into consideration when you're refering to those images.
Also, you have to determine the input nodes and output nodes according to the **definition specified above**, not by the positions of them in the image.



Contact

王淞平 tommy25582143.cs11@nycu.edu.tw