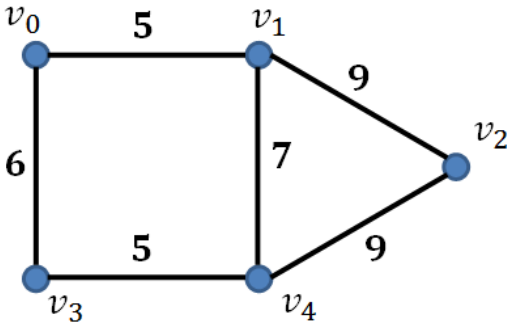
Homework #8
Programming Assignment

(1) Implement a program that finds a shortest path from a simple weighted graph in Python. Submit the python code file named '2015XXXX_yourname.py'. (20 points)

(See "Dijkstra's algorithm" in the textbook)

~Graph Representation~

Adjacency Matrix for a Weighted Graph:

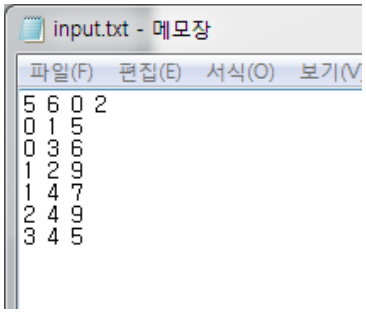


Graph W

	v_0	v_1	v_2	v_3	v_4
v_0	0	5	0	6	0
v_1	5	0	9	0	7
v_2	0	9	0	0	9
v_3	6	0	0	0	5
v_4	0	7	9	5	0

Adjacency Matrix for the graph W on the left

~Data format~



The information for Graph W and start point, destination is provided in "input.txt" file using the data format

File name: input.txt

First line: [number of vertices] [number of edges] [starting point] [destination point]

From the second line: [index of v_i] [index of v_j] [weight]

The input.txt file will include information about the number of vertices, number of edges and each edge with its weight value, and start/end point

The first part of the following program code reads the lines in input.txt file, and stores the graph in a weighted adjacency matrix

※ "input.txt" file should be placed in same location with your python program code

~Python skeleton Code~

```
file = open("input.txt", "r");
# read integers from input text file
file2 = open("output.txt","w");
# write integers to output text file


vnum,enum,st,ed = [int(x) for x in file.readline().split()]
graph = [[0]*vnum for _ in range(vnum)]

for line in file:
    u, v, weight = [int(x) for x in line.split()]
    graph[u][v] = weight
    graph[v][u] = weight
#save edges information as a adjcency matrix

#### Your dijkstra algorithm will be HERE##########

################################################

summary = "%d %d %d\n" % (st, ed, total_weight)
file2.write(summary)

sample_edge1 = "%d %d %d\n" % (0,1,5)
file2.write(sample_edge1)

#output example
```

Your task is to Implement Dijkstra's algorithm (or a shortest path finding algorithm) using the skeleton code provided

Please output your resulting shortest path tree into a file named "output.txt" which has the same data format as the input.

You can find how to write the result into a text file at the end of the skeleton code

~Output format~

File name: output.txt

The output file format is the same as input file format

The "output.txt" should include the

1. ***[index of v_i] [index of v_j] [sum of weight]***

2. ***List of path from source to destination with each cost of edge.***

In given example, shortest path from v_0 to v_2 is $v_0 \rightarrow v_1 \rightarrow v_2$.

Then your output should be

0 2 14

0 1 5

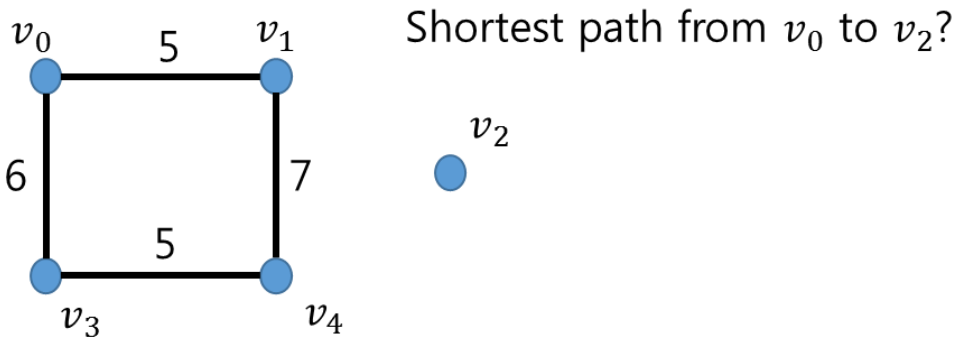
1 2 9

※ Your source code should include input validation code for two exceptional case as follows

- There is no path from source to destination (disconnected graph)
- Start node = End node

※ Assumption

- Every edge has positive integer weight value (Greater than 0)



You should handle this kind of exceptional case if input file requires the shortest path from v_0 to v_2

(2) Analyze your source code which you implemented in problem 1 as follows.
Submit the report in pdf file format named '2015XXXX_yourname_report.pdf'. (30 points)

(a) Provide your time complexity analysis using Big-O notation. (10 points)

(b) Write a pseudo code for the algorithm which will find shortest path between two nodes. (10 points)

(c) Provide brief descriptions of your program. Your description should include such contents: Python version you used, How to run this program, Expected behavior of your program (if necessary) (10 points)

※ Zip your source code and report together for submission.
Save as named '2015XXXX_yourname_PA.zip'