Laboratory – Comparing Priority Queues

Read in a series of undirected weighted graphs which have been described as a list of edges with a starting and a destination vertex and then for each graph output the **length of the shortest path** between the source and the destination.

Perform your algorithm with a Priority Queue implemented with a Sort, one with a Priority Queue implemented with a linear search of a list, and one with a Priority Queue implemented using a Heap.

Compare the times between the three implementations of the Priority Queue and justify your findings bases on algorithmic complexity.

Input format

There will be multiple graphs to analyse. The first line in the file will state the number of cases in the file.
Each case will be composed of
A line with the number of vertices and the number of edges.
A line with the starting vertex and the destination vertex
A series of lines each describing an edge of the graph and consisting of a tuple of integers: the two endpoints then the cost of traversing the edge.

Output format

For each case output the case number then the **length of** shortest path from the start to the destination on a line by itself as shown in the example below.
If there is no path output **No path**

SAMPLE INPUT	SAMPLE OUTPUT
1 5 7 0 2 0 3 1 0 1 5 3 4 1 4 1 1 3 2 3 1 2 2 4 2 2	Case 0: 4

To record observed execution times for different Priority Queues:
In C# use the StopWatch Class, in Java use System.CurrentTimeMillis( )