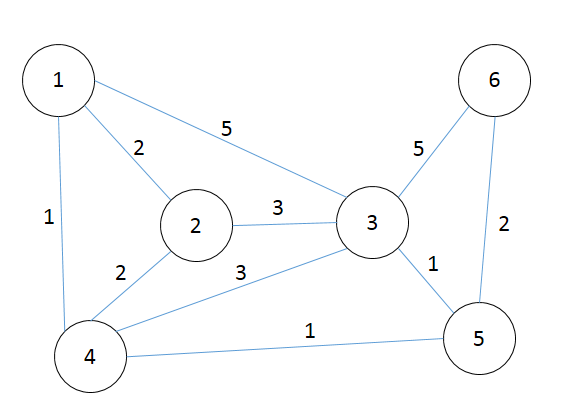
# **Lab 2 - Task 1**

Consider the situation where you have completed your lab task for ELEC1200.3x and have hit the "submit" button. In order for you to receive credit for your work, the packets containing your solution must be sent to the edX server for grading. However, your computer is not directly connected with the edX server. As a result, the packets must pass through a sequence of nodes before it arrives at the edX server. The process by which each node in the network takes incoming an packet and passes it to the next node along the path towards its final destination is known as forwarding.

In this task, we will learn how nodes use the information stored in their routing tables to perform forwarding, and how packets can be routed globally through the network (from source to destination) using only local operations (forwarding).

**INSTRUCTIONS**

Consider the simple network shown below consisting of N = 6 nodes, indexed from 1 to 6. Each link between nodes is labelled by the cost (usually interpreted as delay) associated with taking that link. Nodes that are not connected by links are not directly connected to each other.



The MATLAB code below simulates the process by which packets move from source to destination in this network. In order to perform forwarding, each node in the network maintains a routing table. In our code, we represent each routing table as a matrix with N rows and two colums, where N is the number of nodes in the network (6 in this example). The entire set of routing tables for all nodes in the network is represented by a cell array **RT**, where the cell **RT{i}**, contains the routing table for node **i**.

For example, the routing table of the first node is given by:

RT{1} =

0 0

2 2

4 4

4 1

4 2

4 4

The **m**-th row of the routing table matrix contains the information required to forward the packet to node **m**. The first element **RT{1}(m,1)** indicates the node to which the packet needs to be forwarded to. If the packet is already at its destination, then this entry is 0. The second element **RT{1}(m,2)** shows the total cost to get to node **m**.

Suppose node 1 receives a packet whose destination is node "**5**". Node looks at the **5**th row of the matrix. Since **RT{1}(5,1)=4** , node 1 passes this packet out along the link leading to node 4. The cost to reach node 5 is **RT{1}(5,2)=2**. The entry **RT{1}(1,1)=0** indicates that the packet has arrived at its destination (node 1), and no forwarding is needed. The entry **RT{1}(2,1)=2** indicates that if the packet is destined for node 2, it should go out on the link connecting node 1 to node 2. Thus, nodes 1 and 2 are directly connected (neighbors), and the cost of the link between them is **RT{1}(2,2)=2**. Similarly, **RT{1}(4,:) = [4 1]** indicates that nodes 1 and 4 are directly connected with cost 1. On the other hand, even though nodes 1 and 3 are directly connected, the entry **RT{1}(3,1)=4** tells node 1 that packets destined for node 3 should first goto node 4. This is because the direct link between nodes 1 and 3 is relatively slow (cost = 5), whereas the cost to get to node 3 through node 4 is only **RT{1}(3,1)=4**.

The MATLAB code below simulates the iterative process by which a packet gets from the node **start\_node** to the node **dest\_node** by forwarding at each node. We assume that the routing tables have already been computed and are stored in the cell array **RT**. We start the process by setting the current node of the packet to the starting node with the line of code

**curr\_node = start\_node**

Each iteration proceeds as follows, the current node looks at its routing table, **RT{curr\_node}**, to determine node to pass the packet to. The code then passes the packet to that node by updating the value of **curr\_node** to that node for next iteration. The iteration terminates when the **curr\_node** is equal to zero or when the number of loops exceeds the **timeout**. Note that this is because we set **RT{i}(i,1)=0**, so that **curr\_node=0** indicates that the packet has arrived at its destination.

The code simulates multiple packets going from a variety of start nodes to a variety of destination nodes. The paths taken by these packets are stored in the matrix **path**.

Your task is to revise the code between the lines

% % % % Revise the following code % % % %

and

% % % % Do not change the code below % % % %

so that the variable **curr\_node** at the end of each iteration is updated so that the packet has been forwarded to the next node. Do not change other parts of the code.