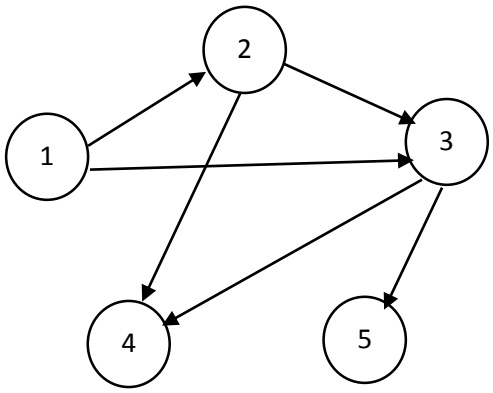
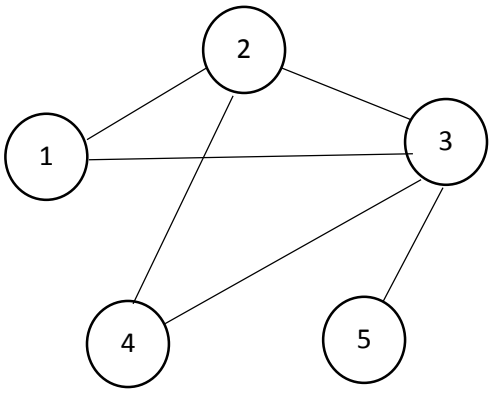
Graph Representation

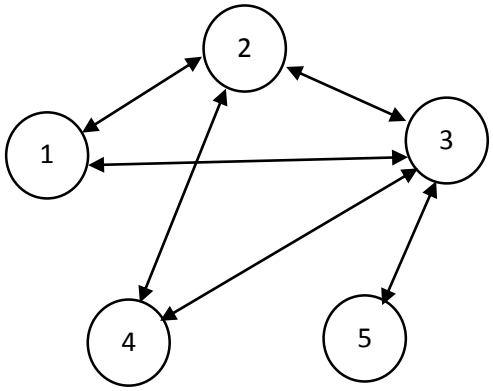
Consider the following graphs-



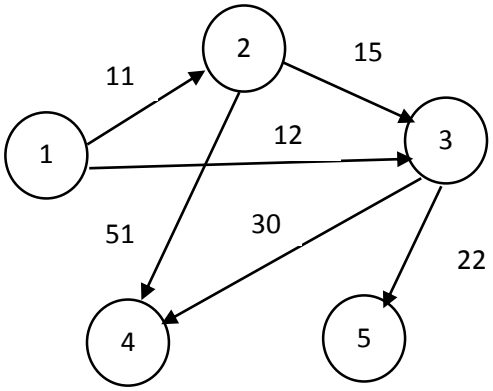
1. Directed graph



2. Undirected graph



3. Bi-directional graph



4. Directed graph with weight (Weighted directed graph)

Question 1: Do you think that a Bi-directional graph is an undirected graph and vice-versa?

Question 2: Can there be weight in the Bidirectional graph of figure 3?

Question 3: Can there be weight in the undirected graph of figure 2?

Question 4: What can we say if all the edges have same weight in figure 4?

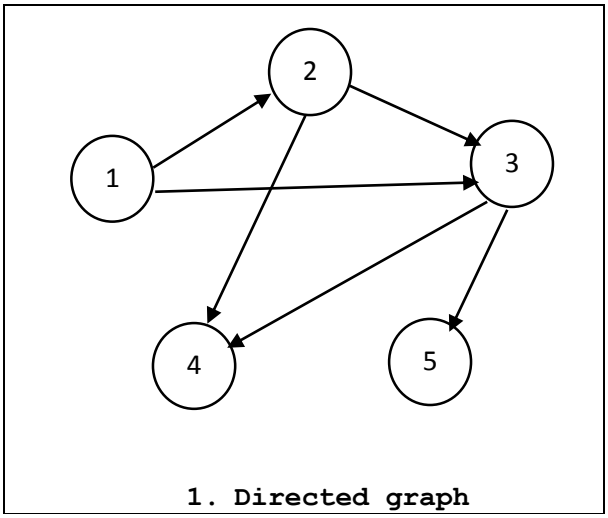
Question 5: What is a Unidirectional graph?

Graph input (without weight) both for directed and undirected graph

```
Number of Nodes | Number of Edges

Node 1 | Node 2
Node A | Node B
.........................
.........................
.........................
Node X | Node Y
```

Example: For the directed graph in figure 1, we have 5 nodes and 6 edges.



1. Directed graph

```
5 6  <-- 5 indicates number of nodes
1 2       6 indicates number of edges
2 3
1 3  --- 6 edges.
2 4      2 4 means we have an edge from node 2 to node 4.
3 4
3 5
```

Question 1: Can the number of nodes and number of edges be equal?

Question 2: Can the number of nodes is greater than the number of edges?

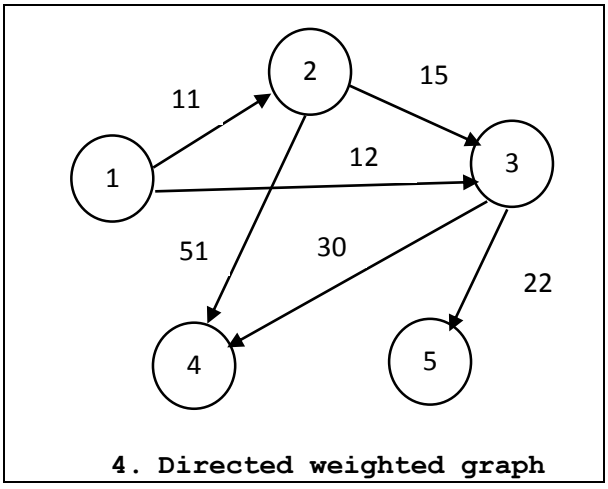
Question 3: How many edges we will need if we want to connect all the nodes with one another?

Graph input (with weight) both for directed and undirected graph

```
Number of Nodes | Number of Edges | Weight between edges

Node 1 | Node 2 | Weight between Node 1 & Node 2
Node A | Node B | Weight between Node A & Node B
.................................................
.................................................
.................................................
Node X | Node Y | Weight between Node X & Node Y
```

Example: For the directed graph in figure 4, we have 5 nodes and 6 edges.



4. Directed weighted graph

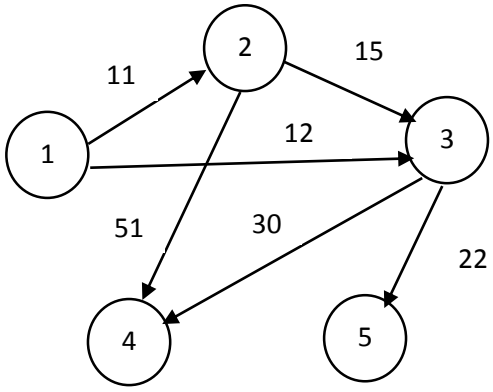
```
                5 indicates number of nodes
5 6  <--------  6 indicates number of edges
1 2 11
2 3 15
1 3 12          6 edges with weight between nodes.
2 4 51          2 4 51 means we have an edge from node 2 to node 4 and
3 4 30          to go from node 2 to node 4 we need weight 51
3 5 22
```

Question 1: In case of 2 4 51, what is the weight to go from node 4 to node 2?

Question 2: If the graph is Bi-directional but the weight between nodes are same then what we can say about 2 4 51?

Question 3: If the graph is undirected and the weight between nodes are same then what we can say about 2 4 51?

Node Adjacency Matrix Vs. Node Adjacency List



4. Directed weighted graph

Node Adjacency Matrix:

Nodes	1	2	3	4	5
1	0	1	1	0	0
2	0	0	1	1	0
3	0	0	0	1	1
4	0	0	0	0	0
5	0	0	0	0	0

Node Adjacency List:

Consider the above Node Adjacency Matrix:

Nodes	1	2	3	4	5
1	0	1	1	0	0
2	0	0	1	1	0
3	0	0	0	1	1
4	0	0	0	0	0
5	0	0	0	0	0

If we color the cells with 1 in the Adjacency matrix, then we can construct the following List which is called Node Adjacency List:

Nodes -> Other Connected Nodes

1 -> | 2 | 3 |

2 -> | 3 | 4 |

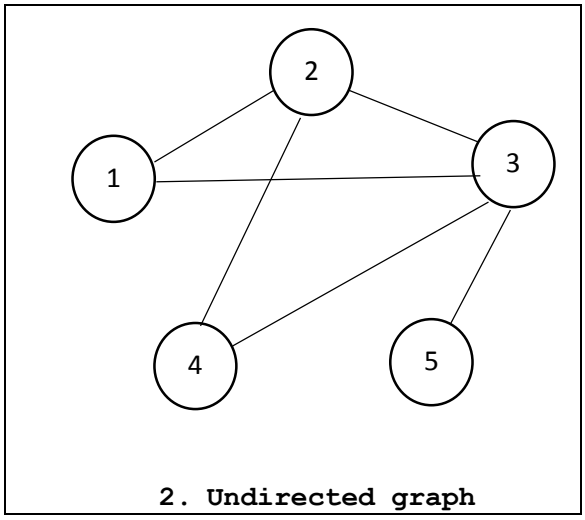
3 -> | 4 | 5 |

4 ->

5 ->

Practice

Now, consider the following graph:



2. Undirected graph

Question 1: Can you build the Node Adjacency Matrix for this graph?

Nodes	1	2	3	4	5
1					
2					
3					
4					
5					

Question 2: Can you build the Node Adjacency List?

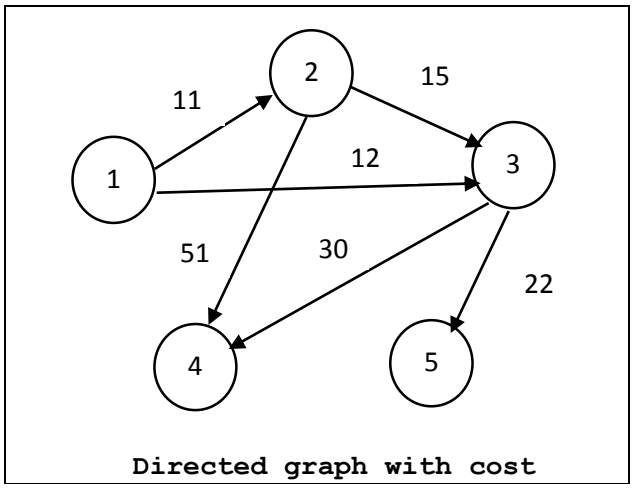
```
1 ->
2 ->
3 ->
4 ->
5 ->
```

Question 3. Can you say which one is static or dynamic? Matrix or List?

Question 4. What happens when we insert a new edge with cost in the graph? Which one will be static or dynamic?

Question 5. Does Node Adjacency List depend on the input order? Will it change if the input order is changed?

Weight Adjacency Matrix vs. Weight Adjacency List



Directed graph with cost

Weight Adjacency Matrix:

Nodes	1	2	3	4	5
1	∞	11	12	∞	∞
2	∞	∞	15	51	∞
3	∞	∞	∞	30	22
4	∞	∞	∞	∞	∞
5	∞	∞	∞	∞	∞

Weight Adjacency List:

Consider the above Weight Adjacency Matrix:

Nodes	1	2	3	4	5
1	∞	11	12	∞	∞
2	∞	∞	15	51	∞
3	∞	∞	∞	30	22
4	∞	∞	∞	∞	∞
5	∞	∞	∞	∞	∞

If we color the cells those don't have infinity values in the Adjacency matrix, then we can construct the following List which we call Weight Adjacency List:

Nodes -> Weights to go to adjacent Nodes

1 -> | 11 | 12 |

2 -> | 15 | 51 |

3 -> | 30 | 22 |

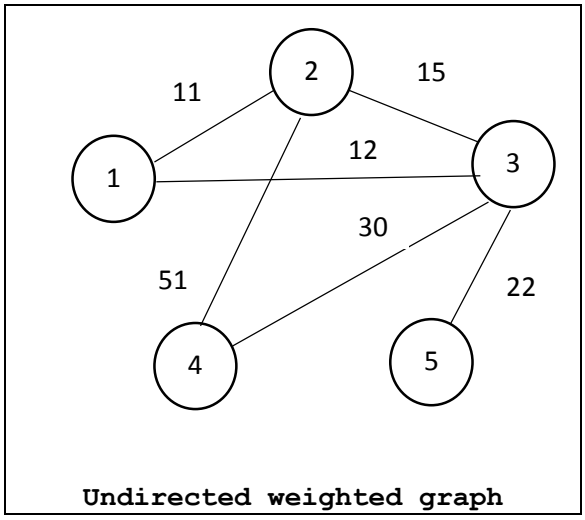
4 ->

5 ->

ALERT !!
We are listing Weights not Nodes!!!
Adjacent Nodes: 1 -> 2, 3
Weight List: 1 -> 11, 12
To go from node 1 to node 2, we need weight 11

Practice

Now, consider the following graph:



Undirected weighted graph

Question 1: Can you build the Weight Adjacency Matrix for this graph?

Nodes	1	2	3	4	5
1					
2					
3					
4					
5					

Question 2: Can you build the Weight Adjacency List?

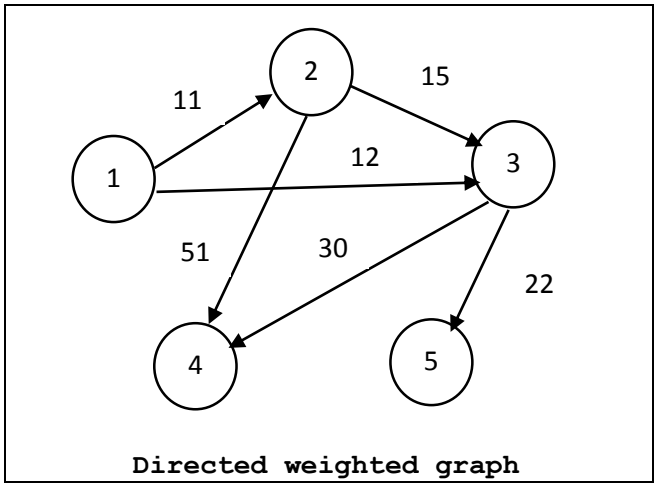
```
1 ->
2 ->
3 ->
4 ->
5 ->
```

Question 3. Can you say which one is static or dynamic? Matrix or List?

Question 4. What happens when we insert a new edge with weight in the graph? Which one will be static or dynamic?

Question 5. Does Weight Adjacency List depend on the input order? Will it change if the input order is changed?

ALL IN ONE NOW



Directed weighted graph

Input

```
5 6      <- 5 indicates number of nodes
         <- 6 indicates number of edges
1 2 11
2 3 15
1 3 12   <- 6 edges with weight between nodes.
2 4 51   <- 2 4 51 means we have an edge from node 2 to node 4 and
3 4 30      to go from node 2 to node 4 we need weight 51
3 5 22
```

NODE ADJACENCY LIST

1 ->	2	3
2 ->	3	4
3 ->	4	5
4 ->		
5 ->		

WEIGHT ADJACENY LIST

1 ->	11	12
2 ->	15	51
3 ->	30	22
4 ->		
5 ->		

WELCOME TO CODE SECTION

Graph Input (without weight):

```
#include<bits/stdc++.h>
using namespace std;

vector<int>Node_Vec[100];

int main()
{
    int node_no, edge_no;

    printf("Enter the number of nodes: ");
    scanf("%d", &node_no);

    printf("Enter the number of edges: ");
    scanf("%d", &edge_no);

    for(int i = 1; i <= edge_no; i++)
    {
        int nodeA, nodeB;
        scanf("%d %d", &nodeA, &nodeB);
        Node_Vec[nodeA].push_back(nodeB);  // for directed graph
    }

    //Print Node Adjacency List by traversing the list
    for(int i = 1; i <= node_no; i++)
    {
        printf("%d -> ", i);
        for(int j = 0; j < Node_Vec[i].size(); j++)
        {
            printf("%d ", Node_Vec[i][j]);
        }
        printf("\n");
    }
}
```

Question 1. If the graph is undirected, where will you modify in this code?

Question 2. for(int j = 0; j < Node_Vec[i].size(); j++) here, why the loop starts from 0?

Graph Input (with weight)

```
#include<bits/stdc++.h>
using namespace std;

vector<int>Node_Vec[100];
vector<int>Node_Cost[100];

int main()
{
    int node_no, edge_no;

    printf("Enter the number of nodes: ");
    scanf("%d", &node_no);

    printf("Enter the number of edges: ");
    scanf("%d", &edge_no);

    for(int i = 1; i <= edge_no; i++)
    {
        int nodeA, nodeB, cost;
        scanf("%d %d %d", &nodeA, &nodeB, &cost);
        Node_Vec[nodeA].push_back(nodeB);  // for directed graph
        Node_Cost[nodeA].push_back(cost);  // for directed graph
    }

    //Print Cost Adjacency List by traversing the list
    for(int i = 1; i <= node_no; i++)
    {
        printf("%d -> ", i);
        for(int j = 0; j < Node_Cost[i].size(); j++)
        {
            printf("%d ", Node_Cost[i][j]);
        }
        printf("\n");
    }
}
```

Question 1. If the graph is undirected, where will you modify in this code?

Question 2. for(int j = 0; j < Node_Cost[i].size(); j++) here, why this loop starts from 0?

Question 3. Why we have used Node_Cost vector instead of Node_Vec?

Prepared by
G. M. Shahariar Shibli
Lecturer, Department of CSE, AUST