

## Experiment No.: 7(C2)

### Title:

There are flight paths between cities. If there is a flight between city A and city B then there is an edge between the cities. The cost of the edge can be the time that flight takes to reach city B from A or the amount of fuel used for the journey. Represent this as a graph. The node can be represented by airport name or name of the city. Use adjacency list or use adjacency matrix representation for graph. Check whether the graph is connected or not. Justify the storage representation used.

### Learning Objectives:

- To understand concept of Graph data structure
- To understand concept of representation of graph.

### Learning Outcome:

- Define class for graph using Object Oriented features.
- Analyse working of functions.

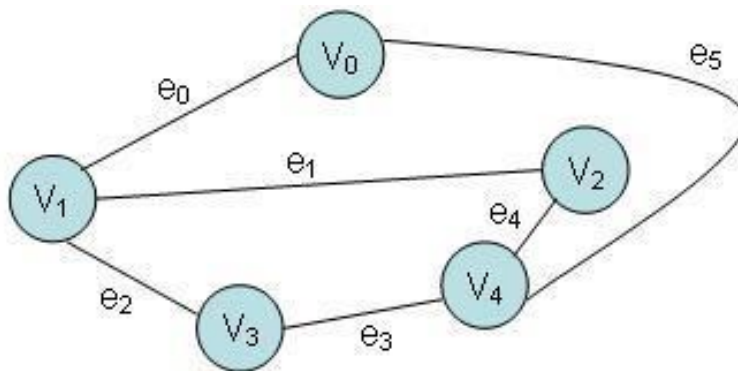
### Theory:

#### Graph:

Graphs are the most general data structure. They are also commonly used data structures. A non-linear data structure consisting of nodes and links between nodes.

#### Undirected graph:

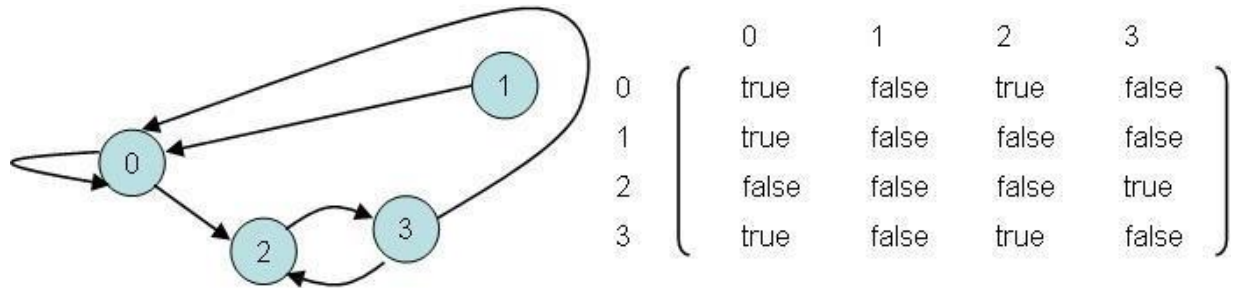
- An undirected graph is a set of nodes and a set of links between the nodes.
- Each node is called a **vertex**, each link is called an **edge**, and each edge connects two vertices.
- The order of the two connected vertices is unimportant.
- An undirected graph is a finite set of vertices together with a finite set of edges. Both sets might be empty, which is called the empty graph.



## Graph Implementation:

Different kinds of graphs require different kinds of implementations, but the fundamental concepts of all graph implementations are similar. We'll look at several representations for one particular kind of graph: directed graphs in which loops are allowed.

### Representing Graphs with an Adjacency Matrix



Graph and adjacency matrix

### Adjacency Matrix:

An adjacency matrix is a square grid of true/false values that represent the edges of a graph

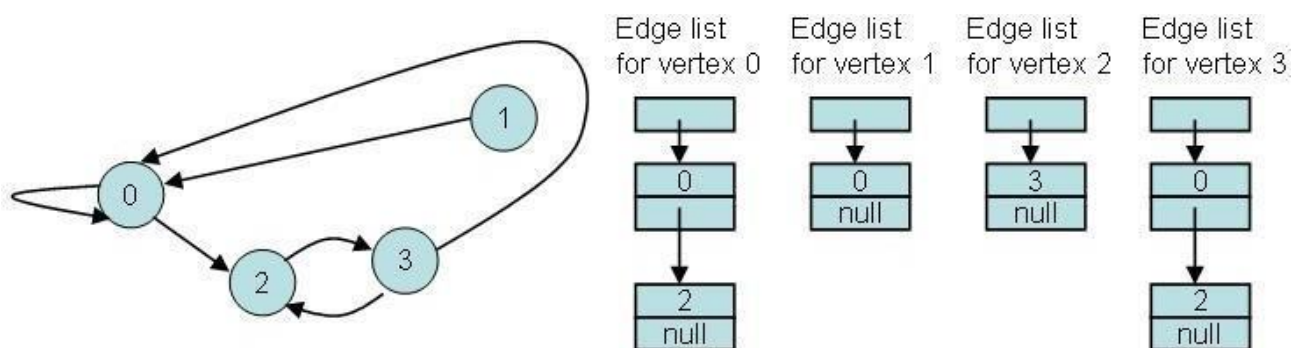
- If the graph contains  $n$  vertices, then the grid contains  $n$  rows and  $n$  columns.
- For two vertex numbers  $i$  and  $j$ , the component at row  $i$  and column  $j$  is true if there is an edge from vertex  $i$  to vertex  $j$ ; otherwise, the component is false.

We can use a two-dimensional array to store an adjacency matrix:

```
boolean[][] adjacent = new boolean[4][4];
```

Once the adjacency matrix has been set, an application can examine locations of the matrix to determine which edges are present and which are missing.

### Representing Graphs with Edge Lists



Graph and adjacency list for each node

### Adjacency List:

- A directed graph with  $n$  vertices can be represented by  $n$  different linked lists.
- List number  $i$  provides the connections for vertex  $i$ .
- For each entry  $j$  in list number  $i$ , there is an edge from  $i$  to  $j$ .

Loops and multiple edges could be allowed.

## Representing Graphs with Edge Sets

To represent a graph with  $n$  vertices, we can declare an array of  $n$  sets of integers. For example:

**IntSet [] connections = new IntSet[10]; // 10 vertices**

A set such as connections[ $i$ ] contains the vertex numbers of all the vertices to which vertex  $i$  is connected.

### Conclusion:-

---

---

---

### Questions:-

1. What are different types of graph?

---

---

---

---

---

2. What are different ways to represent the graph?

---

---

---

---

---

3. What is the complexity of representing graph using adjacency matrix?

---

---

---

---

---

**4. Which operations are possible if graph is represented as adjacency list?**

---

---

---

---

---

**5. What are the advantages and disadvantages of various graph representation?**

---

---

---

---

---

**Staff Signature & Date**

