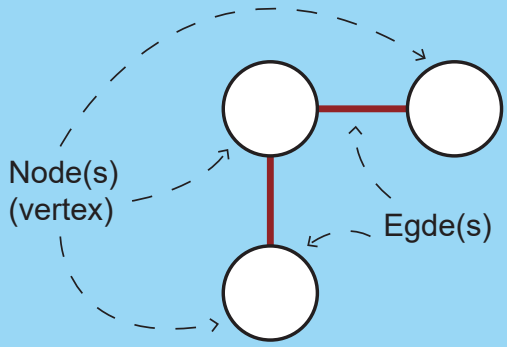
The ADT Grahp

Definition:

- The **ADT Graph** consists of a finite set of vertices(or nodes) and set of Edges which connect a pair of nodes, together with primitve operations.

Graph ADT

Representation:



V is a set of **vertex**, **E** is a collection of **edges**

{ *inv:* e_i (V_k, V_j) *with* $0 \leqslant k \leqslant j$ $|V|$ }

Primitive Operations:

+ createGraph:	⟶ *Graph*
+ addVertex: *Graph x Vertex*	⟶ *Graph*
+ addEgde: *Graph x Vertex x Vertex*	⟶ *Graph*
+ removeVertex: *Graph x Vertex*	⟶ *Graph*
+ removeEgde: *Graph x Egde*	⟶ *Graph*
+ vertexAdjacent: *Graph x Vertex*	⟶ *List of Vertex*
+ areConnected: *Graph x Vertex x Vertex*	⟶ *boolean*

createGraph()

"Creates a new empty graph"

{ *pre:* TRUE}

{ *post:* }

createGraph()
"Creates a new empty graph"

{ *pre:* TRUE}

{ *post:* }

addVertex(*Graph, Vertex***)**
"Adds a new vertex to the graph"

{ *pre:* Graph != null ^ V1 != null}

{ *post:* }

addEgde(*Graph, Vertex, Vertex***)**
"Adds a new edge to the graph connecting two existing vertices"

{ *pre:* V1 != null ^ V2 != null ^ Graph != null}

{ *post:* }

removesVertex(*Graph, Vertex***)**
"Removes a given vertex from the graph"

{ *pre:* V1 != null ^ V2 != null}

{ *post:* }

removesEgde(*Graph, Vertex***)**
"Removes a given egde from de the graph disconecting two vertices "

{ *pre:* TRUE}

{ *post:* }

vertexAdjacent(*Graph, vertex***)**
"Returns a list of vertices that contains all the adjacent vertices of a given vertex"

{ *pre:* Graph != null ^ V1 != null}

{ *post:* }

areConected(*Graph, vertex, vertex***)**
"Indicates whether two given vertices share an egde or not"

{ *pre:* Graph != null ^ V1 != null ^ V2 != null}

{ *post:* }