Nicolás Penagos Montoya
Mateo Loaiza Zuñiga
Juan Jose Calderón

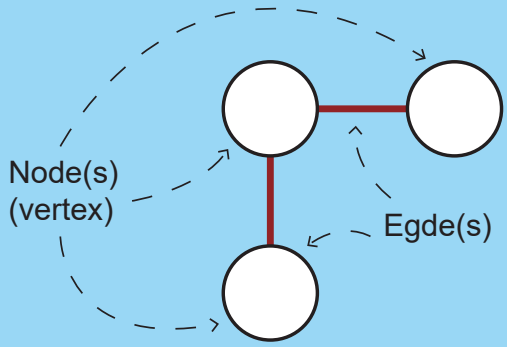
The ADT Grahp

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* $0 \leqslant k \leqslant j$ $|V|$ }

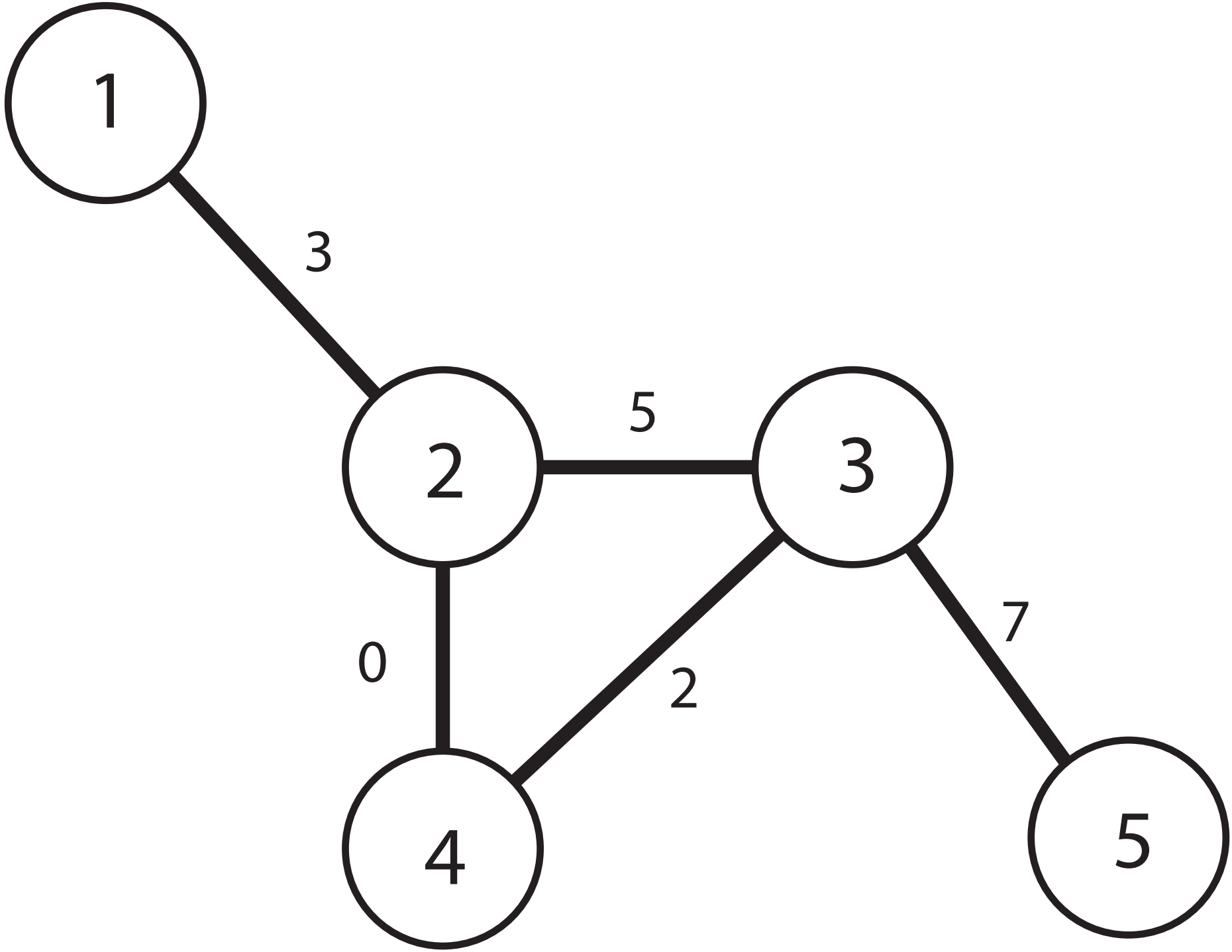
Primitive Operations:

+ 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: Graph G* }


1
3
2
5
3
0
2
7
4
5

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

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

{ *post:*

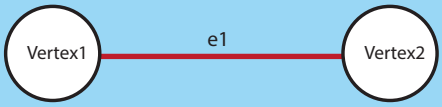
}

addEgde(*Graph, Vertex1, Vertex1, e1***)**
"Adds a new edge to the graph connecting two existing vertices"

{ *pre:* Vertex1 != null ^ Vertex2 != null ^ Graph != null^ (Vertex1^ Vertex2 ∈ Graph)}

}

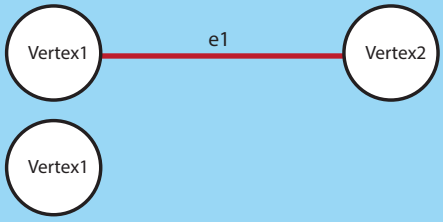
{ *post:*



}

removesVertex(*Graph, Vertex2***)**
"Removes a given vertex from the graph"

{ *pre:*Vertex1 != null ^ Vertex2 != null ^ Graph != null^ (Vertex1^ Vertex2 ∈ Graph)}



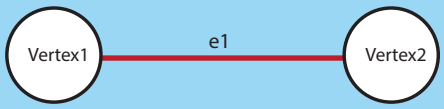
{ *post:*

}

removesEgde(*Graph, e1*)

"Removes a given egde from de the graph disconecting two vertices "

{ *pre:*Vertex1 != null ^ Vertex2 != null ^ Graph != null^ (Vertex1^ Vertex2 Graph)^ e1!=null}



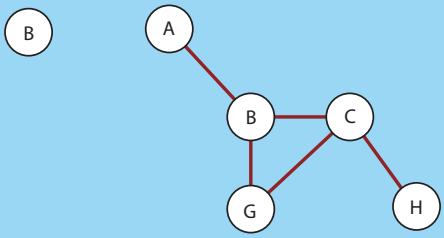
{ *post:*

}

vertexAdjacent(*Graph, B*)

"Returns a list of vertices that contains all the adjacent vertices of a given vertex"

{ *pre:*



{ *post:*

areConected(*Graph, vertex1, vertex2*)

"Indicates whether two given vertices share an egde or not"

{ *pre:*Vertex1 != null ^ Vertex2 != null ^ Graph != null^ (Vertex1^ Vertex2 ∈Graph)}

{ *post: if the vertices share an egde it returns TRUE, else it returns FALSE.*
}