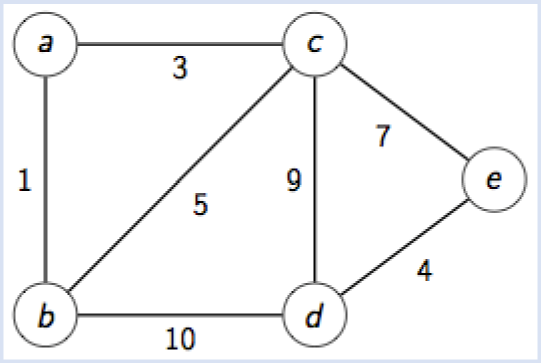
TAD Graph $\langle \boldsymbol{K}, \boldsymbol{E} \rangle$



{inv: $\boldsymbol{G} = (V, E), \forall (\boldsymbol{a}, \boldsymbol{b}) \in E \; \exists \; (\boldsymbol{b}, \boldsymbol{a}) \in E$}

<u>Primitive operations:</u>

• getGraph		->ArrayList
• getWeigthMatrix:		->double[][]
• addVertex:	Vertex<E>	->void
• getVertex:	int	->Vertex<E>
• deleteVertex:	Vertex<E>	->void
• addEdge:	Vertex<E> Vertex<E>	->void
• addEdge:	Vertex<E> Vertex<E> double	->void
• deleteEdge:	Vertex<E> Vertex<E>	->void
• isDirected:	Vertex<E>	->boolean
• getAdjacents:	Vertex<E>	->ArrayList

getGraph
"Returns the graph."
{pre: TRUE}
{post: ArrayList with its respective identifier (K) and the object it contains (E)}
Analyzer

getWeigthMatrix
"Returns a matrix where we can observe the weights of each edge."
{pre: A weighted graph must exist}
{post: Graph weight matrix}
Analyzer

addVertex (v)
"Adds a new vertex to the graph."
{pre: TRUE}
{post: The vertex has been added}
Modifier

getVertex (index)
"Returns the vertex of a given index."
{pre: The vertex must exist}
{post: The vertex has been returned}
Analyzer

deleteVertex (v)
"Deletes the vertex v from the graph."
{pre: The vertex must exist}
{post: The vertex has been deleted}
Modifier

addEdge (u,v)
"Adds a new edge to the graph given two vertexes."
{pre: TRUE}
{post: The edge has been added between the two vertices}
Modifier

addEdge (u,v,w)
"Add an edge between the two vertices, assigning it a weight w."
{pre: TRUE}
{post: The edge has been added between the two vertices with its respective weight.}
Modifier

deleteEdge (u,v)
"Delete an edge between the two vertices."
{pre: The edge must exist}
{post: The vertex has been deleted}
Modifier

isDirected (v)
"Returns a boolean indicating if the graph is directed or undirected."
{pre: The graph must exist}
{post: Indicates if the graph is directed or undirected}
Analyzer

getAdjacents (v)
"Given a vertex, it returns an ArrayList with the nodes adjacent to said vertex."
{pre: The vertex must exist}
{post: ArrayList with the nodes adjacent to the given vertex}
Analyzer