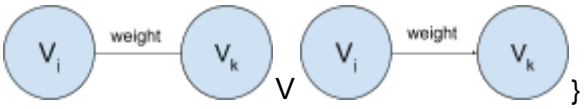
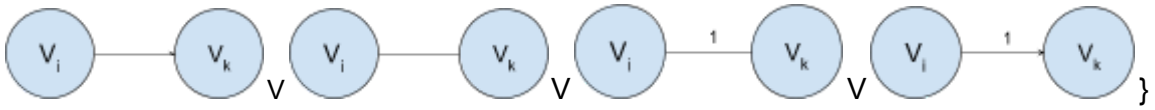
TAD Graph
$G = (V, E);\ V = \{v_0, v_1, v_2, ...v_i\}\ ,\ E = \{\{v_{s1}, v_{e1},\ w_1\}, \{v_{sw}, v_{e1},\ w_2\}...\{v_{ia}, v_{eb},\ w_k\}\}$
{ inv: $\forall\ e_i(v_k, v_j, w_h)\ (v_k \wedge v_j \in V) \wedge e_i \in E \wedge w_h \in Z$ }

Primitive operations:

• Graph	boolean x boolean	
• addVertex	Vertex	→ boolean
• addEdge	Vertex x Vertex x weight	→ boolean
• addEdge	Vertex x Vertex	→ boolean
• isEmpty		→ boolean
• removeVertex	Vertex	→ boolean
• removeEdge	Vertex x Vertex	→ boolean
• getWeightMatrix		→ int[ ][ ]
• getAdjacencyList		→ Vertex[ ][ ]
• getEdgeList		→ Edge[ ]
• getEdgeList	Vertex	→ Edge[ ]
• getVertexList		→ Vertex[ ]
• getAdjacentVertices	Vertex	→ Vertex[ ]

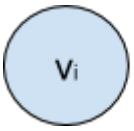
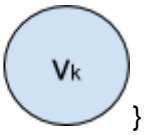
Graph(weightedGraph, directedGraph)
“Creates a new weighted or unweighted graph ”
{ pre: true }
{ post: Graph ≠ Ø, weighted = true ∨ false, directed = true ∨ false}
Operation type: constructor

addEdge($v_i, v_k, weight$)
“Adds a new edge to the Graph with the given weight”
{ pre: (v_i, v_k) ≠ null, $v_i \wedge v_k \in V$, weighted = true}
{ post:  }
Operation type: modifier

addEdge(v_i, v_k)
“Adds a new edge to the Graph with weight 1 if weighted or edge without weight if not weighted”
{ pre: (v_i, v_k) ≠ null, $v_i \wedge v_k \in V$ }
{ post:  }
Operation type: modifier

addVertex(v_k)
“Adds a new vertex to the Graph.”
{ pre: (Graph, v_k) ≠ null}
{ post: Graph = { $v_0, v_1, v_2 ... v_n, E$ } U { v_k }
Operation type: modifier

removeVertex(v_k)
“Erase a vertex from the Graph.”
{ pre: ($Graph, v_k$) ≠ null $\wedge v_k \in Graph$ }
{ post: $(e(v_k, v_x, w), e(v_x, v_k, w)) = null$, Graph = { $v_0, v_1 ... v_{n-1}$ }, }
Operation type: modifier

removeEdge(v_i, v_k)
“Erases an edge from the Graph.”
{ pre: ($Graph, v_i, v_k$) ≠ null ∧ $v_i, v_k \in Graph$ }
{ post:   }
Operation type: modifier

isEmpty()
“Determines if the graph is empty.”
{ pre: Graph ≠ null}
{ post: true if Graph = Ø, false if Graph ≠ Ø}
Operation type: analyzer

getWeightMatrix( )
“Returns a weight matrix representation of this graph.”
{ pre: $Graph$ ≠ null}
{ pos: true}
Operation type: analyzer

getAdjacencyList( )
“Returns an adjacencyList representation of this graph.”
{ pre: $Graph$ ≠ null}
{ pos: true}
Operation type: analyzer

getEdgeList( )
“Returns a list of the edges of this graph.”
{ pre: $Graph$ ≠ null}
{ pos: true}
Operation type: analyzer

getEdgeList(v_k)
“Returns a list of all the edges associated to vertex v_k”
{ pre: (Graph, v_k) ≠ null}
{ post: true}

getVertexList( )
“Returns a list of the vertex of this graph.”
{ pre: $Graph$ ≠ null}
{ pos: true}
Operation type: analyzer

getAdjacentVertices(v_k)
“Returns a list of all the adjacent vertices to vertex v_k”
{ pre: (Graph, v_k) ≠ null}
{ post: true}
Operation type: modifier