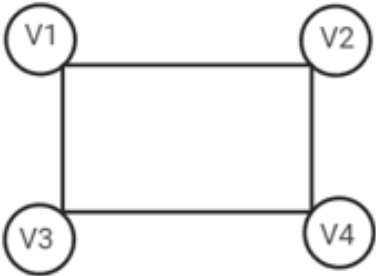


TAD Grafo
GrafoTAD = {V,E}; V = {V1,V2,V3...Vn}; E = {E1,E2,E3...En}
Un grafo consiste en un conjunto V de nodos y E de aristas. Una arista vendria siendo es un par de nodos como (V1,V2) pertenecientes al conjunto V. El orden de estos depende del grafo, si es dirigido iporta el orden, si no es dirigido no importa.
V = {V1,V2,V3,V4} E = {(V1,V2),(V1,V4),(V2,V3),(V3,V4)}

{inv: V = {V1,V2,V3...V4}} {inv: V = {E1,E2,E3...E4}} {inv: V != empty}
Primitive operations : createGraph: --> Graph createNode: Element x arrayConections x arrayWeights --> Graph addNode: node x arrayListConections x numWeights x Graph --> Graph dijkstra: numSource x Graph --> Graph floydWarshall: Graph --> arrayNum

createGraph(): Constructor
"Create a new empty graph"
<pre>{ pre: true } { post: Instance and create a new graph with a determined nodes and matrix }</pre>
createNode(island): Modifier
"Create a new node using an island and modify the adjacency matrix and array of nodes"
<pre>{ pre: The graph must be initialized } { post: Adjacency matrix modified and a new vertex created inside the ArrayList of nodes }</pre>
addNode(node, connections, weights): Modifier
"Insert a new node inside the graph with connections associated with their respective weights"
<pre>{ pre: The graph must be initialized } { post: The graph has a new node inserted with connections and weights }</pre>
dijkstra(): Analyzer
"Use the acquaintance dijkstra method to return a matrix of mininum weights"
<pre>{ pre: The graph must be initialized and have its respective adjacency matrix} { post: Returns a matrix of calculated weights }</pre>
floydWarshall(): Analyzer
"Use the acquaintance floydWarshall method to return a matrix of weights"
<pre>{ pre: The graph must be initialized and have its respective adjacency matrix} { post: Returns a matrix of calculated weights }</pre>