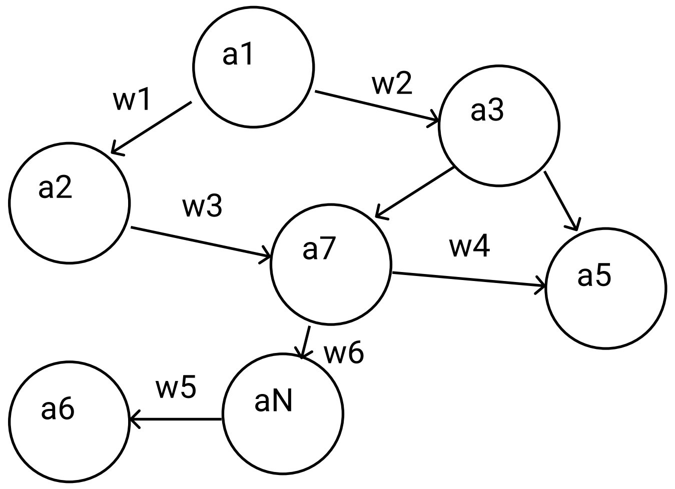
ADT GraphArray



Invariant
A={ a1, a2, a3, a4,a5,a6...aN}
W={w1,w2,w3,w4,w5,w6...wN}

1. wN>0
2. A != empty
3. No double connection (⟷)

Primitive operations:

-createGraph: ->Graph
-createVertex:Element x arrayConections x arrayWeigths ->Grap
-addVertex:vertex x arrayListConections x numWeigths x Graph
->Graph
-dijkstra:numSource x Graph ->Graph
-floydWarshall: Graph ->arrayNum

GraphArray( ) : Constructor

“Create a new empty graph”

{ pre: true }
{ post: fInstance and create a new graph with a determinated vertexes and matrix
}

createVertex(island) : Modifier

“Create a new vertex using an island and modifier the adyacency matrix and array of vertexes”

{ pre: The graph must be initializated }
{ post: Adyacency matrix modified and a new vertex created inside the ArrayList of vertexes }

addVertex(vertex, connections, weights) : Modifier

“Insert a new vertex inside the graph with a connections associated adn their respective weights”

{ pre: The graph must be initializated }
{ post: The graph with a new vertex inserted and connections and weights }

dijkstra( ) : Analyzer

“Use the acquaintance dijkstra method to return a matrix of mininum weights”

{ pre: The graph must be initialized and with its adjacency matrix }
{ post: Return a matrix of calculated weights }

floydWarshall( ) : Analyzer

“Use the acquaintance floydWarshall method to return a matrix of weights”

{ pre: the graph must be initialized and with its adjacency matrix }
{ post: Return a matrix of calculated weights }