TAD <Graph >

Graph={arrayList, Matrix}

Inv: {vertex diferent from null and vertices>0}

Primitive Operations:

AddVertice(name): -> Void

RemoveVertice(V1): -> Void

AddEdge(V1, V2): -> Void

RemoveEdge(V1, V2): -> Void

FindVertex(Name): -> Vertice

BFS(V): ->Void

DFS(): ->Void

Dijkstra(V1, V2, weight): ->ArrayList

Floyd(): ->double[][]

PrimL(): ->GraphAdjacencyList

PrimM(): ->GraphAdjacencyMatrix

KruskalL(): -> GraphAdjacencyList

KruskalM(): -> GraphAdjacencyMatrix

Graph

AddVertice(Name)

“Creates a new vertexes and adds it to graph”

{pre: Name}

{pos: Void}

RemoveVertice(V)

“Removes vertexes from graph and all its connections”

{pre: Vertex}

{pos: Boolean}

AddEdge(V1, V2)

“Adds edge between two vertexes”

{pre: V1,V2}

{pos: Void}

RemoveEdge(V1, V2)

“Removes edge between two vertexes”

{pre: V1, V2}

{pos: Void}

FindVertex(Name)

“Finds Vertex”

{pre: Name}

{pos: Vertex}

Dijkstra(V1, V2, Weight)

“Finds shortest path between two vertexes when sending a certain weight”

{pre: V1, V2, Weight}

{pos: ArrayList<Vertex>}

BFS()

“Explores arraylist from first vertice spreading level by level”

{pre: }

{pos: Void}

DFS()

“Explores arraylist from first vertice going as deep as possible before backtracking”

{pre: }

{pos: Void}

Floyd()

“Makes a matrix that saves all the minimum path weights between any Vertex, if its between the same vertexes its equal to 0, if Vertexes aren’t connected directly or indirectly then the value will be infinite.”

{pre: }

{pos: double[][]}

PrimL()

“Gets the minimum path by weight between all vertices without any cicles using each vertex as a index to find smallest edges.”

{pre: }

{pos: GraphAdjacencyList}

PrimM()

“Gets the minimum path by weight between all vertices without any cicles using each vertex as a index to find smallest edges.”

{pre: }

{pos: GraphAdjacencyMatrix}

KruskalL()

“Gets the minimum path by weight between all vertices without any cicles, even between non connected Vertexes it goes edge by edge from smallest to largest until everything is connected.””

{pre: }

{pos: GraphAdjacencyList}

KruskalM()

“Gets the minimum path by weight between all vertices without any cicles, even between non connected Vertexes it goes edge by edge from smallest to largest until everything is connected.”

{pre: }

{pos: GraphAdjacencyMatrix}