ADT Undirected Graph

Representation:
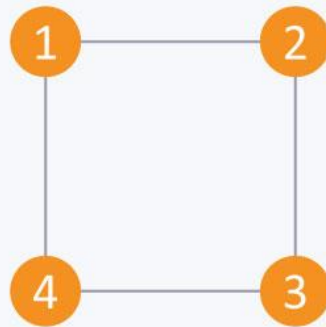
Invariant:
G = {V,E**}** | V = Vertex ∧ E = Edges

Operations:

CreateGraph		→Graph
AddVertex	Vertex	
AddEdge	Vertex,Vertex, Integer	
GetVertex	Vertex	→Vertex
GetVertexes		→List
DeleteVertex	Vertex	
DeleteEdge	Edge	
BFS	Vertex	
DFS	Vertex	
Clear		
IsEmpty		→Boolean
IsEdge	Vertex,Vertex	→Boolean
Size		→Integer
ShortestPath	Vertex,Vertex	→List

AddVertex(Vertex)

Adds a Vertex to the graph.
Pre: void
Post: V = V + 1

AddEdge(Vertex,Vertex)

Adds a weighted edge from one vertex to
another.
Pre: void
Post: E = E + 1

GetVertex(Vertex)

Gives the information of a Vertex.
Pre: Vertex $\in$ Graph $\land$ Vertex $\neq$ nil
Post: Vertex

GetVertexes()

Gives the list of the vertexes of the graph.
Pre: Vertex(i) $\in$ Graph $\land$ Vertex(i) $\neq$ nil
Post: Vertex {V1,V2...Vn}

DeleteVertex(Vertex)

Deletes a Vertex of the graph.
Pre: Vertex $\in$ Graph $\land$ Vertex $\neq$ nil
Post: V = V - 1

DeleteEdge(Edge)

Deletes an edge of the graph.
Pre: Edge $\in$ Graph $\land$ Edge $\neq$ nil
Post: Graph.weight = Graph.weight - e.weight

BFS(Vertex)

Explores all the neighbor nodes of a vertex selected
as root node.
Pre: Vertex ∈ Graph ∧ Vertex ≠ nil
Post: void

DFS(Vertex)

Explores the possible routes as far as possible of a vertex
selected as root node.
Pre: Vertex ∈ Graph ∧ Vertex ≠ nil
Post: void

Clear()

Deletes all the vertexes and edges.
Pre: void
Post: void

IsEmpty()

Looks if there's no edges or vertexes in the graph.
Pre: void
Post: true ∨ false

IsEdge(Vertex,Vertex)

Looks if there's an edge between two specific vertexes.
Pre: Vertex ∈ Graph ∧ Vertex ≠ nil
Post: true ∨ false

Size()

Returns the number of vertexes.
Pre: void
Post: V.

ShortestPath(Vertex)

Searches for the shortest path to go from a source vertex to a destination vertex.
Pre: Source $\in$ Graph $\land$ Source $\neq$ nil $\land$ Destination $\in$ Graph $\land$ Destination $\neq$ nil
Post: Vertexes{Vs,Vi,…Vd}