TAD<Graph>

G = (V, E), where V is a set of vertices, and E is a set of edges

{inv: There cannot be two vertexes with the same value on the Graph.}

Primitive operations:

- Graph	constructor	Graph	-> Graph
- newVertex	modifier	Graph x Value	-> Graph
- deleteVertex	modifier	Graph x Value	-> Graph
- edge	modifier	Graph x Value1 x Value2	-> Graph
- edgeWeight	modifier	Graph x Value1 x Value2 x Weight	-> Graph
- deleteEdge	modifier	Graph x Value1 x Value2	-> Graph

Graph()
Creates a new Graph
{ pre: TRUE }
{ post: Graph is created}

newVertex (G, u)
Adds vertex u to the graph G.
{ pre: TRUE }
{ pos: The vertex is added to the graph G }

deleteVertex (G, u)
Removes vertex u from the graph G.
{ pre: u must belong to the set of vertices of the graph G }
{ pos: The vertex is removed from the graph G }

edge (G, u, v)
Adds the arc or edge (u,v) to the graph G.
{ pre: u and v must belong to the set of vertices of the graph }
{ post: An edge connecting u with v is created in the graph G }

edgeWeight (G, u, v, w)
For a valued graph, adds the arc (u,v) to the network and the cost of the edge, w.
{ pre: u and v must belong to the set of vertices of the graph }
{ post: An weighted edge connecting u with v is created }

deleteEdge (G, u, v)
Removes arc(u,v) from the graph G.
{ pre : There must be an edge between u and v }
{ pos : The edge is removed from the graph G}