TAD Graph

Graph G = (V, E)

{inv: |V| = n ∧ {{u, v} / u, v ∈ V → {u, v} ∈ E }}

Operaciones primitivas:

Graph:		→ Graph
addVertex:	Vertex v	→ Graph
addEdge:	Vertex v1 X Vertex v2	→ Graph
deleteVertex:	Vertex v	→ Graph
deleteEdge:	Edge e	→ Graph
searchVertex:	Element e	→ Vertex
searchEdge:	Vertex v1 X Vertex v2	→ Edge

Especificación operaciones:

Graph()
Builds an empty Graph
{pre: TRUE}
{post: $Graph\ G$ }

addVertex(Vertex v)
adds a Vertex v in Graph
{pre: TRUE}
{post: Graph G}

addEdge(Vertex v1, Vertex v2)
adds a Edge e in Graph
{pre: Graph G ∧ $v1\ \in V \wedge\ v2 \in V$}
{post: Graph G}

deleteVertex(Vertex v)
delete a Vertex v from Graph
{pre: Graph G ∧ $v\ \in V$ }
{post: Graph G}

deleteEdge(Edge e)
delete a Edge e from Graph
{pre: Graph G ∧ $e\ \in E$ }
{post: Graph G}

searchVertex(Element e) search a Vertex v from Graph {pre: Graph G ∧ $v \in V$ } {post: $\mathrm{Vertex}\ v$ }
searchEdge(Vertex v1, Vertex v2) search an Edge {v1, v2} from Graph {pre: Graph G ∧ $\{\mathrm{v1}, \mathrm{v2}\} \in \mathrm{E}$ } {post: $\mathrm{Edge}\ \mathrm{e}$ }