|  |
| --- |
| **TAD Graph** |
| Graph = {V = {v1, v2, …, vn}, E = {e1 = (vi1, vj1, w1), e2 = (vi2, vj2, w2), em = (vim, vjm, wm)}, directed, weighted} |
| {inv:  1. ∀ek ∈ E, vik ∈ V 𝖠 vjk ∈ V, wk > 0  2. directed = false ⇒ (∀(𝑎, 𝑏) ∈ 𝐸 ∃ (𝑏, 𝑎) ∈ 𝐸, 𝑎, 𝑏 ∈ 𝑉)  3. weighted = false ⇒ ∀ek ∈ E, wk = 1  } |
| **Primitive Operations**   * Graph <> → <Graph> Constructor * insertVertex <Vertex> → <Graph> Modifier * insertEdge <Vertex, Vertex> → <Graph> Modifier * delete < Vertex> → <Graph> Modifier * searchVertex <Graph> → List<Vertex> Analyzer * dijkstra < Vertex, Vertex> → Analyzer |

Operations

|  |
| --- |
| Graph (boolean directed, boolean weighted, int n) |
| Create a new graph that may or may not be directed or weighted. |
| {pre: } |
| {post: Graph = {V={}, E={}, directed, weighted } |

|  |
| --- |
| insertVertex (Vertex v) |
| Insert a vertex in the graph. |
| {pre: v ∉ g.V} |
| {post: v ∈ g.V} |

|  |
| --- |
| insertEdge (Vertex v1, Vertex v2) |
| Add an edge of weight 1 that goes from v1 to v2. If the graph is not directed, it also adds it  from v2 to v1. |
| {pre: v1, v2 ∈ g.V } |
| {post: edge = (v1, v2, 1) ∈ g.E. If g.directed = false, edge = (v2, v1, 1) ) ∈ g.E } |

|  |
| --- |
| insertEdge (Vertex v1, Vertex v2, double weight) |
| Add an edge of weight 1 that goes from v1 to v2. If the graph is not directed, it also adds it  from v2 to v1. |
| {pre: v1, v2 ∈ g.V, g.weight = true, w > 0} |
| {post: edge = (v1, v2, weight) ∈ g.E. If g.directed = false, edge = (v2, v1, weight) ) ∈ g.E } |

|  |
| --- |
| delete (Vertex v) |
| Eliminate v from the graph |
| {pre: v ∈ g.V } |
| {post: v ∉ g.V. All vertices that are incidents with v ∉ g. E  } |

|  |
| --- |
| dijkstra (Vertex u, Vertex v) |
| Carry out the Dijkstra algorithm, taking u as the initial vertex and v as the target vertex |
| {pre: u ∈ g.V, v ∈ g.V, g} |
| {post: ∀v ∈ g.V, adds attributes u.pred and u.d, corresponding respectively to the  predecessor and the distance added by Dijkstra's algorithm} |

|  |
| --- |
| search(Vertex v) |
| Returns if there is a vertex with the given value in the graph. |
| {pre: } |
| {post: true if ∃x ∈ g.V : value |