*Julián Santiago Tauta*

*Juan Camilo Tobar*

**TAD Grafo**

|  |
| --- |
| **TAD Grafo** |
| Grafo = {Contenedora de Vértices = <vertices>, Contenedora de Aristas = <adyacencias>} |
| {Inv: Grafo.DarVertice != null, Grafo.DarArista != null} |
| Operaciones:   * CrearGrafo: Boolean * AgregarVertice: Grafo x Vertice * EliminarVertice: Grafo x Vertice Boolean * DarVertice: Grafo x Llave Vertice * AgregarArista: Grafo x Vertice x Vertice x Peso * EliminarArista: Grafo x Arista Boolean * DarArista: Grafo x Arista Arista * Bfs: Grafo Colección de Vertices * Dfs: Grafo Colección de Vertices |

|  |
| --- |
| **CrearGrafo (**Boolean**)**  ” Crea un nuevo Grafo”  {pre: TRUE}  {post: vertices != null, adyacencias != null} |

|  |
| --- |
| **AgregarVertice (**Vertice**)**  ” Agrega un nuevo vértice al grafo y lo sitúa en la contenedora de vértices”  {pre: vertice.valor ϵ TipoValor}  {post: TRUE} |

|  |
| --- |
| **EliminarVertice (**Vertice**)**  ” Elimina el vértice de la contenedora de vértices”  {pre: TRUE}  {post: TRUE} |

|  |
| --- |
| **DarVertice (**Llave**)**  ” Retorna el vértice correspondiente con la llave de la contenedora de vértices, sino existe retorna null”  {pre: TRUE}  {post: TRUE} |

|  |
| --- |
| **AgregarArista (**Vertice, Vertice, Peso**)**  ” Agrega una nueva adyacencia representada en una arista que contiene ambos vértices”  {pre: Peso ϵ TipoPeso}  {post: TRUE} |

|  |
| --- |
| **EliminarArista (**Arista**)**  ” Elimina la arista correspondiente en la contenedora de aristas”  {pre: TRUE}  {post: TRUE} |

|  |
| --- |
| **DarArista (**Arista**)**  ” Retorna la arista correspondiente de la contenedora de aristas, sino existe retorna null”  {pre: TRUE}  {post: TRUE} |

|  |
| --- |
| **Bfs ()**  ” Retorna la colección de elementos que especifican el recorrido completo por expansión”  {pre: TRUE}  {post: TRUE} |

|  |
| --- |
| **Dfs ()**  ” Retorna la colección de elementos que especifican el recorrido completo por profundidad”  {pre: TRUE}  {post: TRUE} |