|  |
| --- |
| **TAD GRAFO** |
| **Invariantes:**  El grafo G consiste en un conjunto de vértices (V) y un conjunto de aristas (E). Cada arista conecta dos vértices, y ambos conjuntos (V y E) son no vacíos. |
| Operaciones Primitivas:  **1.AgregarVértice(): -> Vértice**   * Descripción: Agrega un nuevo vértice al grafo. * Entrada: Ninguna. * Salida: Un nuevo vértice agregado al grafo.   **2.** **AgregarArista(v1, v2, peso): -> Arista**   * Descripción: Crea una arista que conecta dos vértices existentes en el grafo. * Entrada: Dos vértices v1 y v2, y un peso para la arista. * Salida: Una nueva arista que conecta los vértices v1 y v2 con el peso dado   **3.EliminarVértice(v):**   * Descripción: Elimina un vértice específico del grafo. * Entrada: El vértice v a ser eliminado. * Salida: Ninguna.   **4.EliminarArista(a):**   * Descripción: Elimina una arista específica del grafo. * Entrada: La arista a a ser eliminada. * Salida: Ninguna.   **5** **ObtenerVertices(): -> List<Vertice>**   * Descripción: Devuelve el conjunto de vértices presentes en el grafo. * Entrada: Ninguna. * Salida: Lista de vértices del grafo.   **6.ObtenerAristas(): -> List<Arista>**   * Descripción: Devuelve el conjunto de aristas presentes en el grafo. * Entrada: Ninguna. * Salida: Lista de aristas del grafo.   **7.ObtenerAristas(): -> List<Arista>**   * Descripción: Devuelve el conjunto de aristas presentes en el grafo. * Entrada: Ninguna. * Salida: Lista de aristas del grafo.   **Dijkstra(inicio, fin): -> List<Position>**   * Descripción: Encuentra el camino más corto desde el vértice de inicio hasta el vértice de fin utilizando el algoritmo de Dijkstra. * Entrada: Vértice de inicio y vértice de fin. * Salida: Lista de posiciones que representan el camino más corto.   **FloydWarshall(inicio, fin): -> List<Position>**   * Descripción: Encuentra el camino más corto desde el vértice de inicio hasta el vértice de fin utilizando el algoritmo de Floyd-Warshall. * Entrada: Vértice de inicio y vértice de fin. * Salida: Lista de posiciones que representan el camino más corto.   **ImprimirCamino(camino)**   * Descripción: Imprime el camino dado. * Entrada: Lista de posiciones representando el camino. * Salida: Ninguna. |