|  |
| --- |
| **TAD Grafo** |
| El grafo tiene un conjunto de vértices V = {v1, v2, …, vn}  El grafo tiene un conjunto de aristas E = {e1, e2, …, ek} |
| {inv.: }  {} (grado= arista incidente) |
| Operaciones primitivas:   |  |  |  | | --- | --- | --- | | Crear grafo: |  |  | | BFS | V vértice |  | | DFS | V vértice |  | | Dijkstra | V vértice | Los caminos (distancia y secuencia) más cortos entre un vértice y el resto | | Floyd Warshall |  | Los caminos (distancia y secuencia) más cortos para cada par de vértices | | Prim |  | Un árbol de expansión mínima | | Kruskal |  | Un bosque de espansión mínima | |

|  |
| --- |
| **DFS**(Vértice) |
| “Recorre el grafo desde un vértice siguiendo el patrón: recorrer al padre, luego a su hijo, y luego al siguiente hijo hasta llegar a una hoja. Luego, vuelve al antecesor más cercano y repite el proceso para los hijos no visitados.” |
| {pre: grafo = }  {pos: } |

|  |
| --- |
| **BFS**(Vértice) |
| “Recorre el grafo desde un vértice siguiendo el patrón: recorrer al padre, luego a los hijos del padre y luego a los hijos de los elementos recorridos. (Recorriendo por amplitud)” |
| {pre: grafo =  }  {pos: } |

|  |
| --- |
| **DIJKSTRA**(Vértice) |
| “Encuentra los caminos mínimos y sus respectivas distancias, para visitar cada vértice del grafo desde un vértice ‘x’ en específico |
| {pre: graph =  x V el grafo es ponderado}   * Sea una permutación (camino) <v1, v2, …, vi> de algún conjunto * Sea el peso de un camino c.   *{pos: C = { c1, c2, …, cn } {w1, w2, …, wi} |*  *i(wi es el peso del camino ci)}* |

|  |
| --- |
| **FloydMarshal**() |
| “Encuentra los caminos mínimos y sus respectivas distancias, entre cada par de vértices” |
| {pre: grafo =  y el grafo es ponderado}   * Sea una permutación (camino) <v1, v2, …, vi> de algún conjunto * Sea el peso de un camino c.   *{pos:*   *|*  *i (wij es el peso del camino cij)}* |

|  |
| --- |
| **Prim**() |
| “Construye un grafo que a la vez es un árbol de expansión mínima del grafo principal” |
| {pre: grafo A =  el grafo es ponderado y se compone de una sola componente conexa }  {pos: Grafo B  } |

|  |
| --- |
| **Kruskal**() |
| “Construye un grafo que a la vez es un árbol de expansión mínima del grafo principal” |
| {pre: grafo A =  el grafo es ponderado y puede tener más de una componente conexa}  {pos: G grafos } |