**UML Diagrams**

|  |
| --- |
| Class: TownGraphManager |
| Graph: GRAPH |
| TownGraphManager()  populateTownGraph(File): void  addRoad(String, String, int, String): Boolean  getRoad(String, String): String  addTown(String): Boolean  getTown(String): Town  containsTown(String): Boolean  containsRoadConnection(String, String): Boolean  allRoad(): ArrayList<String>  allTowns(): ArrayList<String>  deleteRoadConnection(String, String, String): Boolean  deleteTown(String): Boolean  getPath(String, String): ArrayList<String> |

|  |
| --- |
| **Interface:** TownGraphManagerInterface |
|  |
| TownGraphManager()  populateTownGraph(File): void  addRoad(String, String, int, String): Boolean  getRoad(String, String): String  addTown(String): Boolean  getTown(String): Town  containsTown(String): Boolean  containsRoadConnection(String, String): Boolean  allRoad(): ArrayList<String>  allTowns(): ArrayList<String>  deleteRoadConnection(String, String, String): Boolean  deleteTown(String): Boolean  getPath(String, String): ArrayList<String> |

|  |
| --- |
| Class: Graph |
| Towns: Set<Town>  Roads: Set<Road>  townMap: Map<Town, Town>  weightMap: Map<Town, Integer> |
| Graph()  getEdge(Town, Town): Road  addEdge(Town, Town, int, String): Road  addVertex(Town): Boolean  containsEdge(Town, Town): Boolean  containsVertex(Town): Boolean  edgeSet(): Set<Road>  vertexset(): Set<Town>  edgesOf(Town): Set<Town>  removeEdge(Town, Town, int, String): Road  removeVertex(Town): Boolean  shortestPath(Town, Town): ArrayList<String>  dijkstraShortestPath(Town): Void |

|  |
| --- |
| **Road** |
| roadName: String  source: Town  destination: Town  distance: int |
| Road(Town, Town, int, String)  Road(Town, Town, String)  Contains(Town): Boolean  compareTo(Road): int  equals(Object): boolean |

|  |
| --- |
| **Town** |
| townName: String |
| Town(String)  Town(Town)  compareTo(Town): int  equals(Object): Boolean  getName(): String  hashCode(): int  toString(): String |

|  |
| --- |
| **Interface:** GraphInterface |
|  |
| Graph()  getEdge(Town, Town): Road  addEdge(Town, Town, int, String): Road  addVertex(Town): Boolean  containsEdge(Town, Town): Boolean  containsVertex(Town): Boolean  edgeSet(): Set<Road>  vertexset(): Set<Town>  edgesOf(Town): Set<Town>  removeEdge(Town, Town, int, String): Road  removeVertex(Town): Boolean  shortestPath(Town, Town): ArrayList<String>  dijkstraShortestPath(Town): Void |