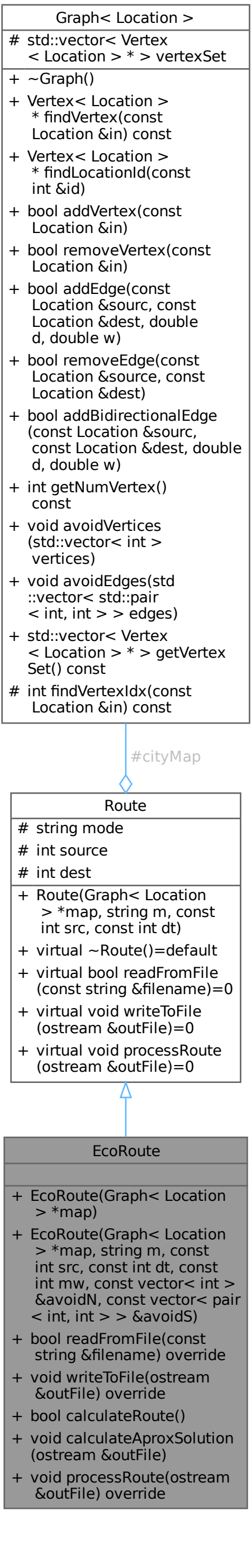
Graph< Location >
std::vector< Vertex < Location > * > vertexSet
+ ~Graph()
+ Vertex< Location > * findVertex(const Location &in) const
+ Vertex< Location > * findLocationId(const int &id)
+ bool addVertex(const Location &in)
+ bool removeVertex(const Location &in)
+ bool addEdge(const Location &sourc, const Location &dest, double d, double w)
+ bool removeEdge(const Location &source, const Location &dest)
+ bool addBidirectionalEdge (const Location &sourc, const Location &dest, double d, double w)
+ int getNumVertex() const
+ void avoidVertices (std::vector< int > vertices)
+ void avoidEdges(std ::vector< std::pair < int, int > > edges)
+ std::vector< Vertex < Location > * > getVertex Set() const
# int findVertexIdx(const Location &in) const
#cityMap
Route
string mode
int source
int dest
+ Route(Graph< Location > *map, string m, const int src, const int dt)
+ virtual ~Route()=default
+ virtual bool readFromFile (const string &filename)=0
+ virtual void writeToFile (ostream &outFile)=0
+ virtual void processRoute (ostream &outFile)=0
EcoRoute
+ EcoRoute(Graph< Location > *map)
+ EcoRoute(Graph< Location > *map, string m, const int src, const int dt, const int mw, const vector< int > &avoidN, const vector< pair < int, int > > &avoidS)
+ bool readFromFile(const string &filename) override
+ void writeToFile(ostream &outFile) override
+ bool calculateRoute()
+ void calculateAproxSolution (ostream &outFile)
+ void processRoute(ostream &outFile) override