Arianna Lezzi 947010, Fabio Giordana 886377, Samuel Gonçalves Barbosa 955864

**Scelte implementative**

Si è scelto di implementare l’algoritmo di *Dijkstra* attraverso la definizione di una classe che lavora con tipi generici (sia riguardo ai vertici del grafo che ai pesi degli archi); gli oggetti di questa classe vengono creati passando al costruttore il grafo su cui applicare l’algoritmo, la sorgente da cui partire, degli oggetti delle classi con cui effettuare operazioni sul tipo degli archi (per confrontarli e sommarli tra loro) e i valori di minimo e massimo delle distanze con cui inizializzare l’algoritmo.

Internamente gli oggetti della classe utilizzano un heap minimo per implementare l’algoritmo; all’interno dell’heap sono contenuti elementi della classe *Node* che è definita internamente alla classe *Dijkstra* e che contiene, oltre all’attributo sul vertice del grafo a cui fa riferimento, i valori di cammino minimo e del vertice genitore (quello precedente lungo il cammino minimo dalla sorgente). I nodi sono ordinati in base al loro attributo “diff” (cammino minimo).

Per decrementare i nodi nell’heap si definisce internamente una classe *NodeDecreaser* con un attributo “diff” il cui valore viene usato per modificare l’attributo cammino minimo di un nodo.

Sono state realizzate due classi esterne di supporto: *DijkstraResult* e *DijkstraIncreaser ;* *DijkstraIncreaser* è un’interfaccia usata per implementare oggetti con cui sommare i pesi degli archi del grafo, invece *DijkstraResult* è una classe generica al cui interno è presente una Hash map che associa a ogni vertice del grafo i due attributi calcolati dall’algortimo: cammino minimo e genitore del vertice, ai quali si può accedere con metodi opportuni; l’algoritmo di Dijkstra restituisce un oggetto di questa classe. Si è pensato a questa soluzione per non avere come valori di ritorno oggetti della classe Node essendo questa interna alla classe Dijkstra e perché l’hash map interna all’oggetto restituito permette di accedere alle informazioni su un vertice del grafo in tempo O(1).

La classe *GraphUsage* utilizza l’algoritmo di Dijkstra per trovare il cammino minimo tra due città, creando un oggetto della classe Dijkstra che lavora con String e Float (per questo all’interno della classe vengono anche definite le classi per comparare e sommare i float che verranno successivamente usate da Dijkstra). Il grafo da passare alla classe Dijkstra viene creato a partire da un file ‘.csv’ preso come input; il valore di ritorno di Dijkstra viene utilizzato per stampare a video la distanza tra sorgente e destinazione e il percorso da seguire lungo il grafo.