GRUPPO 3: PROGETTO 4

**ESERCIZIO 1**

**Greedy Vertex Cover**

La funzione non prende nulla in ingresso, mentre in uscita restituisce un dizionario vertexCover contenente non più del doppio dei vertici rispetto alla soluzione ottima per risolvere il problema richiesto.

Viene usato un set di appoggio E che contiene al suo interno tutti gli archi del grafo. Poiché l’algoritmo è di tipo *greedy*, si procede per passi successivi: finché il numero di archi non diviene pari a zero, ad ogni iterazione si seleziona un arco **(u,v)** di E, e si aggiungono i suoi vertici **u** e **v** al dizionario vertexCover; si eliminano dal set E tutti gli archi incidenti dei vertici **u** e **v**, per poi procedere all’iterazione successiva, finché E non è vuoto. Infine viene restituito il dizionario.

Riempire il set di appoggio prende tempo del numero di archi, mentre il while prende al massimo tempo del numero di archi in E e, per ogni vertice inserito prende tempo del grado dei vertici. Rimuovere l’arco dal set prende tempo costante. Quindi l’algoritmo complessivo ha complessità O(n + m) dove n è il numero di vertici ed m il numero di archi.

**Min Vertex Cover**

La funzione non prende nulla in ingresso e restituisce un dizionario vertexCover di dimensione minima.

**ESERCIZIO2**

**ESERCIZIO3**