Algorithm 1: Boruvka(Graph$(V, E, w)) \rightarrow$ Graph

Input: Un grafo non orientato pesato connesso con vertici V, archi E e funzione di peso w
Output: Un grafo rappresentante un albero di copertura di peso complessivo minimo per il grafo in input

```
T = new Graph
for each v ∈ V do
    aggiungi a T un vertice corrispondente a v
while T contiene più di una componente connessa do
    S = ∅                                   // S è un insieme di archi inizialmente vuoto
    for each C componente connessa in T do
        aggiungi a S l'arco con peso minimo nell'insieme { {u, v} ∈ E | u ∈ C, v ∉ C }
    for each {u, v} ∈ S do
        aggiungi a T un arco dal vertice corrispondente a u al vertice corrispondente a v di peso w(u, v)
return T
```

1 Descrizione

L'obiettivo del progetto è quello di realizzare una classe Java per la computazione del Minimum Spanning Tree in un grafo non orientato pesato connesso utilizzando l'algoritmo di Boruvka. L'algoritmo di Boruvka è probabilmente il più antico per il calcolo del Minimum Spanning Tree. Venne ideato nel 1926 dal matematico Otakar Boruvka per progettare la rete elettrica di una regione della attuale Repubblica Ceca. Lo pseudocodice dell'algoritmo è riportato in Algorithm 1.

L'algoritmo inizia considerando ogni singolo vertice come una componente connessa indipendente. Procede poi iterativamente scegliendo in modo greedy, ad ogni iterazione, l'arco che per ogni componente connessa risulta essere quello di peso minimo che collega un qualche vertice della componente ad un vertice non appartenente alla componente. Per la gestione delle componenti connesse si consiglia di utilizzare una struttura union-find; ogni componente connessa è rappresentata da un insieme disgiunto e due insiemi disgiunti vengono uniti quando l'algoritmo sceglie un arco che collega le corrispondenti componenti connesse.

Si noti inoltre che lo pseudocodice include cicli **while** e **for** che fanno controlli sul numero di componenti connesse o iterano su tali componenti, e utilizza un insieme di archi S. La vostra implementazione in Java deve seguire la logica dell'algoritmo senza dover obbligatoriamente prevedere cicli che iterano su componenti connesse o utilizzare obbligatoriamente la struttura dati *Set*. Si noti comunque che dal punto di vista della logica dell'algoritmo usare un insieme evita duplicazioni di archi che durante le scelte greedy potrebbero essere scelti due volte. Infatti, un medesimo arco non orientato $\{u, v\}$ potrebbe essere scelto sia quando si considera la componente connessa di u, sia quando si considera la componente connessa di v. L'insieme S conterrà tale arco una sola volta e quindi l'arco verrà inserito nella soluzione una sola volta.

Assieme a questo PDF è disponibile un package precostruito contenente il quantitativo minimo di file di cui avete bisogno per completare il progetto. Oltre a questo package viene fornito un file di test *ProjectTest.java* speculare a quello che verrà utilizzato durante la fase di correzione per permettere di testare l'algoritmo e guidarvi nell'implementazione delle interfacce necessarie. Vengono inoltre forniti un file *grafoMST.txt* contenente un esempio dell'input e un file *output.txt* contenente il corrispondente output atteso.

2 Consegna

Una volta completato il progetto, si richiede di comprimere in formato *.zip* la cartella *src* contenente tutto il codice e consegnarla nell'apposita sezione presente sulla pagina Virtuale di questo corso. Il file *.zip* deve essere rinominato nel formato "nome_cognome_matricola.zip". Come anticipato avete a disposizione una consegna circa una settimana prima di ognuno degli appelli d'esame della sessione giugno/settembre.

3 Valutazione

Il progetto viene valutato al più 3 punti e vengono presi in considerazione i seguenti criteri:

- Correttezza dell'algoritmo;
- Efficienza;
- Codice ben strutturato e commentato.

4 FAQ

- Posso modificare il file di test? No;
- Posso creare nuovi file, nuove classi o interfacce? Sì;
- Posso modificare le interfacce presenti? Sì a patto che la scelta sia motivata nel file;
- Posso utilizzare librerie esterne? Sono consentite solo le strutture dati presenti nel package Java.util (standard library Java).
- Devo fare prima il progetto o l'esame scritto? Il progetto va consegnato obbligatoriamente prima dell'esame scritto;
- Posso consegnare il progetto e fare lo scritto più avanti? Sì;
- Dove verranno publicati i voti del progetto? I voti del progetto verranno pubblicati su Virtuale, mentre su AlmaEsami verrà pubblicata una proposta di voto comprensiva della valutazione dello scritto e del progetto.
- Posso consegnare il progetto più volte? Una volta che è stato consegnato un progetto valido a questo verrà assegnato un voto e lo studente non potrà effettuare altre consegne negli appelli successivi.

The End.