

# Seconda prova pratica in itinere

## Ingegneria degli Algoritmi 2018/2019

Ovidiu Daniel Barba  
ovidiodaniel.barba@alumni.uniroma2.eu

Luca Pepè Sciarria  
luca.pepesciarria@alumni.uniroma2.eu

7 Gennaio 2019

## 1 Modalità di svolgimento

Il progetto può essere svolto singolarmente oppure in gruppi da massimo 3 persone (consigliato gruppo di 2 persone).

È altamente **sconsigliato** di copiare codice reperito on-line o da altri elaborati e di usare librerie esterne, poichè provoca l'annullamento della consegna con conseguente esclusione dalle prove in itinere.

È invece **consigliato** di usare e manipolare il codice visto a lezione e disponibile sul [sito](#) della parte pratica del corso.

## 2 Scelta del progetto

Una volta individuato il gruppo, definiamo  $s$  come la somma delle matricole dei componenti del gruppo. Il progetto assegnato è dato dal valore di  $s \bmod 2 + 1$ . Per esempio, se il gruppo è composta da 2 persone con matricola 0244962 e 0246425,  $s = 244962 + 246425 = 491387$  e il progetto assegnato è il secondo poichè  $s \bmod 2 + 1 = 491387 \bmod 2 + 1 = 1 + 1 = 2$ .

### 3 Materiale da consegnare

- Breve relazione in formato pdf contenente:
  - scelte implementative
  - risultati sperimentali sia in forma tabulare che grafica con unità di misura appropriate
  - commento dei risultati ottenuti
- Codice sorgente:
  - implementazione richiesta nella traccia
  - esempio di uso dell'algoritmo o struttura dati implementata
  - esperimenti effettuati

### 4 Consegna

Tutto il materiale richiesto dovrà essere caricato su un repository online (ad esempio GitHub, Dropbox, Google Drive, ecc), il cui link dovrà essere inviato per email a entrambi i tutor. Includere nella mail nome, cognome, matricola ed email di tutti i componenti del gruppo.

### 5 Progetto

Implementare l'algoritmo `distanzeAciclico(grafo  $G$ , vertice  $s$ )` (capitolo 14 del libro di testo) per il calcolo delle distanze a partire da una sorgente  $s$  in un grafo orientato aciclico  $G$ , indipendentemente dalla sua implementazione (liste di adiacenza, matrice di adiacenza, ecc). L'algoritmo usa una variante dell'ordinamento topologico in cui i gradi entranti dei nodi vengono mantenuti e aggiornati in una coda con priorità  $S$ . Tale coda viene usata per identificare i nodi senza archi entranti. Analizzare il tempo di esecuzione di `distanzeAciclico` con diversi tipi di code con priorità:

- $d$ -heap con  $d \in \{3, 5, 7\}$
- heap binari
- heap binomiali

Confrontare tali tempi di esecuzione con quello dell'algoritmo di *Bellman e Ford* al variare della grandezza del grafo di input. Inoltre, implementare un algoritmo per la generazione di grafi aciclici con pesi random per usarlo nella fase di confronto delle performance.