

Prova finale di algoritmi e strutture dati

Obiettivi didattici e realizzazione

- Obiettivi
 - Applicazione pratica delle tecniche apprese nel modulo di algoritmi e strutture dati del corso di algoritmi e principi dell'informatica
 - Implementazione di una soluzione ad un problema prestando attenzione ad aspetti concreti di efficienza del codice
- Realizzazione
 - Linguaggio C (C11, VLA ammessi)
 - Nessuna libreria esterna al di là della libreria standard C
 - No multithreading
 - Dati in ingresso ricevuti via stdin, risultati da fornire via stdout

Modalità di realizzazione

- Il progetto è **strettamente individuale**
 - Non utilizzate **alcun frammento di codice altrui**
- Siete responsabili del vostro codice
 - Non caricatelo su repository pubblici
 - Non condividetelo con colleghi per "prendere ispirazione"
 - Non utilizzate alcun frammento di codice reperito
- In caso di plagio o uso di codice altrui, **tutti i progetti coinvolti** saranno annullati

Criteri di valutazione

- La correttezza e l'efficienza della soluzione proposta sono valutate con batterie di test automatizzate
- Verranno forniti input/output d'esempio per poter collaudare la soluzione in locale
 - Non sottoponete soluzioni senza aver verificato che funzionino localmente
 - Verrà fornito anche uno strumento di generazione automatica di casi di test (input/output), per facilitarvi il testing in locale
- Il sistema di verifica calcola il tempo macchina e la memoria utilizzati
- La valutazione è immediatamente calcolata (e subito visibile), mediante 3 batterie di test:
 - la prima vale 18 punti (*pass or fail*)
 - la seconda fino a 12 (6 test da 2 punti ognuno)
 - l'ultima per la lode

Criteri di valutazione

- Nessun limite al numero di sottoposizioni, né penalità per sottoposizioni multiple
- È possibile migliorare la valutazione quante volte si desidera
- **Avvertenza:** viene valutata l'ultima sottoposizione fatta ad ogni batteria di test. Tutte le sottoposizioni valutate devono utilizzare lo stesso sorgente
 - Se siete in dubbio, ri-sottoponete lo stesso sorgente a tutte le batterie di test per buona misura
- Verificatore disponibile al <https://dum-e.deib.polimi.it>
- Credenziali di accesso via mail polimi

Scadenze e pianificazione

- Per i laureandi di luglio
 - 4 luglio, ore 23.59 CEST. Segnalate (email al docente) la necessità di valutazione
- Per tutti gli altri
 - 7 settembre, ore 23.59 CEST, dopo di che la piattaforma verrà **chiusa**
- Per laureandi di gennaio/febbraio (SUPERATO 145 CFU (di qualsiasi genere) + essere iscritto all'esame di laurea)
 - la piattaforma sarà riaperta indicativamente tra il 31 gennaio 2022 e l'11 febbraio 2022, fino alle ore 23.59 CEST
- Iniziare a lavorare ad una settimana dalla scadenza è uno dei modi migliori per **non** riuscire a superare la prova

Tutoraggio

- Sezione A-D (Barenghi):
 - Amedeo Cavallo (amedeo.cavallo@mail.polimi.it) (A -- Canestraci)
 - Mariarosaria Caterino (mariarosaria.caterino@mail.polimi.it) (Cangemi-- E)
- Sezione E-O (Pradella):
 - Valentina Deda (valentina.deda@mail.polimi.it) (E -- Mazzola)
 - Vittorio Torri (vittorio.torri@mail.polimi.it) (Mazzone -- O)
- Sezione P-Z (Martinenghi):
 - Moreno Di Berardino (moreno.diberardino@mail.polimi.it) (P -- Sarno)
 - Vittorio Torri (vittorio.torri@mail.polimi.it) (Sarrocco -- Z)

GraphRanker

- L'obiettivo del progetto di quest'anno è la gestione di una classifica tra grafi diretti pesati
 - La classifica tiene traccia dei k "migliori" grafi
- Il programma da realizzare riceve in ingresso
 - due parametri, una sola volta (sulla prima riga del file, separati da spazio)
 - d: il numero di nodi dei grafi
 - k: la lunghezza della classifica
 - Una sequenza di comandi tra
 - `AggiungiGrafo [matrice-di-adiacenza]`
 - `TopK`

AggiungiGrafo

Richiede di aggiungere un grafo a quelli considerati per stilare la classifica. È seguito dalla matrice di adiacenza del grafo stesso, stampata una riga per ogni rigo, con gli elementi separati da virgole.

I nodi del grafo sono da considerarsi etichettati logicamente con un indice intero tra 0 e $d-1$; il nodo in posizione 0 è quello la cui stella uscente è descritta dalla prima riga della matrice.

I pesi degli archi del grafo elementi sono interi nell'intervallo $[0, 2^{32} - 1]$.

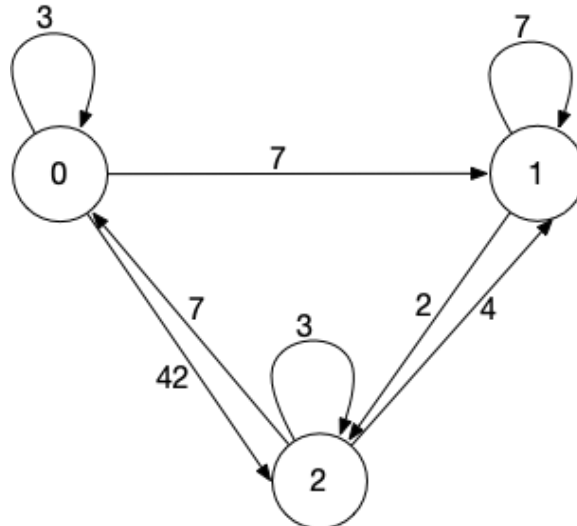
- Esempio per $d=3$

AggiungiGrafo

3, 7, 42

0, 7, 2

7, 4, 3



TopK

- Si consideri ogni grafo dall'inizio del programma fino al comando TopK etichettato con un indice intero corrispondente al numero di grafi letti prima di esso (partendo da 0)
- TopK richiede al programma di stampare gli indici interi dei k grafi aventi i k valori più piccoli della seguente metrica:
 - Somma dei cammini più brevi tra il nodo 0 e tutti gli altri nodi del grafo
- Se ci sono più grafi con lo stesso valore della metrica, si dà la precedenza ai primi arrivati
- I k indici interi sono stampati, su un unico rigo, separati da uno spazio, in un qualunque ordine

Un'esecuzione d'esempio

Input ricevuto

3,2 (spazio non virgola)

AggiungiGrafo

0,4,3

0,2,0

2,0,0

AggiungiGrafo

0,0,2

7,0,4

0,1,0

AggiungiGrafo

3,1,8

0,0,5

0,9,0

TopK

Commenti e Output Atteso

Si richiede di manipolare grafi da 3 nodi e riportare i k=2 migliori
Aggiunta del primo grafo (indice 0, somma cammini = 7)

Aggiunta del secondo grafo (indice 1, somma cammini = 5)

Aggiunta del terzo grafo (indice 2, somma cammini = 7)

0 1 Oppure 1 0