



## Esercitazione di laboratorio n. 8

### Esercizio n.1: Corpo libero (vers. greedy)

A partire dallo scenario introdotto nell'esercizio 2 del Laboratorio 7, si risolva il problema proponendo uno o più algoritmi greedy, definendo opportune funzioni obiettivo.

### Esercizio n.2: Rete di elaboratori

Un grafo non orientato e pesato rappresenta una rete di elaboratori appartenenti ciascuno ad una sottorete. Il peso associato ad ogni arco rappresenta il flusso di dati tra due elaboratori della stessa sottorete o di sottoreti diverse, come nell'esempio seguente (cfr. figura successiva).

Il grafo è contenuto in un file, il cui nome è passato come argomento sulla linea di comando. Il file è composto da un numero indefinito di righe ciascuna delle quali contiene una quaterna di stringhe alfanumeriche, di al massimo 30 caratteri, e un intero:

ind

```
<id_elab1> <id_rete1> <id_elab2> <id_rete2> <flusso>
```

Si facciano anche le seguenti assunzioni:

- i nomi dei singoli nodi sono univoci all'interno del grafo
- non sono ammessi cappi
- tra due nodi c'è al massimo un arco (non è un multigrafo)
- le sotto-reti sono sotto-grafi non necessariamente connessi.

Si scriva un programma in C in grado di caricare in memoria il grafo, leggendone i contenuti da file e di potervi effettuare alcune semplici operazioni.

La rappresentazione della struttura dati in memoria deve essere fatta tenendo conto dei seguenti vincoli:

- il grafo sia implementato come ADT di I classe, predisposto in modo tale da poter contenere sia la matrice sia le liste di adiacenza. Nella fase di caricamento dei dati da file si generi solamente la matrice di adiacenza, su comando esplicito va generata anche la lista di adiacenza
- si utilizzi una tabella di simboli tale da fornire corrispondenze "da nome a indice" e "da indice a nome".

Sul grafo, una volta acquisito da file, sia possibile:

- elencare in ordine alfabetico i vertici e per ogni vertice gli archi che su di esso insistono, sempre in ordine alfabetico
- dati 3 vertici i cui nomi sono letti da tastiera, verificare se essi sono adiacenti a coppie, cioè se formano un sottografo completo. Tale funzione sia implementata sia per la rappresentazione con matrice delle adiacenze, sia per la rappresentazione con lista delle adiacenze
- generare la rappresentazione a lista di adiacenza, SENZA leggere nuovamente il file, a partire da quella a matrice di adiacenza.

In allegato al testo è presente il grafo d'esempio (grafo.txt) rappresentato a seguire:





Politecnico  
di Torino

03AAX ALGORITMI E STRUTTURE DATI CORSO DI LAUREA IN INGEGNERIA INFORMATICA A.A. 2022/23

$$Q_i = \frac{\sum_j v_{ij} \cdot n_{ij}}{\sum_j n_{ij}}$$

Si noti che, indentificando con  $j$  la  $j$ -esima transazione del titolo alla data  $i$ ,  $v_{ij}$  è il valore del titolo nella transazione, mentre  $n_{ij}$  è la quantità di titoli scambiati in tale transazione.

Si realizzino:

- un quasi ADT per la data e uno per l'ora (o un unico ADT per entrambe), implementati come `struct` con campi interi per anno, mese e giorno, ore e minuti
- un ADT di I classe per il titolo e uno per collezione di titoli (a scelta se in un solo modulo o in due). Per la collezione di titoli si faccia uso di una lista ordinata (si usi il codice del titolo come chiave di ordinamento)
- un quasi ADT per la quotazione giornaliera e un ADT di I classe per collezione di quotazioni giornaliere (a scelta se in un solo modulo o in due). Per la collezione di quotazioni si faccia uso di un BST (con data come chiave di ricerca e ordinamento). Per gli inserimenti di dati nel BST sono sufficienti gli inserimenti in foglia
- un client che fornisca le seguenti funzionalità:
  1. acquisizione del contenuto di un file contenente un insieme di transazioni
  2. ricerca di un titolo azionario (ricerca in lista)
  3. ricerca, dato un titolo precedentemente selezionato, della sua quotazione in una certa data (ricerca in un BST)
  4. ricerca, dato un titolo precedentemente selezionato, della sua quotazione minima e massima in un certo intervallo di date (si noti che la ricerca, in un BST, di più chiavi comprese in un dato intervallo, non è una funzione standard e va quindi realizzata: si consiglia una variante di un algoritmo di visita in-order)
  5. ricerca, dato un titolo precedentemente selezionato, della quotazione minima e massima lungo tutto il periodo registrato (il problema può essere ricondotto a un caso particolare del punto precedente)
  6. dato un titolo precedentemente selezionato, bilanciamento dell'albero di quotazioni se il rapporto tra il cammino più lungo e più corto nell'albero supera una certa soglia  $S$ .

*Nota: si ricorda che è possibile bilanciare un dato BST mediante applicazione ricorsiva del partizionamento rispetto alla chiave mediana.*

Parte che non  
clera all'esame  
PARTE PIÙ COMPLESSA

**Valutazione: tutti gli esercizi saranno oggetto di valutazione**

**Scadenza caricamento di quanto valutato: entro le 23:59 del 20/01/2023.**