Corso di Algoritmi e Strutture Dati
Esercitazione 6: ancora su alberi

Si consideri il tipo di dato `Albero` implementato nell'esercitazione 3. Lo si estenda e, se necessario, lo si modifichi (ma senza cambiare le interfacce delle operazioni definite per l'esercitazione 3, che dovranno continuare a funzionare come prima) implementando le seguenti operazioni aggiuntive:

1. Ricerca del minimo antenato comune a due nodi identificati dalle proprie etichette. La funzione restituisce un booleano (`false` se e solo se una delle due etichette non è nell'albero) mentre l'etichetta risultato è un parametro OUT (se viene ritornato `false` allora tale etichetta non è significativa).

2. Cancellazione di un nodo dall'albero data l'etichetta che lo identifica. Se il nodo è una foglia va semplicemente cancellato; se il nodo è intermedio, va cancellato e i suoi figli vanno collegati al padre; in entrambi i casi la funzione restituisce `true`. Se l'etichetta è quella del nodo radice e la radice ha figli, oppure se l'etichetta non è presente nell'albero, tale cancellazione evidentemente non si può fare e dunque la funzione restituisce `false`.

3. Conteggio del numero di nodi presenti nell'albero. La complessità di questa operazione deve essere la minima possibile (anche a costo di apportare opportune modifiche allo `Albero` della esercitazione 3).

4. "Visualizzazione strutturata": si tratta di visualizzare il contenuto dell'albero, mediante visita in ampiezza oppure in profondità, ma in modo da offrire le informazioni necessarie a ricostruirlo; dunque non basta visualizzare una dopo l'altra le etichette dei nodi, ma occorre dare informazioni sulla struttura ramificata dell'albero. Se la visita è condotta in ampiezza, un formato possibile per l'output è quello usato per l'input. Se la visita è condotta in profondità, un formato possibile per l'output è quello in cui si usa l'indentazione per dare informazioni su quali nodi siano i figli di chi, come nell'esempio seguente riferito all'esercitazione 3:

```
mammalia
 primates
  anthropoidea
   catarrhini
    hominoidea
     hominidae
      homo
       homoAbilis
       homoErectus
       homoSapiens
        homoSapiensNeanderthaliensis
        homoSapiensSapiens
     pongidae
      pongo
      gorilla
      pan
     hylobatidae
```

```
    cercopithecoidea
  platyrrhini
prosimii
```

La *struct* C++ così estesa deve essere impiegate in un semplice programma C++. Tale programma deve svolgere, nell'ordine, le azioni seguenti:

1. Creazione dell'albero acquisendo da standard input le informazioni che caratterizzano i nodi (le stringhe di caratteri).
2. Un ciclo infinito che consenta la scelta ripetuta di una tra le seguenti opzioni:
   - Ricerca del minimo antenato comune a due nodi le cui etichette sono acquisite da standard input.
   - Cancellazione di un nodo la cui etichetta è acquisita da standard input.
   - Visualizzazione strutturata dell'albero al fine di mostrare che la cancellazione è avvenuta nel modo corretto.
   - Uscita dal programma.