008AA – ALGORITMICA E LABORATORIO

Appello dell'1 febbraio 2016

Cognome Nome: **N. Matricola:** **Corso: A B**

Esercizio 1. *(10 punti)*
Un nodo v in un albero binario si dice **0-bilanciato** se le altezze dei sottoalberi radicati nei suoi due figli sono uguali. Dato un albero binario, progettare un algoritmo *efficiente* che stampi le chiavi di tutti e soli i nodi 0-bilanciati e analizzarne la complessità.

Soluzione.
L'algoritmo restituisce l'altezza del sottoalbero radicato in u, e contemporaneamente stampa i nodi 0-bilanciati.

```
StampaNodiBilanciati(u)
    if (u == null) return -1;
    hSx = StampaNodiBilanciati(u.sx);
    hDx = StampaNodiBilanciati(u.dx);
    if (hSx == hDx) stampa (u.key)   // il nodo u è 0-bilanciato
    h = 1 + max{hSx, hDx};
    return h
```

L'algoritmo esegue una visita dell'albero, la sua complessità in tempo è $T(n) = O(n)$.

Esercizio 2. *(10 punti)*
È dato un array a di n interi non necessariamente distinti. Diciamo che un elemento è di maggioranza se appare in a almeno $\lceil n/2 \rceil$ volte. Progettare un algoritmo di complessità

1. $O(n^2)$
2. $\Theta(n \log n)$

che stabilisca se a contiene un elemento di maggioranza, e in caso affermativo lo restituisca.

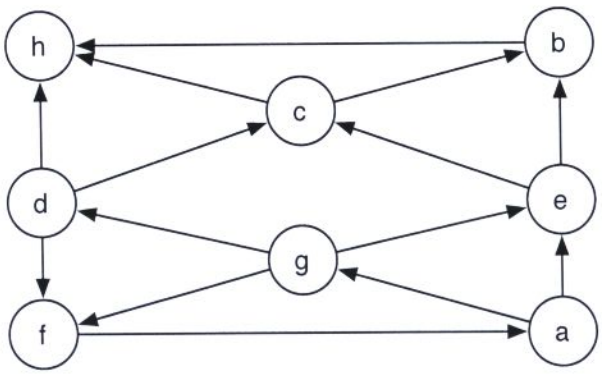
Soluzione.

1)
```
Maggioranza(a)
for i = 1 to n {
    frequenza = 1;
    for j = i+1 to n {
        if (a[i] == a[j]) frequenza++;
        if (frequenza ≥ ⌈n/2⌉) return <TROVATO: a[i]>
    }
}
return <NON TROVATO>;
```

2) Maggioranza(a)

```
if (n == 1) return <TROVATO : a[1]>;
HeapSort(a);
i = 1;
while (i ≤ n/2) {
  if (a[i] == a[i + ⌈n/2⌉ - 1]) return <TROVATO : a[i]>
  else i++;
}
return <NON TROVATO>.
```

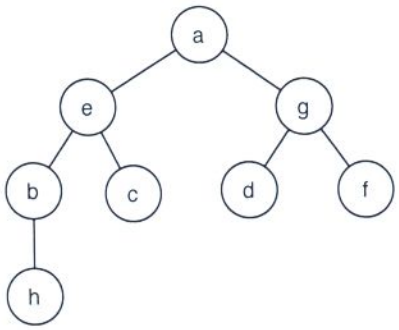
Esercizio 3. *(8 punti)*
È dato il seguente grafo orientato, rappresentato con liste di adiacenza ordinate alfabeticamente:



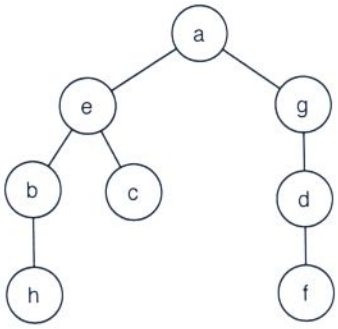
1. Indicare l'ordine di visita BFS e DFS dei vertici del grafo, partendo dal vertice *a*.
2. Disegnare gli alberi BFS e DFS ottenuti con le visite.
3. Indicare la classificazione degli archi indotta dalla visita DFS.

Soluzione.

1. **visita BFS**: a, e, g, b, c, d, f, h
   visita DFS: a, e, b, h, c, g, d, f
2.



Albero BFS



Albero DFS

3. **Archi dell'albero**: (a, e), (a, g), (e, b), (e, c), (b, h), (g, d), (d, f)
   Archi all'indietro: (f, a)
   Archi in avanti: (g, f)
   Archi trasversali a sinistra: (c, b), (c, h), (d, c), (d, h), (g, e)

Esercizio 4. *(4 punti)*
Dimostrare per induzione che il numero di foglie L_i di un albero di Fibonacci di altezza i è uguale a F_{i+1}, dove F_i l'i-esimo numero di Fibonacci.

SOLUZIONE

BASE

$i = 0 \qquad L_0 = 1 = F_1$ ✓

$i = 1 \qquad L_1 = 1 = F_2$ ✓

IPOTESI INDUTTIVA

$L_j = F_{j+1} \qquad j < i$

PASSO

$$L_i = L_{i-1} + L_{i-2} \underset{\text{i.i}}{=} F_i + F_{i-1} = F_{i+1}$$

□