Algoritmi e Strutture Dati (modulo I) - testo prova scritta 12/09/2023

docenti: Luciano Gualà & Andrea Clementi

Cognome:........................... Nome:............................... Matr.:...............

Esercizio 1 [16 punti]

A: notazione asintotica. Dire quali delle seguenti relazioni asintotiche sono vere:

V $n + n\sqrt{n}\log^2 n = o(n^{1.8})$; V $\log^3 n = o(\sqrt[4]{n})$; F $n = \Omega(\frac{n}{\log\log\log n})$; V $\frac{n^{1.5}\sqrt{n+\log n}}{\sqrt{n^3+3}} = \Theta(\sqrt{n})$;

V $(\frac{7}{3})^n = \omega(2^n)$; F $2^n = \Theta(2^n \log\log n)$; F $2^n = \omega(2^n + n^2)$; V $2^{n+8} = \Theta(2^{n-8})$;

B: equazioni di ricorrenza. Fornire la soluzione asintotica alle seguenti relazioni di ricorrenza:

$T(n) = 4\,T(\frac{n}{16}) + n^2$; Soluzione: $\Theta(m^2)$

$T(n) = T(\sqrt{n}) + 1$; Soluzione: $O(\log(\log(m)))$

C: algoritmi e complessità. Quale algoritmo useresti e quanto costa se devi:

- Dato un grafo diretto G, stabilire se tutti i nodi possono raggiungere un nodo specifico t: BFS

- In un grafo non orientato, completo e pesato, calcolare l'albero dei cammini minimi con sorgente s:

  DJKSTRA

$K = n^2$

- Ordinare un vettore di n interi compresi fra n e n^2: Radix $T(n) = O\left(n \frac{\log K}{\log n}\right)$

  $O\left(n \frac{2\log n}{\log n}\right) = O(n)$

- Fondere due alberi AVL, uno contenente n nodi e l'altro $\log n$ nodi:

  $\log(n)$ insert => $\log^2(n)$

Esercizio 2 [8 punti]

Sia T un albero binario con n nodi. Si progetti un algoritmo che dato T e due interi h_1 e h_2, con $h_1 \leq h_2$, restituisca il numero di nodi non foglia di T che hanno profondità h tale che $h_1 \leq h \leq h_2$.[1]

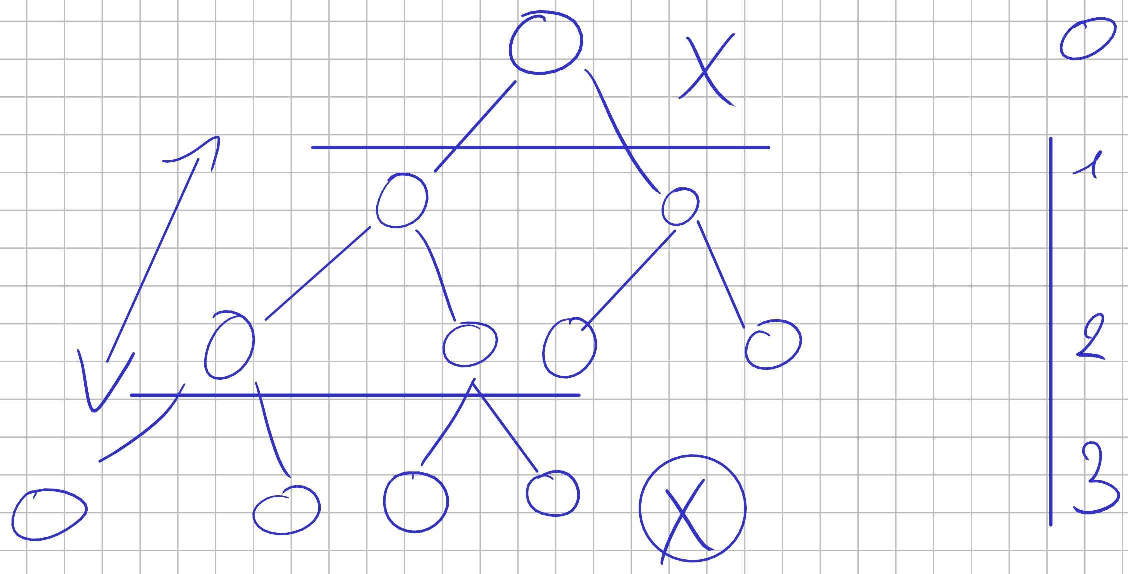
Si assuma che T è rappresentato tramite una struttura dati collegata, con record e puntatori, dove il record di ogni nodo contiene il puntatore al figlio sinistro e al figlio destro del nodo. L'algoritmo deve avere complessità $O(n)$. Si fornisca lo pseudocodice dettagliato.

Esercizio 3 [8 punti]

Sia $A[1:n]$ un vettore di n interi positivi. Diremo che un elemento $A[i]$ è *felice al quadrato* se esiste un indice j tale che $A[j] = A[i]^2$.

Si progetti un algoritmo che dato A dica in tempo $O(n \log n)$ se esiste almeno un elemento felice al quadrato. Si fornisca lo pseudocodice dettagliato.

[1]Si ricordi che la profondità di un nodo è la sua distanza (misurata in numero di archi) dalla radice.



$$h_1 \le h \le h_2$$

$h_1 = 1, \quad h_2 = 3$

```
CalcoloP(nodo v, h1, h2, h)
    if (v == null) return 0
    if (v.sin == NULL AND v.des == NULL) return 0
            v è foglia
    Sx = CalcoloP(sin(v), h1, h2, h+1)
    dx = CalcoloP(des(v), h1, h2, h+1)
    if h1 ≤ h ≤ h2
        return 1 + Sx + dx
    else return Sx + dx
```

1	2	3	4	5	6	7	8	9	10	11
2	6	4	36	8	10	3	9	11	24	6

$A[1]$ è felice $\Rightarrow A[3] = A[1]^2$

$A[2]$ è felice $\Rightarrow A[4] = A[2]^2$

Ordino integer Sort $O(n+k)$

2	4	6	6	8	9	10	11	24	36
1	2	3	4	5	6	7	8	9	10

Per $\forall i \in 1 \ldots n$ ricerca binaria

```
felice(A)
  integerSort(A)
  for i = 1 to n do
        k = A[i]^2
        j = ricercabinaria(A, 1, n, k)
        if (j ≠ -1)
            return 1
  return 0
```