Cognome: **Nome**: **# Matricola**: **Riga**: **Col**:

Algoritmi e Strutture Dati – 05/11/15

Esercizio 1 – Punti ≥ 6 (Parte A)

1. Utilizzando *un qualunque metodo*, trovare i limiti superiore e inferiore per la seguente ricorrenza (assumendo che $0 < \beta < 1$):

$$T(n) = \begin{cases} T(\lfloor \beta n \rfloor) + n^{\beta} & n > 1 \\ 1 & n \leq 1 \end{cases}$$

2. Utilizzando *il metodo di sostituzione*, detto anche per tentativi, trovare i limiti superiore e inferiore per la seguente ricorrenza:

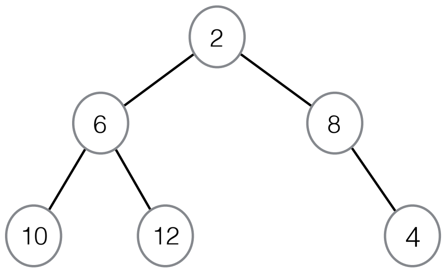
$$T(n) = \begin{cases} T\left(\frac{1}{2}n\right) + T\left(\frac{4}{5}n\right) + T\left(\frac{3}{10}n\right) + n^2 & n > 1 \\ 1 & n \leq 1 \end{cases}$$

Esercizio 2 – Punti ≥ 6 (Parte A)

Si consideri un albero binario T i cui nodi contengono chiavi intere. Scrivere un algoritmo efficiente che, quando invocato su un nodo t di T calcola la differenza tra il valore massimo e il valore minimo contenuti nelle chiavi del sottoalbero radicato nel nodo t. Vi ricordo che le chiavi sono contenute nel campo *key* della struttura dati Tree.

Discutere informalmente la correttezza della soluzione proposta e calcolare la complessità computazionale.

Esempio: dato questo albero come input e applicando l'algoritmo alla radice, l'output deve essere pari a: $12 - 2 = 10$.



Esercizio 3 – Punti ≥ 9 (Parte A)

Sia $G = (V, E)$ un grafo **orientato** e siano s, t, u tre vertici distinti in V.

1. Scrivere un algoritmo che restituisca **true** se **ogni** cammino da s a t passa per u, **false** altrimenti (se non esiste alcun cammino da s a t, si ritorni **true**).

2. Scrivere un algoritmo che restituisca **true** se **esiste** un cammino da s a t che passa per u, **false** altrimenti.

Discutere informalmente la correttezza della soluzione proposta e calcolare la complessità computazionale.

Esercizio 4 – Punti ≥ 9 (Parte A)

Sia dato in input un vettore **ordinato** di interi A, contenente n valori presi da un insieme composto da **solo due possibili valori**: $\{a, b\}$, con $a < b$. Essendo ordinato, tutti i valori a si trovano nel vettore prima dei valori b. Si assuma che nel vettore sia presente almeno un valore a e almeno un valore b. Scrivere un algoritmo efficiente che restituisca il numero di occorrenze del valore più frequente contenuto nel vettore. Soluzioni con complessità $O(n)$ non verrano considerate.

Discutere informalmente la correttezza della soluzione proposta e calcolare la complessità computazionale.

Esempio:

- Input: $A = [1, 1, 1, 1, 1, 1, 2, 2, 2, 2]$, $n = 10$
- Output: 6, in quanto 1 appare 6 volte e 2 solo 4.

Cognome: **Nome:** **# Matricola:** **Riga:** **Col:**

Algoritmi e Strutture Dati – 05/11/15

Esercizio 0 Scrivere correttamente nome, cognome, numero di matricola, riga e colonna.

Esercizio 1 – Punti ≥ 6 (Parte A)

1. Utilizzando *un qualunque metodo*, trovare i limiti superiore e inferiore per la seguente ricorrenza (assumendo che $0 < \beta < 1$):

$$T(n) = \begin{cases} \left\lfloor \frac{1}{\beta} \right\rfloor T(\lfloor \beta n \rfloor) + n^{\beta} & n > 1 \\ 1 & n \leq 1 \end{cases}$$

2. Utilizzando **il metodo di sostituzione**, detto anche per tentativi, trovare i limiti superiore e inferiore per la seguente ricorrenza:

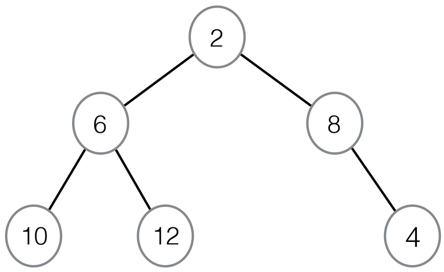
$$T(n) = \begin{cases} T\left(\frac{1}{2}n\right) + T\left(\frac{2}{5}n\right) + T\left(\frac{7}{10}n\right) + n^2 & n > 1 \\ 1 & n \leq 1 \end{cases}$$

Esercizio 2 – Punti ≥ 6 (Parte A)

Si consideri un albero binario T i cui nodi contengono chiavi intere. Scrivere un algoritmo efficiente che, quando invocato su un nodo u di T calcola la media dei valori delle chiavi presenti nel sottoalbero radicato nel nodo u, cioè la somma dei valori delle chiavi diviso il numero di nodi del sottoalbero. Vi ricordo che le chiavi sono contenute nel campo *key* della struttura dati TREE.

Discutere informalmente la correttezza della soluzione proposta e calcolare la complessità computazionale.

Esempio: dato questo albero come input e applicando l'algoritmo alla radice, l'output deve essere pari a: $\frac{2+6+8+10+12+4}{6} = \frac{42}{6} = 7$.



Esercizio 3 – Punti ≥ 9 (Parte A)

Sia $G = (V, E)$ un grafo **orientato** e siano s, t, u tre vertici distinti in V.

1. Scrivere un algoritmo che restituisca **true** se *ogni* cammino da s a t passa per u, **false** altrimenti (se non esiste alcun cammino da s a t, si ritorni **true**).

2. Scrivere un algoritmo che restituisca **true** se *esiste* un cammino da s a t che passa per u, **false** altrimenti.

Discutere informalmente la correttezza della soluzione proposta e calcolare la complessità computazionale.

Esercizio 4 – Punti ≥ 9 (Parte A)

Sia dato in input un vettore **ordinato** di interi A, contenente n valori presi da un insieme composto da solo due possibili valori: $\{a, b\}$, con $a < b$. Essendo ordinato, tutti i valori a si trovano nel vettore prima dei valori b. Si assuma che nel vettore sia presente almeno un valore a e almeno un valore b. Scrivere un algoritmo efficiente che restituisca la somma di tutti i valori contenuti nel vettore A. Soluzioni con complessità $\Omega(n)$ non verrano considerate. Discuterne correttezza e complessità.

Discutere informalmente la correttezza della soluzione proposta e calcolare la complessità computazionale.

Esempio:

- Input: $A = [1, 1, 1, 1, 1, 2, 2, 2, 2, 2]$, $n = 10$
- Output: 15