Algoritmi e Strutture Dati
a.a. 2011/2012
Appello 12 Gennaio 2013

tempo a disposizione: 3:00 h
giustificare tutte le risposte
consegnare solo la 'bella'

Esercizio 1 [5pt]
Si risolvano le seguenti equazioni di ricorrenza supponendo che il caso base sia $O(1)$:
- $A(n) = 3\,A(n/2) + log(n^3)$.
- $B(n) = B(n/2) + 2\,B(n/4) + n$.
- $C(n) = C(n/2) + log(\sqrt[3]{n})$.

Esercizio 2 [5pt]
Si consideri il seguente algoritmo.

Algorithm 1: `shuffle(array` $A[1..n]$ `di int, int` i`, int` j`)`

```
1 if i < j then
2     for int k := i + 1 to j do
3         if A[i] > A[k] then
4             swap(A[i], A[k])
5     shuffle(A, i + 1, j)
```

La procedura ausiliaria `swap(int` a`, int` b`)` scambia i valori delle due variabili a e b.
L'algoritmo viene inizialmente invocato con `shuffle`$(A, 1, n)$.

a. Scrivere la relazione di ricorrenza che descrive il costo computazionale di `shuffle` in funzione di n.

b. Risolvere la ricorrenza di cui al punto **a**.

c. Quale è l'effetto della chiamata `shuffle`$(A, 1, n)$? Motivare la risposta.

Esercizio 3 [5pt]
Simulare l'esecuzione dell'algoritmo `heapSort` sul seguente array: $[2, 4, 8, 9, 5, 11, 7, 6, 3, 10]$. Mostrare come varia la disposizione degli elementi dell'array nelle varie iterazioni dell'algoritmo. (Se lo trovate più comodo, potete raffigurare l'array direttamente come uno heap binario).

Esercizio 4 [5pt]
a. Scrivere la funzione ricorsiva `printGreaterKeys`(u, a) che, data la radice u di un BST, un valore a di chiave, stampa in ordine tutte le chiavi dell'albero maggiori della chiave data. Calcolare la complessià della soluzione proposta.
b. Scrivere la funzione ricorsiva `countGreaterKeys`(u, a) che, data la radice u di un BST, un valore a di chiave, conta quante sono le chiavi dell'albero maggiori della chiave data. La procedura deve avere costo di esecuzione $O(h)$ (dove h è l'altezza dell'albero considerato) e può servirsi di un campo ausiliario da aggiungere ai nodi .
c. Modificare le procedure di inserimento e cancellazione degli alberi BST in modo da mantenere anche il nuovo campo introdotto al punto **b.** Si assuma che la chiave da cancellare sia effettivamente presente nell'albero e che venga cancellata una sola chiave nel caso di chiavi ripetute. Entrambe le operazioni devono mantenere costo di esecuzione $O(h)$ (dove h è l'altezza dell'albero considerato).

Esercizio 5 [5pt]
La seguente tabella fornisce le distanze (in unità di 100 miglia) tra gli aeroporti delle città di Londra, Città del Messico, New York, Parigi, Pechino e Tokyo:

	L	CM	NY	Pa	Pe	T
L	–	56	35	2	51	60
CM	56	–	21	57	78	70
NY	35	21	–	36	68	68
Pa	2	57	36	–	51	61
Pe	51	78	68	51	–	13
T	60	70	68	61	13	–

Utilizzando l'algoritmo di Prim, determinare un albero di copertura minimo per il grafo corrispondente. Riportare vari passaggi dell'algoritmo.

Esercizio 6 [5pt]
Scrivere l'algoritmo di Dijkstra per determinare i cammini minimi in un grafo orientato, dimostrarne la correttezza e determinarne la complessità computazionale.