Algoritmi e Strutture Dati
a.a. 2015/16

Compito del 12/01/2017

Cognome: ________________________________ Nome: ____________________________________

Matricola: ________________________________ E-mail: ____________________________________

Parte I

(30 minuti; ogni esercizio vale 2 punti)

1. Nell'ipotesi di indirizzamento aperto, scrivere uno pseudocodice per HASH-INSERT. Qual è la complessità nel caso medio?

2. Per un certo problema sono stati trovati due algoritmi risolutivi (A_1 e A_2) con i seguenti tempi di esecuzione:

   A_1 : $T(n) = 7 \cdot T(n/3) + n^2$

   A_2 : $T(n) = 4 \cdot T(n/2) + \log n$

   Si dica, giustificando tecnicamente la risposta, quale dei due algoritmi è preferibile per input di dimensione sufficientemente grande.

3. Si completi la tabella sottostante, specificando le complessità degli algoritmi indicati in funzione della tipologia di grafo utilizzato (si indichi con n il numero dei vertici e con m il numero degli archi del grafo):

	Grafo sparso	Grafo denso
Kruskal		
Dijkstra (heap)		
Floyd-Warshall		

Algoritmi e Strutture Dati

a.a. 2015/16

Compito del 12/01/2017

Cognome: ____________________ Nome: ____________________

Matricola: ____________________ E-mail: ____________________

Parte II

(2.5 ore; ogni esercizio vale 6 punti)

1. Scrivere una **procedura efficiente** *intersezione* che, date due liste di interi (con ripetizioni), la prima *l1* ordinata in modo crescente e la seconda *l2* ordinata in modo decrescente, modifica *l1* in modo che sia l'intersezione fra *l1* e *l2*, ordinata in modo crescente, tenendo conto anche della molteplicità delle occorrenze.

   Scegliere un tipo di lista adeguato e motivare la scelta.
   Analizzare la complessità della funzione.

   Per l'esame da **12 CFU**, deve essere fornita **una procedura C** e si deve dare la **definizione in C del tipo lista**.
   Per l'esame da **9 CFU**, è sufficiente specificare lo pseudocodice.

2. Dare la definizione di albero binario di ricerca.
   Dato un albero binario con radice r, scrivere un algoritmo **efficiente** che verifichi se l'albero radicato in r è un albero binario di ricerca. L'algoritmo restituisce *1* se l'albero binario è di ricerca, *0* altrimenti.
   Analizzare la complessità dell'algoritmo.

   [Suggerimento: Può essere utile fare in modo che l'algoritmo calcoli anche il massimo e il minimo delle chiavi presenti nell'albero.]

3. Sia $G = (V, E)$ un grafo non orientato e connesso con funzione peso $w : E \rightarrow R$, e sia $(u,v) \in E$ un arco arbitrario di G. E' sempre possibile costruire un albero di copertura di G (non necessariamente minimo) che contenga (u,v)? E un albero di copertura minimo? Cosa possiamo dire nel caso in cui l'arco abbia peso minimo (ovvero, $w(u,v) \leq w(x,y)$ per ogni $(x,y) \in E)$?

   Nota: Si forniscano giustificazioni formali. In caso contrario l'esercizio non verrà valutato pienamente, anche in presenza di risposte corrette.

4. Sia $G = (V, E)$ un grafo orientato con funzione peso $w : E \rightarrow R$ e vertici numerati da 1 a n: $V = \{1, 2, ... , n\}$. Dato un vertice $k \in V$, si scriva un algoritmo che, per ogni coppia di vertici $i, j \in V$, determini la lunghezza del cammino minimo tra i e j i cui vertici intermedi non superino k. Si dimostri la correttezza dell'algoritmo proposto e si determini la sua complessità.