Esame di algoritmi e strutture dati

9 giugno 2020

Tempo a disposizione: 2 ore

Esercizio 1

(7 punti)

1. Progettare un algoritmo (pseudocodice):

   $$crescente(Lista[Interi]\ L) \rightarrow\ Booleano$$

   che, presa in input una lista L contenente valori interi, restituisca *true* se e solo se ciascun elemento e della lista è di valore minore o uguale rispetto al valore di tutti gli altri elementi che lo seguono. Esempio:

   - per $L_1 = 1, 9, 8, 12, 30$, $crescente(L_1) = true$
   - per $L_2 = 30, 9, 8, 12, 1$, $crescente(L_2) = false$

2. Determinare il costo temporale dell'algoritmo.

Esercizio 2

(7 punti)

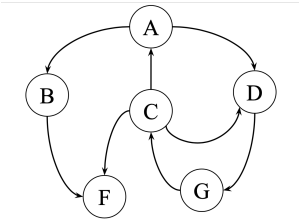
1. Mostrare lo pseudocodice dell'algoritmo di Ricerca Binaria
2. Indicare il costo dell'algoritmo nel caso peggiore
3. Applicare, quando possibile, l'algoritmo ai seguenti input (v denota l'elemento da cercare nell'array A_i):
   - $A_1 = 1, 8, 9, 12, 30$, $v = 6$
   - $A_2 = 30, 9, 8, 12, 1$, $v = 6$
4. Commentare la seguente affermazione:

   > Per cercare un elemento v in un array A, è sempre conveniente ordinare l'array A e successivamente applicare l'algoritmo di Ricerca Binaria.

Esercizio 3

(7 punti)

1. Mostrare lo pseudocodice dell'algoritmo di visita in profondità di un grafo orientato.
2. Indicare il costo dell'algoritmo.
3. Mostrare due possibili sequenze di visita in profondità dei nodi nel grafo seguente, partendo dal nodo C:



Esercizio 4

(6 punti)

1. Fornire la definizione di albero AVL.
2. Discutere le differenze tra un albero binario di ricerca ed un albero AVL.
3. Illustrare il costo dell'operazione di ricerca nei due casi.
4. Illustrare la costruzione di un albero AVL quando vengono inseriti, nell'ordine, elementi con le seguenti chiavi: 2,4,6,8,10,7,12,14.

Esercizio 5

(6 punti)
Indicare, motivando le risposte, quale delle seguenti affermazioni si ritengono vere e quali false.

1. L'algoritmo di ordinamento $mergeSort$ ha costo temporale $T(n) = O(n^2)$
2. L'algoritmo di ordinamento $mergeSort$ ha costo temporale $T(n) = O(n \log n)$
3. L'algoritmo di ordinamento $mergeSort$ ha costo temporale $T(n) = \Omega(1)$
4. L'algoritmo di ordinamento $mergeSort$ ha costo temporale $T(n) = \Theta(n \log n)$