Algoritmi e Strutture Dati

21 Gennaio 2009

Modulo Teoria

Attenzione: Scrivere nome, cognome e numero di matricola sul foglio protocollo.

Esercizio 1 [**10punti**]

Dato un array non ordinato di n interi, progettare un algoritmo efficiente che determini il numero di elementi che occorrono una sola volta e analizzarne la complessitá.

Esercizio 2 [**10punti**]

Date le stringhe $A = RISOTTO$ e $B = PRESTO$, simulare l'algoritmo di programmazione dinamica **LCS** per il calcolo della lunghezza della sottosequenza comune piú lunga, mostrando

1. il contenuto della tabella che l'algoritmi riempie dinamicamente;
2. la soluzione trovata dall'algoritmo.

Esercizio 3 [**10punti**]

Dato un array ordinato di n interi, progettare un algoritmo efficiente che costruisca un albero binario di ricerca di altezza $\Theta(\log n)$ che contiene gli interi dell'array come chiavi. Analizzare la complessità dell'algoritmo costruito utilizzando il Teorema Principale.

Modulo Laboratorio

Esercizio 1 [**14punti**]

Si dia un'implementazione in codice java dell'algoritmo di ordinamento MergeSort.

Esercizio 2 [**16punti**]

1. Si dia una caratterizzazione della struttura dati *Iteratore*.

2. Si commenti con javadoc il seguente codice, che implementa un algoritmo di visita pre-ordine di un albero.

```
void visitaPreOrdine(Node nodo) {

 if (nodo != null) {

    visita(nodo);

    Node figlio = nodo.figlio;

    while (figlio != null) {

     visitaPreOrdine(figlio);

     figlio = figlio.fratello;

    }
 }
}
```

ove si assume definita la seguente classe Node:

```
public class Node {
  Object elemento;
  Node padre,figlio,fratello;

  public Node() {
    elemento = null;
    padre = figlio = fratello = null;
  }
...
}
```