Appello del 20 febbraio 2004
Laboratorio di Algoritmi e Strutture Dati – Prof. Carlo Blundo
Anno Accademico 2003/2003– Matricole Dispari-Pari

Cognome e Nome: Numero di Matricola:

Spazio riservato alla correzione

1	2	3	4	totale
/15	/15	/15	/45	/90

Non usare altri fogli, usare solo lo spazio sottostante. Fogli differenti da questo non saranno presi in considerazione per la correzione.

1. Aggiungere alla classe **LinkedTree** (implementa l'interfaccia **Tree** usando come nodo la classe **TreeNode**) il metodo **boolean isAncestor(TreeNode n1, TreeNode n2)** che restituisca inoutput *true* qualora **n1** sia antenato di **n2**, false altrimenti. Dire quale sia la complessità del metodo proposto nel caso peggiore, motivando la risposta.Una bozza della classe **TreeNode** è la seguente:

```
public class TreeNode implements Position {
  private Object element;  private TreeNode parent;  private NodeList children;
 …..
}
```

2. Si scriva una funzione Java **Stack inverti(Stack s)** che restituisca un nuovo stack ottenuto invertendo il contenuto di **s**. Si osservi che la funzione deve lasciare inalterato il contenuto di **s** alla fine dell'esecuzione (lo può modificare durante).

3. Scrivere una funzione Java **maggiori**(Sequence **a**, Object **val**, Comparator **comp**) che restituisce in output un ObjectIterator sugli elementi di **a** che sono maggiori di **val** secondo il comparatore **comp**.

4. Minimo albero ricoprente

4.1 Scrivere lo pseudocodice dell'algoritmo di **Prim** per il calcolo del minimo albero ricoprente di un grafo (MST). Commentare l'uso delle strutture dati utilizzate. [15 punti]

4.2. Indicare, giustificando la risposta, la complessità dell'algoritmo di **Prim**. [15 punti]

4.3 Codificare l'algoritmo di **Prim** in Java (scrivere una funzione Java che implementi l'algoritmo). Il codice deve far riferimento alle interfacce illustrate durante il corso. [15 punti]