Appello del 22 febbraio 2005
Laboratorio di Algoritmi e Strutture Dati – Prof. Carlo Blundo
Anno Accademico 2004/2005– Matricole Dispari-Pari

Cognome e Nome: Numero di Matricola:

Spazio riservato alla correzione

1	2	3	4	5.1	5.2	5.3	totale
/15	/15	/20	/10	/10	/10	/10	/90

1. Aggiungere alla classe **LinkedTree** il metodo **int [] conta()** che restituisce in output un array **A** di interi contenente nella posizione **i** il numero di nodi dell'albero che hanno **i** figli (in **A[i]** deve essere memorizzato il numero di nodi dell'albero che hanno **i** figli). Per l'implementazione non è possibile utilizzare i metodi e**lements()** e **positions().**Quale è la complessità del metodo proposto (giustificare la risposta).

2. Scrivere la funzione **boolean contiene(Vector V)** che restituisce true se il vettore V contiene due elementi la cui somma è pari a 7. Si supponga che **V** contenga interi positivi. La funzione **contiene** deve avere una complessità di tempo pari ad O(n) dove n è il numero degli elementi contenuti in **V**. Illustrare la complessità di tempo della funzione proposta. Soluzioni con complessità di tempo superiore a O(n) saranno considerate nulle.

3. Implementare l'interfaccia **Map** utilizzando come variabile d'istanza un variabile di tipo **Vector**.

```
public interface Map {
        // Restituisce il numero degli elementi nella mappa
        public int size();

        // Restituisce true se la mappa è vuota
          public boolean isEmpty();

        // Inserisce una coppia chiave-valore nella mappa,  rimpiazzando il precedente se esiste
        public Object put(Object key, Object value)   throws InvalidKeyException;

        //Restituisce il valore associato alla chiave key
          public Object get(Object key) throws InvalidKeyException;

        // Rimuove la coppia chiave-valore specificata da key
          public Object remove(Object key) throws InvalidKeyException;

        // Restituisce un iteratore su tutte le chiavi della mappa
          public Iterator keys();

        // Restituisce un iteratore su tutte i valori della mappa
          public Iterator values();
}
```

4. Implementare il metodo **public boolean areAdjacent(Vertex v1, Vertex v2)** della classe **ILUndirectedGraph** che implementa l'interfaccia **UndirectedGraph** mediante le liste di incidenza.

5. **Pianificazione scolastica**

Ogni anno il Provveditorato agli Studi deve effettuare un censimento per calcolare il numero di bambini che dovranno iscriversi alla scuola elementare e distribuirli tra le varie scuole della città. La distribuzione deve essere effettuata in modo tale da assicurare che ogni bambino possa frequentare una scuola che non disti più di **d** chilometri dalla sua abitazione. D'altra parte, ogni scuola non può accogliere più di un certo numero di bambini, in funzione del numero di maestri e delle strutture che ha a disposizione.

Il programma che acquisisce i dati del censimento annuale e delle scuole funzionanti e provvede alla distribuzione dei bambini tra le varie scuole. La distribuzione calcolata deve rispettare i vincoli sulla distanza che un bambino deve percorrere per raggiungere la sua scuola e sul numero di bambini che ogni scuola può accogliere. Il programma, rispettando i vincoli di capienza di una scuola, deve massimizzare il numero di bambini che possono frequentare scuole che non distino più **d** chilometri dalle proprie abitazioni.

1. Supponendo che un bambino ed una scuola sono rappresentate dalle classi **Bambino** e **Scuola**, rispettivamente, descrivere la struttura dati utilizzata per rappresentare il problema in esame (in quale struttura dati memorizzate i bambini, le scuole, come rappresentate il fatto che ogni bambino può andare in una scuola lontana al più **d** chilometri dalla sua abitazione, mentre ogni scuola non può accettare più bambini del numero di maestri e delle strutture che ha a disposizione …).

2. Scrivere lo pseudo-codice dell'algoritmo utilizzato per massimizzare il numero di persone occupate

3. Analizzare la complessità dell'algoritmo proposto

Nota: l'esercizio è risolvibile usando opportunamente uno degli algoritmi illustrati durante le lezioni di teoria.