Laboratorio di Algoritmi e Strutture Dati

Esercitazioni del 22 Gennaio 2013

Esercizio 3: grafi orientati

Possiamo specificare un grafo non orientato tramite un $file$ il cui formato è il seguente:

- $file$ contiene una sequenza di coppie di interi positivi.
- La prima coppia (nv, ne) specifica il numero di vertici e il numero di archi del grafo. In particolare, gli nv vertici del grafo sono etichettati con gli interi positivi $1, 2, \ldots, nv$.
- Le coppie rimanenti specificano gli archi del grafo in questo modo: la coppia (i, j) specifica che il grafo contiene un arco (non orientato) tra il vertice con etichetta i e il vertice con etichetta j.

Esaminate i file di esempio `grafo1.txt` e `grafo2.txt`.

Il programma `graph.c`, allegato al materiale preparato per questa lezione, viene richiamato come

`graph` $file$

dove $file$ contiene la specifica di un grafo non orientato.

Il programma `graph` opera nel modo seguente:

1. Crea il grafo non orientato specificato da $file$.
2. Stampa l'elenco dei vertici ottenuto tramite una visita in profondità del grafo stesso. In particolar modo, stampa l'elenco dei vertici di componenti connesse distinte su righe distinte.

In questo esercizio si chiede di:

1. Interpretare un $file$ nel formato precedente come la specifica di un grafo *orientato*: la coppia (i, j) ora deve specificare un arco *orientato*, *dal* vertice con etichetta i *al* vertice con etichetta j.
2. Modificare l'implementazione in `graph.c` in modo che leggendo un $file$ nel formato sopra specificato, vada a costruire un grafo orientato.
3. Modificare le funzioni per la visita in profondità del grafo in modo da implementare una funzione:

   ```
   char reachable(struct graph *g, int i, int j)
   ```

   che restituisca 1 se nel grafo `g` vi è un cammino dal vertice etichettato con `i` al vertice etichettato con `j`; 0 se non esiste un siffatto cammino.

Il programma così modificato deve:

1. Creare il grafo orientato specificato da $file$.
2. Stampare ogni coppia di vertici (i, j) connessi da un cammino (dal vertice i al vertice j).