Corso di Informatica 1 (IN110) – Prof. Marco Liverani – a.a. 2014/2015

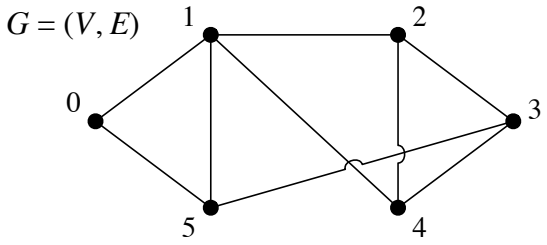
Esame scritto del 11 Settembre 2015 (Appello X)

Si richiede di risolvere entrambi gli esercizi riportando una codifica in linguaggio C completa dei due programmi. Nel caso in cui non si riesca a completare entrambi gli esercizi si suggerisce di riportare almeno la codifica in C delle funzioni principali o una loro pseudo-codifica. È possibile consultare libri e appunti personali, ma non scambiare libri o appunti con altri studenti. I compiti che presenteranno evidenti ed anomale "similitudini" saranno annullati. La prova scritta ha una durata di tre ore, durante le quali non è consentito allontanarsi dall'aula, se non dopo aver consegnato il compito. Si richiede di riportare sul foglio del compito il proprio nominativo completo ed il numero di matricola o un codice identificativo personale equivalente.

Esercizio n. 1

Leggere in input le liste di adiacenza di un grafo non orientato $G = (V, E)$, con n vertici. Stampare le liste di adiacenza di G. Letto in input un intero $k > 0$ e un vertice $v \in V(G)$ stampare tutti i vertici $u \in V(G)$ che hanno in comune con v almeno k vertici adiacenti.

Esempio Si consideri il grafo $G = (V, E)$ rappresentato in figura. Sia $v = 5$ e $k = 2$; i vertici che hanno almeno due vertici in comune con il vertice $v = 5$ sono $u = 2$ e $u = 4$.



Soluzione

```c
#include <stdlib.h>
#include <stdio.h>
#define MAX 100

struct nodo {
  int info;
  struct nodo *next;
};

struct nodo *leggiLista(void) {
  struct nodo *p, *primo=NULL;
  int n, i;
  printf("Numero di elementi: ");
  scanf("%d", &n);
```

```
15    printf("inserisci %d interi: ", n);
16    for (i=0; i<n; i++) {
17      p = malloc(sizeof(struct nodo));
18      scanf("%d", &p->info);
19      p->next = primo;
20      primo = p;
21    }
22    return(primo);
23  }
24
25  int leggiGrafo(struct nodo *G[]) {
26    int i, n;
27    printf("Numero di vertici: ");
28    scanf("%d", &n);
29    for (i=0; i<n; i++) {
30      printf("Lista di adiacenza di %d.\n", i);
31      G[i] = leggiLista();
32    }
33    return(n);
34  }
35
36  void stampaLista(struct nodo *p) {
37    while (p != NULL) {
38      printf("%d --> ", p->info);
39      p = p->next;
40    }
41    printf("NULL\n");
42    return;
43  }
44
45  void stampaGrafo(struct nodo *G[], int n) {
46    int i;
47    for (i=0; i<n; i++) {
48      printf("%d: ", i);
49      stampaLista(G[i]);
50    }
51    return;
52  }
53
54  int adiacente(int x, struct nodo *p) {
55    int r;
56    while (p != NULL && p->info != x)
57      p = p->next;
58    if (p != NULL)
59      r = 1;
60    else
61      r = 0;
62    return(r);
63  }
```

```
64
65 int main(void) {
66   struct nodo *G[MAX], *p;
67   int n, k, u, v, cont;
68   n = leggiGrafo(G);
69   stampaGrafo(G, n);
70   printf("Inserisci un intero k>0 e un vertice v del grafo: ");
71   scanf("%d %d", &k, &v);
72   printf("Vertici con almeno %d adiacenti in comune con %d: ", k, v);
73   for (u=0; u<n; u++) {
74     if (u != v) {
75       cont = 0;
76       p = G[u];
77       while (p!=NULL && cont<k) {
78         if (adiacente(p->info, G[v]))
79           cont++;
80         p = p->next;
81       }
82       if (cont==k)
83         printf("%d ", u);
84     }
85   }
86   printf("\n");
87   return(0);
88 }
```

Esercizio n. 2

Letti in input due interi $n, k > 0$, generare due matrici quadrate di ordine n, A e B, di numeri interi casuali compresi tra 0 e k (estremi inclusi); la matrice A dovrà essere costituita di soli numeri pari e la matrice B di soli numeri dispari. Stampare le matrici A e B. Costruire un vettore C di n numeri interi in cui l'elemento generico c_h $(0 \leq h < n)$ è dato dal prodotto tra il minimo elemento della colonna h di A e il massimo elemento della riga h di B.

Esempio Sia $n = 4$ e $k = 20$; consideriamo le seguenti matrici:

$$A = \begin{pmatrix} 6 & 8 & 8 & 12 \\ 10 & 0 & 16 & 16 \\ 2 & 10 & 16 & 18 \\ 14 & 14 & 10 & 4 \end{pmatrix} \qquad B = \begin{pmatrix} 17 & 1 & 11 & 13 \\ 11 & 3 & 5 & 19 \\ 3 & 17 & 1 & 15 \\ 11 & 9 & 13 & 3 \end{pmatrix}$$

Il vettore C di $n = 4$ elementi è quindi il seguente: $C = (34, 0, 136, 52)$.

Soluzione

```c
#include <stdlib.h>
#include <stdio.h>
#include <time.h>
#define MAX 100

void generaMatricePari(int X[MAX][MAX], int n, int k) {
  int i, j;
  for (i=0; i<n; i++)
    for (j=0; j<n; j++)
      X[i][j] = (rand()%(k/2 + 1)) * 2;
  return;
}

void generaMatriceDispari(int X[MAX][MAX], int n, int k) {
  int i, j;
  for (i=0; i<n; i++)
    for (j=0; j<n; j++)
      X[i][j] = (rand()%(k/2 + k%2)) * 2 + 1;
  return;
}

void stampaMatrice(int X[MAX][MAX], int n) {
  int i, j;
  for (i=0; i<n; i++) {
    for (j=0; j<n; j++)
      printf("%2d ", X[i][j]);
    printf("\n");
  }
  printf("\n");
  return;
}
```

```
32
33 int min(int X[MAX][MAX], int n, int h) {
34   int i, m;
35   m = X[0][h];
36   for (i=1; i<n; i++)
37     if (X[i][h] < m)
38       m = X[i][h];
39   return(m);
40 }
41
42 int max(int X[MAX][MAX], int n, int h) {
43   int i, m;
44   m = X[h][0];
45   for (i=1; i<n; i++)
46     if (X[h][i] > m)
47       m = X[h][i];
48   return(m);
49 }
50
51 void costruisciVettore(int A[MAX][MAX],int B[MAX][MAX],int C[MAX],int n) {
52   int h;
53   for (h=0; h<n; h++)
54     C[h] = min(A, n, h) * max(B, n, h);
55   return;
56 }
57
58 void stampaVettore(int X[], int n) {
59   int i;
60   for (i=0; i<n; i++)
61     printf("%2d ", X[i]);
62   printf("\n");
63   return;
64 }
65
66 int main(void) {
67   int A[MAX][MAX], B[MAX][MAX], C[MAX], n, k;
68   printf("Ordine delle matrici quadrate: ");
69   scanf("%d", &n);
70   printf("Soglia dei numeri casuali: ");
71   scanf("%d", &k);
72   srand((unsigned)time(NULL));
73   generaMatricePari(A, n, k);
74   generaMatriceDispari(B, n, k);
75   stampaMatrice(A, n);
76   stampaMatrice(B, n);
77   costruisciVettore(A, B, C, n);
78   stampaVettore(C, n);
79   return(0);
80 }
```