Corso di Informatica 1 (IN110 - Fondamenti) – Prof. Marco Liverani – a.a. 2009/2010

Esame scritto del 9 luglio 2010

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.

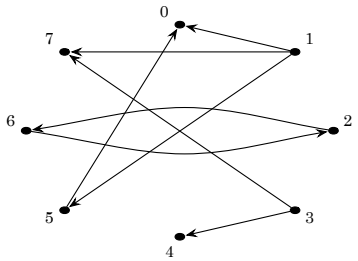
Esercizio n. 1

Letti in input due interi $n,k > 0$ costruire un vettore $X = \{x_0, x_1, \ldots, x_{n-1}\}$ di numeri interi positivi casuali minori di k. Costruire le liste di adiacenza del grafo orientato $G = (V,E)$ tale che $V(G) = \{0,1,\ldots,n-1\}$ ed $E(G) = \{(u,v) : u,v \in V(G), x_v = hx_u$ per qualche $h \in \mathbb{N}, h > 0\}$. Stampare il vettore X e le liste di adiacenza del grafo G.

Esempio Si consideri il caso in cui $n = 8$ e $k = 10$; il vettore di numeri casuali minori di k è il seguente:

$$X = (x_0 = 8, x_1 = 2, x_2 = 7, x_3 = 3, x_4 = 9, x_5 = 4, x_6 = 7, x_7 = 6)$$

Il grafo $G = (V,E)$ è dato dai seguenti insiemi ed è rappresentato in figura: $V(G) = \{0,1,2,3,4,5,6,7\}$, $E(G) = \{(1,0),(1,5),(1,7),(2,6),(3,4),(3,7),(5,0),(6,2)\}$.



Soluzione

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

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

int costruisciVettore(int X[]) {
  int n, i, k;
```

```
13    printf("Numero di elementi: ");
14    scanf("%d", &n);
15    printf("Valore soglia: ");
16    scanf("%d", &k);
17    srand((unsigned)time(NULL));
18    for (i=0; i<n; i++)
19      X[i] = (rand() % (k-1)) + 1;
20    return(n);
21  }
22
23  void stampaVettore(int X[], int n) {
24    int i;
25    for (i=0; i<n; i++)
26      printf("%d ", X[i]);
27    printf("\n");
28    return;
29  }
30
31  void costruisciGrafo(struct nodo *G[], int X[], int n) {
32    int i, j;
33    struct nodo *p;
34    for (i=0; i<n; i++)
35      G[i] = NULL;
36    for (i=0; i<n; i++) {
37      for (j=0; j<n; j++) {
38        if (i != j && X[j] % X[i] == 0) {
39          p = malloc(sizeof(struct nodo));
40          p->info = j;
41          p->next = G[i];
42          G[i] = p;
43        }
44      }
45    }
46    return;
47  }
48
49  void stampaLista(struct nodo *p) {
50    while (p != NULL) {
51      printf("%d --> ", p->info);
52      p = p->next;
53    }
54    printf("NULL\n");
55    return;
56  }
57
58  void stampaGrafo(struct nodo *G[], int n) {
59    int i;
60    for (i=0; i<n; i++) {
61      printf("%d: ", i);
62      stampaLista(G[i]);
63    }
64    return;
65  }
66
```

```c
int main(void) {
  int n, X[MAX];
  struct nodo *G[MAX];
  n = costruisciVettore(X);
  stampaVettore(X, n);
  costruisciGrafo(G, X, n);
  stampaGrafo(G, n);
  return(0);
}
```

Esercizio n. 2

Letti in input tre numeri interi positivi n, h, k, con $h < k$, costruire una sequenza di n numeri casuali compresi tra h e k (estremi inclusi) e memorizzarla nella prima colonna di una matrice quadrata A di ordine n. Costruire le rimanenti $n-1$ colonne della matrice A sulla base della seguente formula:

$$a_{i,j} = \sum_{p=n-1}^{i} a_{p,j-1} \text{ per } i = 0,1,2,\ldots,n-1 \text{ e } j = 1,2,\ldots,n-1$$

Stampare la matrice A.

Esempio Sia $n = 5$, $h = 4$, $k = 10$. La matrice A è la seguente:

$$A = \begin{pmatrix} 7 & 33 & 100 & 232 & 458 \\ 4 & 26 & 67 & 132 & 226 \\ 8 & 22 & 41 & 65 & 94 \\ 9 & 14 & 19 & 24 & 29 \\ 5 & 5 & 5 & 5 & 5 \end{pmatrix}$$

Soluzione

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

void costruisciMatrice(int A[MAX][MAX], int n, int h, int k) {
  int i, j, p, c;
  c = 1;
  srand((unsigned)time(NULL));
  for (i=0; i<n; i++)
    A[i][0] = (rand() % (k-h+1)) + h;
  for (j=1; j<n; j++) {
    for (i=0; i<n; i++) {
      A[i][j] = 0;
      for (p=n-1; p>=i; p--)
        A[i][j] += A[p][j-1];
    }
  }
  return;
}

```

```c
void stampaMatrice(int A[MAX][MAX], int n) {
  int i, j;
  for (i=0; i<n; i++) {
    for (j=0; j<n && A[i][j]>0; j++)
      printf("%3d ", A[i][j]);
    printf("\n");
  }
  return;
}

int main(void) {
  int n, h, k, A[MAX][MAX];
  printf("Ordine della matrice: ");
  scanf("%d", &n);
  printf("Valori soglia: ");
  scanf("%d %d", &h, &k);
  costruisciMatrice(A, n, h, k);
  stampaMatrice(A, n);
  return(0);
}
```

Soluzione alternativa

```c
#include ...
#define MAX 100

void costruisciMatrice(int A[MAX][MAX], int n, int h, int k) {
  int i, j, c;
  c = 1;
  srand((unsigned)time(NULL));
  for (i=0; i<n; i++)
    A[i][0] = (rand() % (k-h+1)) + h;
  for (j=1; j<n; j++) {
    A[n-1][j] = A[n-1][j-1];
    for (i=n-2; i>=0; i--) {
      A[i][j] = A[i+1][j] + A[i][j-1];
    }
  }
  return;
}

void stampaMatrice(int A[MAX][MAX], int n) {
  ...
}

int main(void) {
  int n, h, k, A[MAX][MAX];
  ...
  costruisciMatrice(A, n, h, k);
  stampaMatrice(A, n);
  return(0);
}
```