

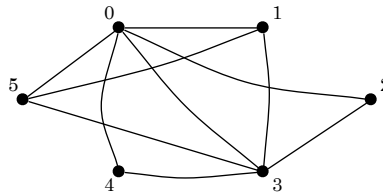
## Esame scritto del 18 febbraio 2011

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 un grafo non orientato  $G = (V, E)$  e memorizzarlo per mezzo di liste di adiacenza. Stampare le liste di adiacenza del grafo. Letto in input un sottoinsieme  $V'$  di vertici di  $V$ , memorizzarlo mediante una lista. Verificare se il sottografo indotto da  $V'$  su  $G$  è un grafo completo.

**Esempio** Si consideri il grafo  $G$  rappresentato in figura. Il sottografo indotto su  $G$  dal sottoinsieme di vertici  $V' = \{0, 1, 3, 5\}$  è completo (tutti i vertici di  $V'$  sono fra loro adiacenti nel grafo  $G$ ); il sottografo indotto da  $V' = \{1, 2, 3\}$  non è completo (i vertici 1 e 2 non sono adiacenti in  $G$ ).



### Soluzione

```

1 #include <stdlib.h>
2 #include <stdio.h>
3 #define MAX 100
4
5 struct nodo {
6     int info;
7     struct nodo *next;
8 };
9
10 struct nodo *leggiLista(void) {
11     struct nodo *p, *primo = NULL;
12     int i, n;
13     printf("Numero di elementi: ");
14     scanf("%d", &n);
15     printf("Inserisci %d vertici: ", 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("G[%d]: ", i);
49         stampaLista(G[i]);
50     }
51     return;
52 }
53
54 int adiacente(struct nodo *G[], int u, int v) {
55     struct nodo *p;
56     int r;
57     p = G[u];
58     while (p != NULL && p->info != v)
59         p = p->next;
60     if (p == NULL)
61         r = 0;
62     else
63         r = 1;
64     return(r);
65 }
66
67 int completo(struct nodo *G[], int n, struct nodo *p) {
68     int flag=1;
69     struct nodo *q, *r;
70     q = p;
71     while (flag == 1 && q != NULL) {
72         r = p;

```

```

73     while (flag == 1 && r != NULL) {
74         if (q->info != r->info && !adiacente(G, q->info, r->info))
75             flag = 0;
76         r = r->next;
77     }
78     q = q->next;
79 }
80 return(flag);
81 }
82
83 int main(void) {
84     struct nodo *p, *G[MAX];
85     int n;
86     n = leggiGrafo(G);
87     p = leggiLista();
88     if (completo(G, n, p))
89         printf("Il sottografo indotto da V' e' completo.\n");
90     else
91         printf("Il sottografo indotto da V' non e' completo.\n");
92     return(0);
93 }

```

## Esercizio n. 2

Letti in input due interi  $n, k > 0$ , con  $k < n$ , costruire un array  $A$  di  $n$  numeri interi casuali compresi tra 0 e 100, estremi inclusi. Stampare il vettore  $A$  e tutte le sequenze di  $k$  elementi consecutivi di  $A$ ,  $(a_i, a_{i+1}, a_{i+2}, \dots, a_{i+k-1})$ , tali che la somma degli elementi della sequenza sia dispari.

**Esempio** Sia  $n = 7, k = 3$  e sia  $A = (3, 4, 27, 84, 36, 36, 11)$ . Le sequenze costituite da  $k = 3$  elementi consecutivi di  $A$  la cui somma è dispari sono le seguenti:

(4, 27, 84)   (27, 84, 36)   (36, 36, 11)

## Soluzione

```

1  #include <stdlib.h>
2  #include <stdio.h>
3  #include <time.h>
4  #define MAX 100
5
6  int generaVettore(int A[]) {
7      int i, n;
8      printf("Numero di elementi: ");
9      scanf("%d", &n);
10     srand((unsigned)time(NULL));
11     for (i=0; i<n; i++)
12         A[i] = rand() % 101;
13     return(n);
14 }
15
16 void stampaVettore(int A[], int n) {
17     int i;

```

```

18     for (i=0; i<n; i++)
19         printf("%d ", A[i]);
20     printf("\n");
21     return;
22 }
23
24 void stampaSequenze(int A[], int n, int k) {
25     int i, j, s;
26     for (i=0; i<=n-k; i++) {
27         s = 0;
28         for (j=0; j<k; j++)
29             s = s + A[i+j];
30         if (s % 2 == 1)
31             stampaVettore(&A[i], k);
32     }
33     return;
34 }
35
36 int main(void) {
37     int n, k, A[MAX];
38     n = generaVettore(A);
39     printf("Inserisci k: ");
40     scanf("%d", &k);
41     stampaVettore(A, n);
42     stampaSequenze(A, n, k);
43     return(0);
44 }

```