

## Esame scritto del 17 gennaio 2008

### Esercizio n. 1

Leggere in input tre stringhe di caratteri  $A$ ,  $B$  e  $C$ . Sostituire nella stringa  $A$  ogni occorrenza della stringa  $B$  con  $C$ . Stampare la stringa  $A$  modificata.

**Esempio** Si considerino le stringhe  $A = \text{"xYZtwYZkf"}$ ,  $B = \text{"YZ"}$  e  $C = \text{"PQRS"}$ . La stringa  $A$  contiene due occorrenze di  $B$ ; dopo aver sostituito  $B$  in  $A$  con la stringa  $C$  si ottiene la seguente stringa  $A$  modificata:  $A = \text{"xPQRStwPQRskf"}$ .

### Soluzione

```

1  #include <stdlib.h>
2  #include <stdio.h>
3  #include <string.h>
4  #define MAX 100
5
6  int findReplace(char a[], char b[], char c[]) {
7      int i, j, h, cont=0;
8      for (i=0; i<=strlen(a)-strlen(b); i++) {
9          for (j=0; j<strlen(b) && a[i+j] == b[j]; j++)
10             ;
11         if (j == strlen(b)) {
12             cont++;
13             /* tolgo b da a */
14             for (h=i; h<=strlen(a); h++) {
15                 a[h] = a[h+strlen(b)];
16             }
17             /* faccio spazio in a per c */
18             for (h=strlen(a); h>=i; h--) {
19                 a[h+strlen(c)] = a[h];
20             }
21             /* inserisco c in a */
22             for (h=0; h<strlen(c); h++) {
23                 a[i+h] = c[h];
24             }
25             i = i+strlen(c)-1;
26         }
27     }
28     return(cont);
29 }
30
31 int main(void) {
32     char A[MAX], B[MAX], C[MAX];
33     int n;
34     printf("Inserisci tre stringhe di caratteri: ");
35     scanf("%s %s %s", A, B, C);
36     n = findReplace(A, B, C);
37     if (n) {

```

```

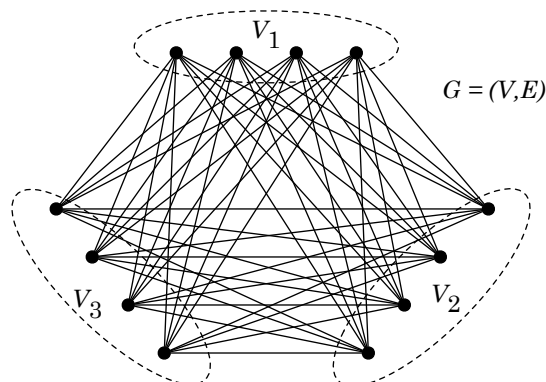
38     printf("Sono state sostituite %d occorrenze di '%s' con '%s'.\n", n, B, C);
39     printf("La nuova stringa e' '%s'\n", A);
40 } else {
41     printf("La stringa '%s' non contiene '%s'.\n", A, B);
42 }
43 return(0);
44 }

```

## Esercizio n. 2

Letti in input due interi  $h, k > 0$ , costruire le liste di adiacenza del grafo  $h$ -partito completo  $K_{\underbrace{k, k, \dots, k}_h}$ , costituito da  $h$  insiemi indipendenti di  $k$  vertici ciascuno. Stampare le liste di adiacenza del grafo.

**Esempio** Siano  $h = 3$  e  $k = 4$ ; il grafo  $K_{4,4,4}$  che si vuole ottenere è rappresentato in figura ed è costituito dai vertici  $V = \{0, 1, 2, \dots, 11\}$ , suddivisi in  $h = 3$  sottoinsiemi indipendenti di  $k = 4$  vertici ciascuno:  $V_1 = \{0, 1, 2, 3\}, V_2 = \{4, 5, 6, 7\}, V_3 = \{8, 9, 10, 11\}$ . L'insieme degli spigoli è  $E = \{(u, v) : u \in V_i, v \in V_j, i \neq j\}$ .



## 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 void stampaLista(struct nodo *p) {
11     while (p != NULL) {
12         printf("%d --> ", p->info);
13         p = p->next;
14     }
15     printf("NULL\n");

```

```

16     return;
17 }
18
19 void stampaGrafo(struct nodo *G[], int n) {
20     int i;
21     for (i=0; i<n; i++) {
22         printf("%d: ", i);
23         stampaLista(G[i]);
24     }
25     return;
26 }
27
28 void aggiungiSpigolo(struct nodo *G[], int u, int v) {
29     struct nodo *p;
30     p = malloc(sizeof(struct nodo));
31     p->info = v;
32     p->next = G[u];
33     G[u] = p;
34     p = malloc(sizeof(struct nodo));
35     p->info = u;
36     p->next = G[v];
37     G[v] = p;
38     return;
39 }
40
41 void grafoCompletoMultipartito(struct nodo *G[], int h, int k) {
42     int i, j, s;
43     for (i=0; i<h*k; i++)
44         G[i] = NULL;
45     for (i=0; i<h; i++)
46         for (j=0; j<k; j++)
47             for (s=i*k+k; s<k*h; s++)
48                 aggiungiSpigolo(G, i*k+j, s);
49     return;
50 }
51
52 int main(void) {
53     int h, k;
54     struct nodo *G[MAX];
55     printf("Inserisci due interi positivi: ");
56     scanf("%d %d", &h, &k);
57     grafoCompletoMultipartito(G, h, k);
58     stampaGrafo(G, h*k);
59     return(0);
60 }

```