Soluzione Scritto del 13/7/18

Esercizio 1

$$ED[i,j] = ED[i-1, j-1] \quad \text{se } S_1[i] = S_2[j]$$

$$\min \{ ED[i-1, j] + 1,\ ED[i, j-1] + 1 \quad \text{se } S_1[i] \neq S_2[j],\ ED[i-1, j-1] + 2 \}$$

		Ε	M	I	C	I	O
		0	1	2	3	4	5
A	1	1	2	3	4	5	6
M	2	2	1	2	3	4	5
I	3	3	2	1	2	3	4
C	4	4	3	2	1	2	3
O	5	5	4	3	2	3	2

−	M	I	C	I	O
A	M	I	C	−	O

Esercizio 2

La modifica richiesta dall'esercizio impone di riscrivere la BFS in maniera simile alla BFS e quindi facendo alla costruzione di una foresta di alberi BFS secondo il grafo G non fosse connesso.

```
BFS(G)                                   (*)
  for u in V(G) do
    if (color(u) == BIANCO)
      BFS(G, u)
```

Occorre poi modificare la BFS(G,u) vista in classe in modo da far partire in cui i nodi vengono colorati

di BIANCO e "inizializzati" nei campi π e d

```
For each u ∈ V(G) do
    color[u] = BIANCO
    d[u] = +∞
    π[u] = NIL
```

} ★

dove erano anticipate nella porzione indicata con (★) all'interno di BFS(G), affinché l'inizializzazione sia effettuata una sola volta e il risultato delle sorie chiamate a BFS(G,u) sia visibile alla procedure master BFS(G).

Esercizio 3

Il modo più semplice per risolvere l'esercizio era quello di ripotarsi nella situazione nota, cioè con la ricerca eseguita a partire dalla radice dell'albero T.

```
Tree-search*(x, π)
    y = x;
    while parent(y) ≠ NIL do        // nella ricerca si potrebbe
        y = parent(y);                 anche controllare se
                                       y.key == k.
    return Tree-search-clone(y, k);  // vista in classe
```

La complessità al caso peggiore di Tree-search* è identica a quella della Tree-search vista in classe in quanto stiamo eseguendo una ricerca nell'albero T che quindi ha costo pari all'altezza dell'albero al più.