

```

(int, int) CamminiMinimi(GRAPH G, NODE s)

// Inizializzazione dei vettori
int[] d ← new int[1...G.n]                                // distanze dalla sorgente
int[] T ← new int[1...G.n]                                  // vettore dei padri
bool[] b ← new bool[1...G.n]                               // per sapere in tempo costante se  $u \in S$ 

// Inizializzo tutti i nodi tranne la sorgente
foreach  $u \in G.V - \{s\}$  do
    T[u] ← nil // non hanno padri
    d[u] ← +∞ // non li ho ancora raggiunti
    b[u] ← false // non appartengono ancora all'insieme

// Inizializzo la sorgente
T[s] ← nil // non ha padre d[s] ← 0 // per convenzione b[s] ← true // appartiene all'insieme
STRUTTURADATI S ← StrutturaDati S.aggiungi(s)

while not S.isEmpty do
    int u ← S.estrai // estraggo un nodo
    b[u] ← false // non è più contenuto nella struttura dati

    foreach  $v \in G.adj(u)$  do // per tutti i vicini
        if  $d[u] + G.w(u,v) < d[v]$  then // se migliora la stima
            if not b[v] then // se non fa già parte dell'insieme
                S.aggiungi(v) // aggiungilo b[v] ← true // fa parte dell'insieme
            else
                // Azione da intraprendere nel caso  $v$  sia già presente in  $S$ 
                // aggiorno i vettori
                T[v] ← u d[v] ← d[u] + G.w(u,v)

return (T,d)

```