

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

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

// Inizializzo tutti i nodi tranne la sorgente

foreach $u \in G.V - \{s\}$ do

$T[u] \leftarrow \text{nil}$ // non hanno padri
 $d[u] \leftarrow +\infty$ // non li ho ancora raggiunti
 $b[u] \leftarrow \text{falso}$ // non appartengono ancora all'insieme

// Inizializzo la sorgente

$T[s] \leftarrow \text{nil}$ // non ha padre
 $d[s] \leftarrow 0$ // per convenzione
 $b[s] \leftarrow \text{vero}$ // appartiene all'insieme

(1) STRUTTURADATI $S \leftarrow \text{StrutturaDati}$

$S.\text{aggiungi}(s)$

(2) while not $S.\text{isEmpty}$ do

 int $u \leftarrow S.\text{estrai}$ // estraggo un nodo
 $b[u] \leftarrow \text{falso}$ // non è più contenuto nella struttura dati
 foreach $v \in G.\text{adj}(u)$ 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
 $S.\text{aggiungi}(v)$ // aggiungilo
 $b[v] \leftarrow \text{vero}$ // fa parte dell'insieme

 else

 // Azione da intraprendere nel caso v sia già presente in S

 // aggiorno i vettori

$T[v] \leftarrow u$

$d[v] \leftarrow d[u] + G.w(u,v)$

return (T, d)