L'algoritmo di Prim produce ovviamente un albero, perché ad ogni inserimento volta aggiunge un'altro dei nodi già connettendo l'albero in S e non nodi "fuori" di S (quindi non crea cicli).

Ad ogni passo aggiunge a T l'arco di minimo costo che ha un estremo u in S ad un nodo $v \in V-S$.

Esercizio 34

```
Kruskal(G=(V,E), c)
  for ogni vertice v ∈ V
     MAKE-SET(v)
  ORDINA c(e) dove e=(u,v) per non decrescente
  T ← ∅
  foreach e(u,v) ∈ E
      if FINDSET(u) ≠ FIND-SET(v)
          T ← T U (u,v)
          UNION(u,v)
  return T
```

L'algoritmo di Kruskal è efficiente perché confrontando le componenti connesse determina se l'aggiunta di un arco (u,v) creerà un ciclo. Se $c(u)=c(v)$ creerà un ciclo, se diverse non creerà un ciclo.

Aggiungiamo a T ad uno ad uno gli archi del grafo, in ordine crescente, saltando gli archi che creano cicli con gli archi già aggiunti.