Diseñar un algoritmo de tiempo $O(n+m)$ que, dado un grafo conexo G con pesos en sus aristas y un vértice v, determine el árbol de menor peso de entre todos los árboles v-geodésicos de G. **Justificar** que el algoritmo propuesto es correcto. **Ayuda:** pensar cuáles aristas pueden pertenecer a un árbol v-geodésico cualquiera, para elegir las que minimicen el peso total.

$G = (V, E)$

Suponemos que si es ponderado existe una función C tal que $C(v, w) \to n, n \in \mathbb{N}, v, w \in V(G)$

Algoritmo

```
Algoritmo(G,v,C):
  dist <- BFS(G,v) tal que devuelve un vector de distancias de v a todos

  minp <- array todo en inf de tamaño n
  padre <- array todo en -1 de tamaño n

  Para cada nodo u:
    Para cada nodo w en ady(u):
      Si dist[u]=dist[w]+1:
        Si C(u,w) < minp[w]:
          padre[w] = u
          minp[w] = C(u,w)

  A <- vector vacio usado como lista de aristas

  Para cada u: 1...n:
    Si u!=v y padre[u] != Nil:
      A = A ∪ (padre[u],u)

  Return A
```

Correctitud

BFS puede devolver un vector `dist` de las distancias de la raiz a cada nodo, fué probado en la teórica.

Luego nos fijamos para cada arista `u,w` $\in V(G)$ si `dist[u]=dist[w]+1` tal que nos asegure que sea v-geodésico, y si lo es, al final del loop tendremos que `minp[w]` es el peso mínimo y el indicente de cada nodo w.

`dist[u]=dist[w]+1` nos asegura que es v-geodésico

Dado que cada nodo tiene **exactamente un padre**, es árbol.

Como solo guardamos las aristas de peso mínimo con `minp[w], padre[w]`, es mínimo.

Por último solo construimos el árbol uniendo cada `padre[u]`, `u` con $u \in V$ en una lista de aristas o cualquier representación válida de un grafo, y este será nuestro resultado.