Algoritmo de Kruskal

Es un algoritmo de la teoría de grafos para encontrar un árbol recubridor mínimo en un grafo conexo y ponderado. Es decir, busca un subconjunto de aristas que, formando un árbol, incluyen todos los vértices y donde el valor de la suma de todas las aristas del árbol es el mínimo. Si el grafo no es conexo, entonces busca un bosque expandido mínimo (un *árbol expandido mínimo* para cada componente conexa). Este algoritmo toma su nombre de Joseph Kruskal, quien lo publicó por primera vez en 1956. Otros algoritmos que sirven para hallar el **árbol de expansión mínima** o **árbol recubridor mínimo** son el algoritmo de Prim, el algoritmo del borrador inverso y el algoritmo de Boruvka.

Pseudocodigo

```
función Kruskal(G)
  Para cada v en V[G] hacer
    Nuevo conjunto C(v) ← {v}.
  Nuevo heap Q que contiene todas las aristas de G, ordenando por su peso
  Defino un árbol T ← Ø
  // n es el número total de vértices
  Mientras T tenga menos de n-1 aristas y !Q.vacío() hacer
    (u,v) ← Q.sacarMin()
    // previene ciclos en T. agrega (u,v) si u y v están
    // diferentes componentes en el conjunto.
    // Nótese que C(u) devuelve la componente a la que pertenece u
    Si C(v) ≠ C(u) hacer
      Agregar arista (v,u) a  T
      Merge C(v) y C(u) en el conjunto
  Responder árbol T
```