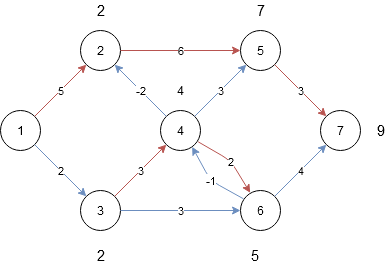
2.



Las distancias son los números fuera de los nodos. Los arcos azules son los recorridos de menor distancia, los arcos rojos no se consideran.

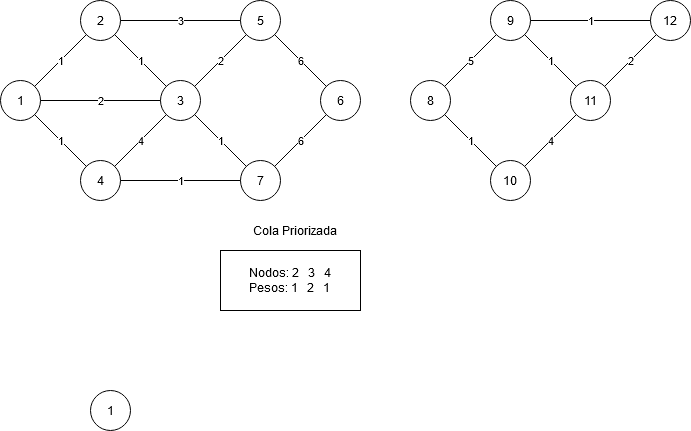
1. Si el peso del arco 6 → 4 cambiase a -3, la distancia al nodo 4 sería 2, la distancia a 2 sería 0, la distancia al nodo 5 seria 5, y la distancia al nodo 7 sería de 8 en lugar de 9, ya que en vez de acceder a ese nodo por el arco 6 → 7, se accedería a través de los nodos 4 y 5.
2. Si todos los pesos fueran positivos, el algoritmo se podría seguir utilizando. Sin embargo, sería más eficiente usar el algoritmo de Dijkstra, que es más sencillo, ya que el algoritmo de Bellman-Ford es simplemente una versión modificada del algoritmo de Dijkstra usada cuando hay arcos con pesos negativos.

3.

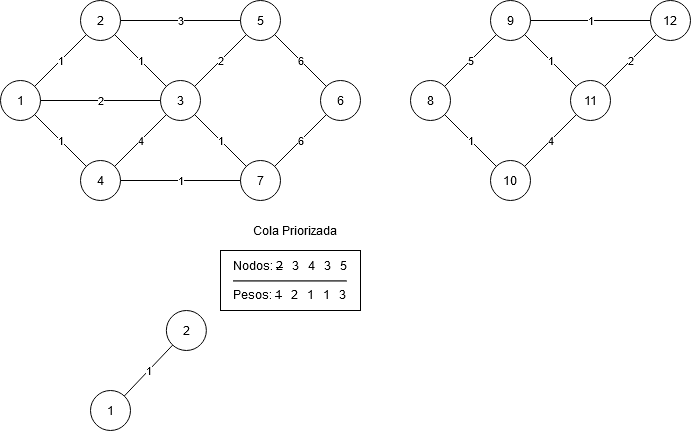
|  |  |
| --- | --- |
| Algoritmo | Complejidad |
| Dijkstra | O(log(V)\*E) |
| Bellman-Ford | O(V\*E) |
| Floyd-Warshall | O(V^3) |

4. Prim:

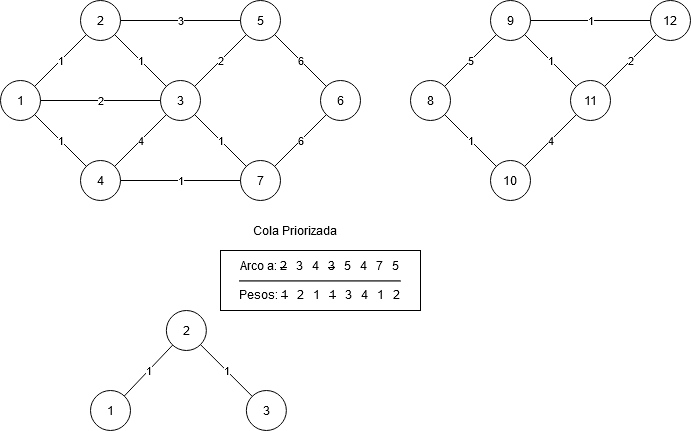
Empezamos con el nodo 1, lo agregamos al grafo y agregamos sus arcos a la cola:



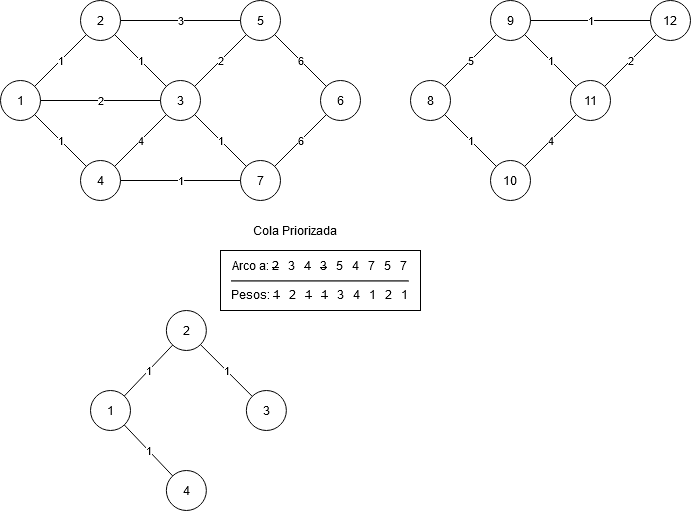
Continuamos con el nodo 2 por tener el arco de menor peso:



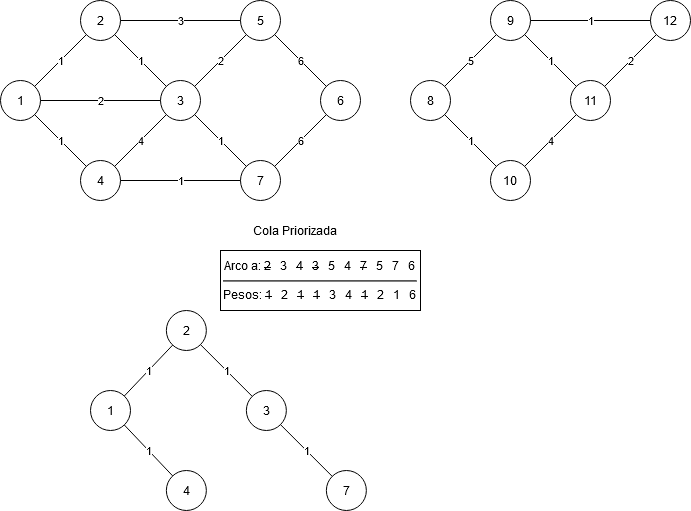
Se procesa el nodo 3:



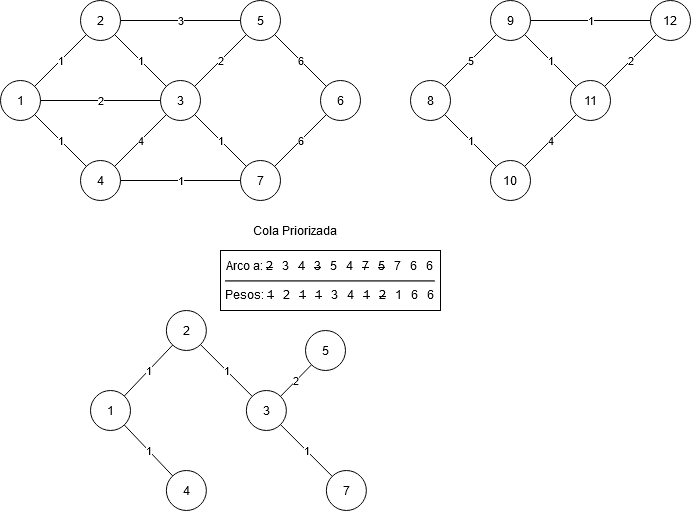
Se procesa el 4:



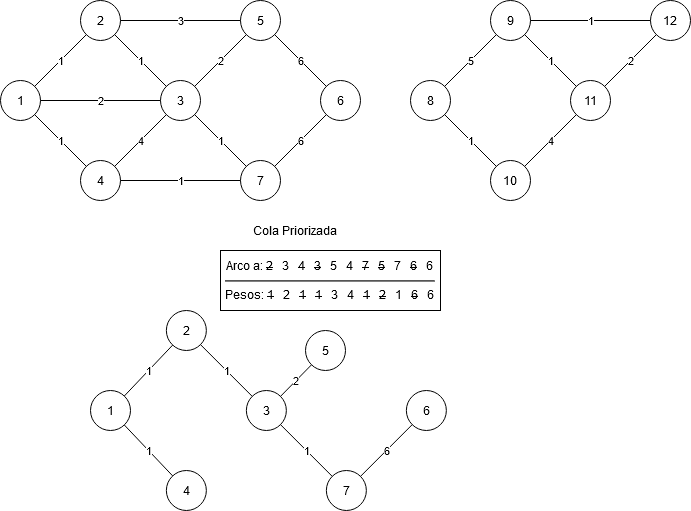
Se procesa el 7:



Se procesa el 5:

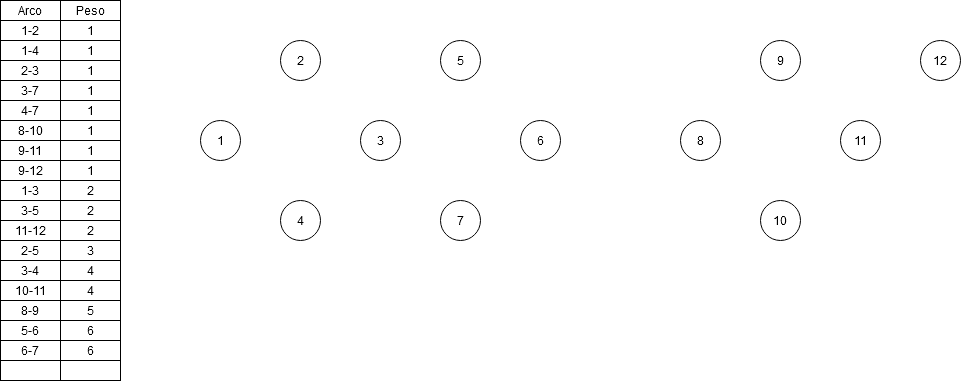


Finalmente se procesa el 6:

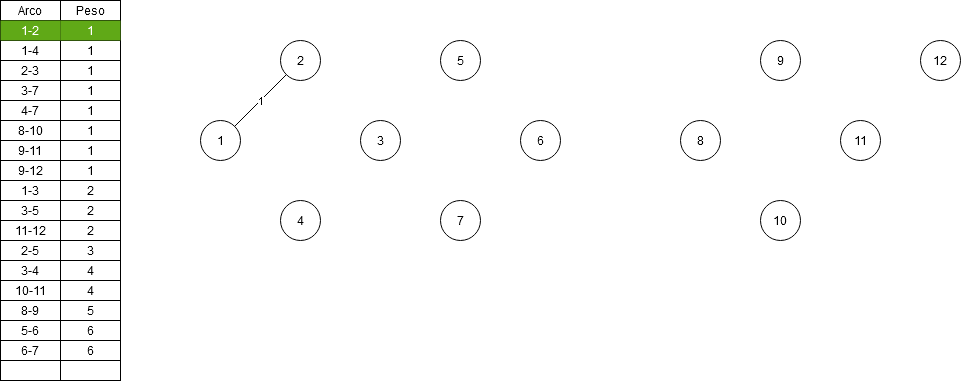


Kruskal:

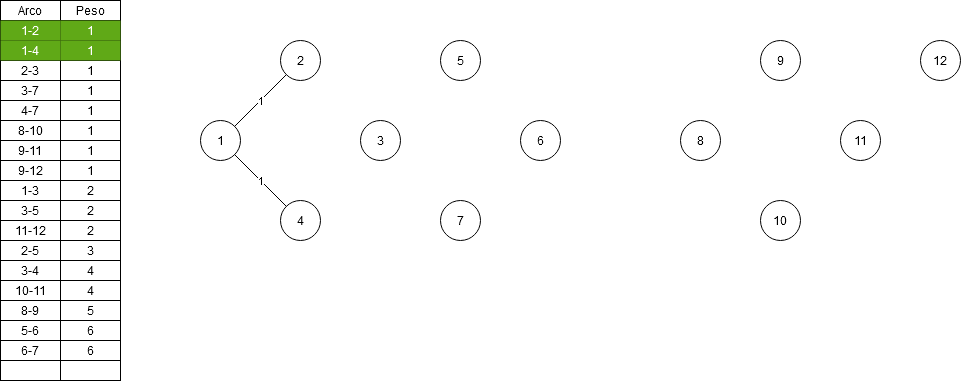
Se crea un bosque con cada nodo como un árbol independiente:



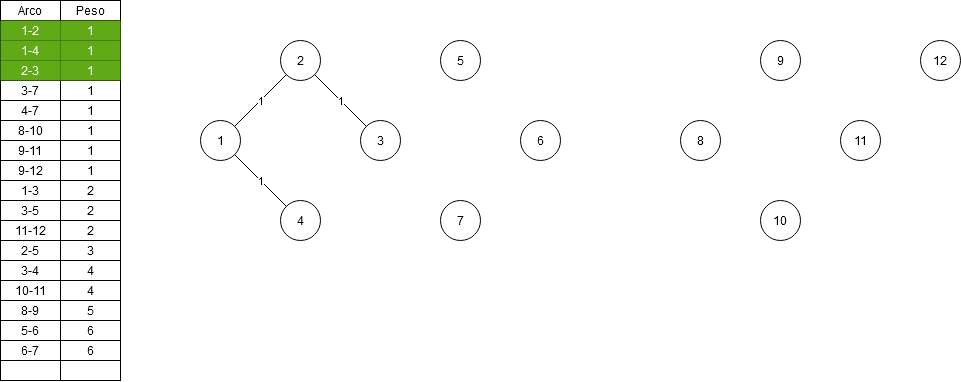
Colocamos el primer arco:



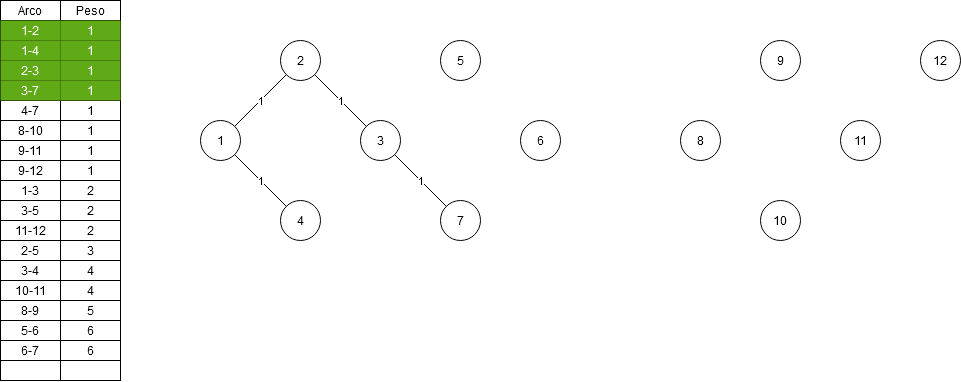
Colocamos el segundo arco:



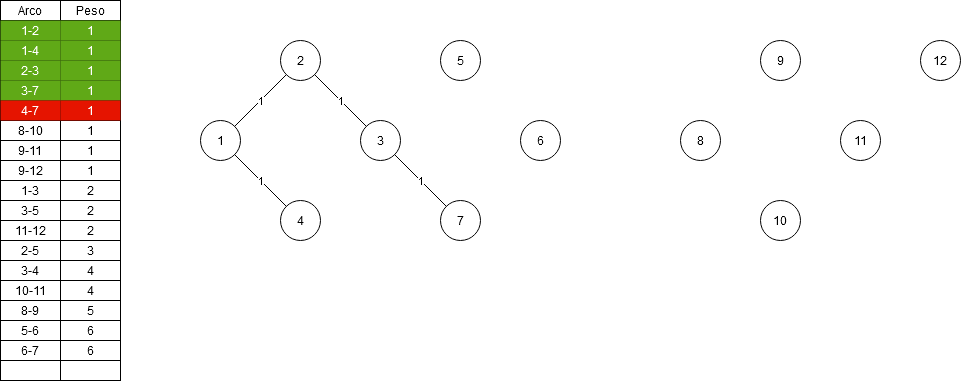
Tercer arco:



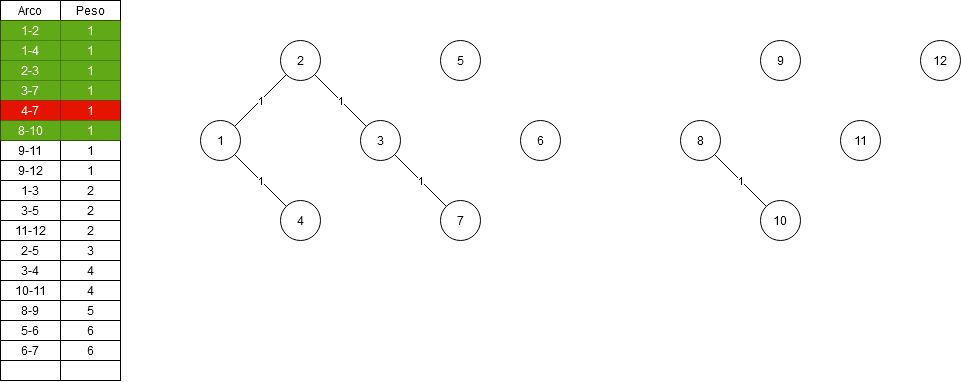
Cuarto arco:

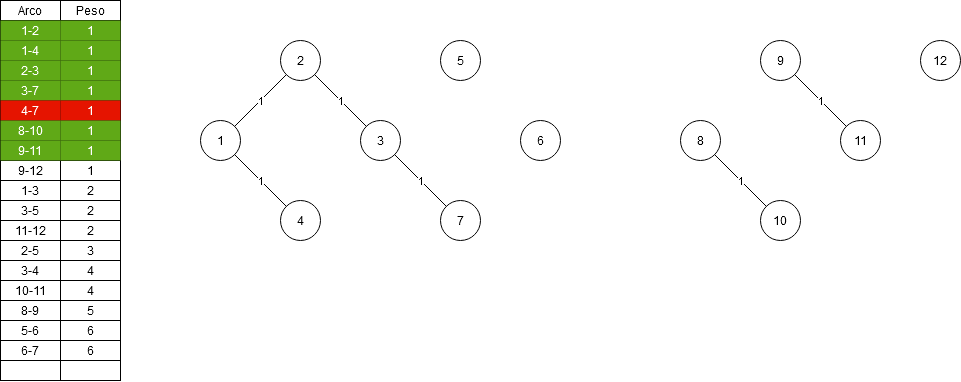


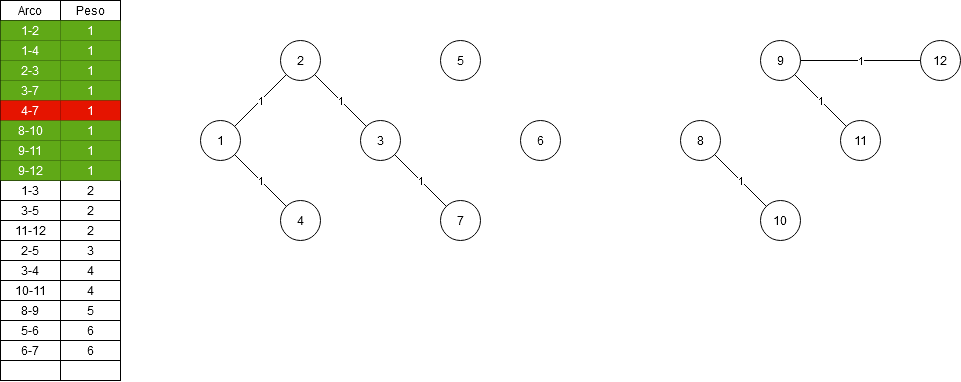
El quinto arco, 4-7, une dos nodos que ya están en un mismo árbol, por lo que no se inserta y lo descartamos:



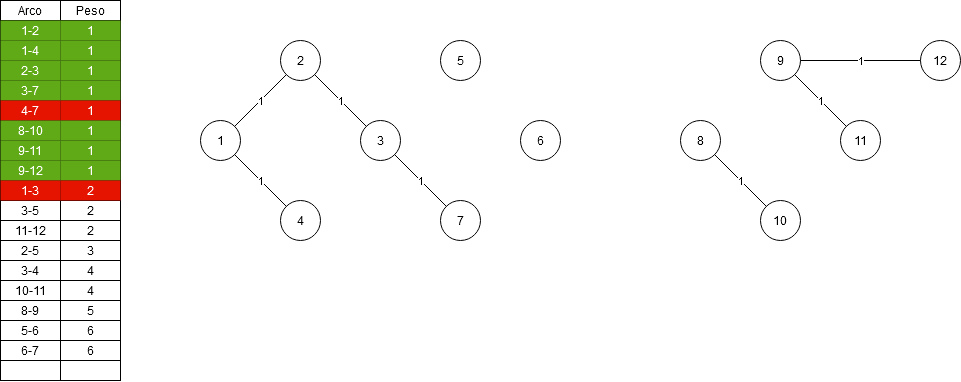
Continuamos agregando arcos:



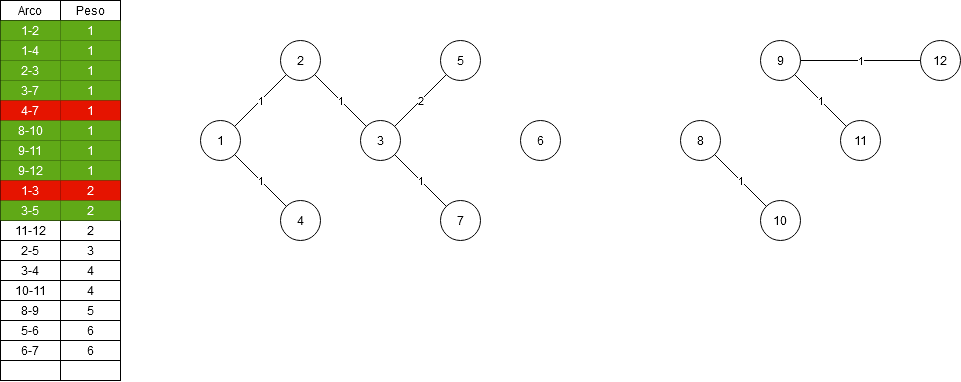




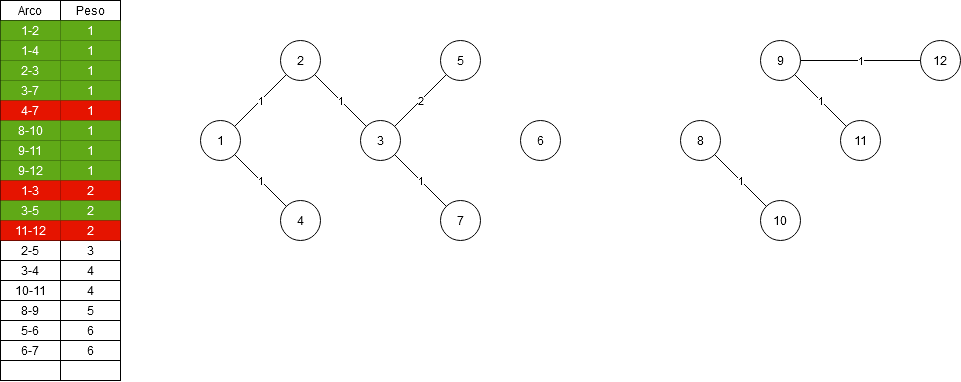
El arco 1-3 une dos nodos de un mismo grafo, lo descartamos:



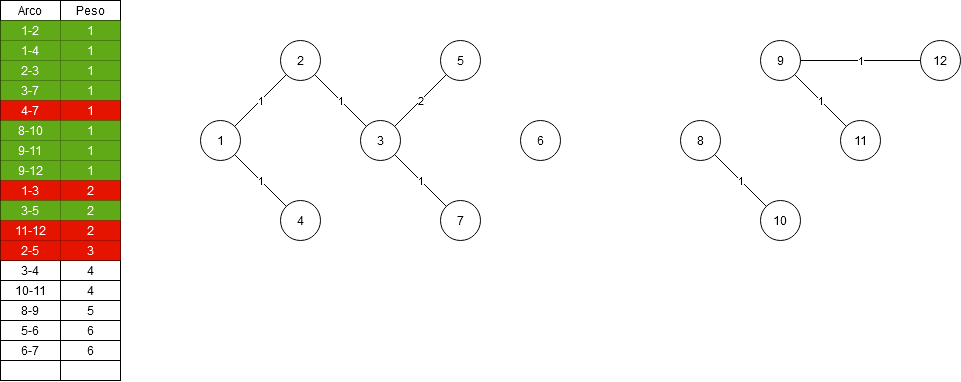
Agregamos el siguiente:

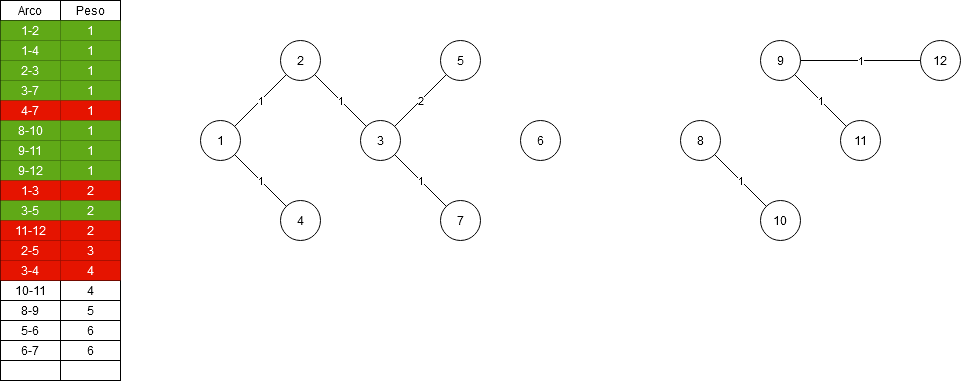


El arco 11-12 une dos nodos de un mismo árbol, lo descartamos:

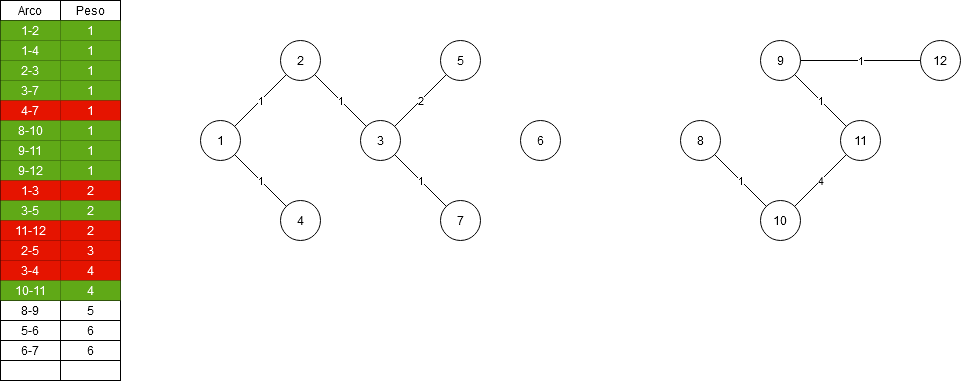


Lo mismo con el 2-5 y 3-4:

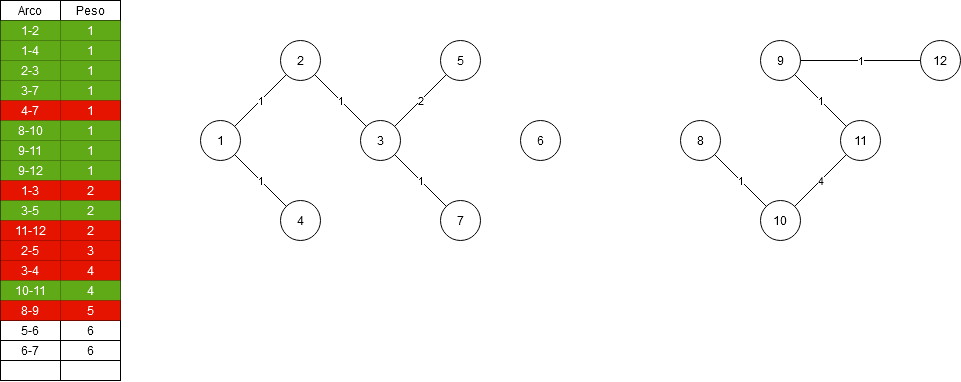




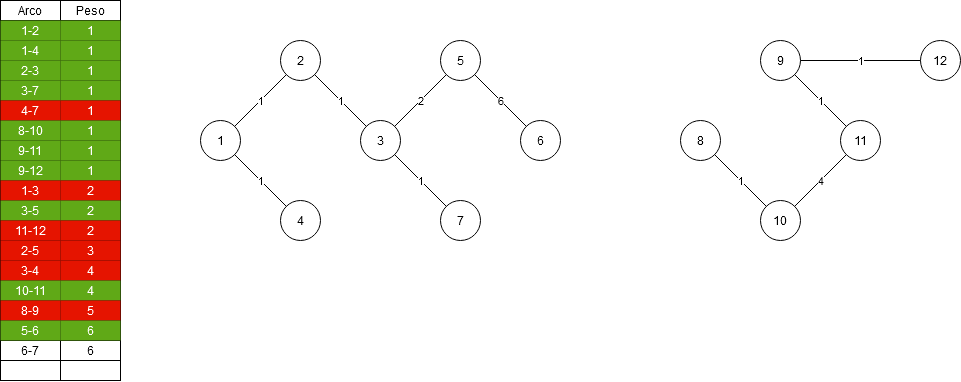
Agregamos el 10-11:



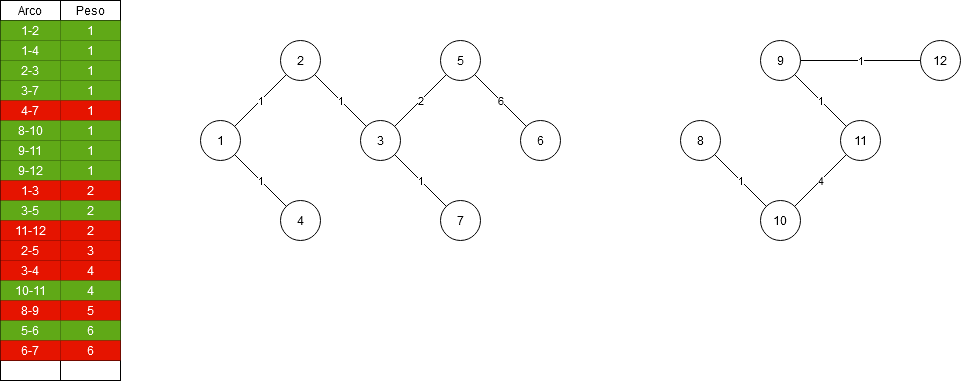
Descartamos 8-9:



Agregamos 5-6:



Descartamos 6-7, y nos queda el grafo terminado:



b. El algoritmo de Prim comienza con un solo nodo y los va agregando progresivamente, mientras que el algoritmo de Kruskal inicia con todos los nodos y solo agrega los arcos.

c. Prim: O((E+V)log(V)).

Kruskal: O(E\*log(E).

d. El algoritmo de Prim permitiría reconocer el corte ya que al momento de obtener el A.E.M solo se verá una porción del grafo, ya que, por su funcionamiento solo agrega al A.E.M los nodos que tengan conexión al nodo elegido como inicial.