**Cuestiones**

I-C. Cuestiones sobre CDs y CCs:

* ***1. Sin darnos cuenta, en nuestro algoritmo de encontrar CCs podemos pasar listas con ramas repetidas o donde los vértices coinciden con los de una que ya está aunque en orden inverso. ¿Afectará esto al resultado del algoritmo? ¿Por qué?***

*No, no afectará al resultado, esto se debe a que en nuestro algoritmo se busca el representante de cada conjunto, de no coincidir los representantes se realizará una unión, si un nodo ya está en el conjunto detectará que el representante es igual, si por ejemplo pasamos de argumento [(1,0),(0,1)] veremos que el resultado será {1:[1,0]}.*



Podemos probar con algo más elaborado:

g = [(0, 12), (10, 0), (7, 12), (1, 9), (3, 8), (3, 9), (11, 6), (2, 5), (4, 5), (0, 2)]

g = [(0, 12), (10, 0), (7, 12), (1, 9), (3, 8), (3, 9), (11, 6), (2, 5), (4, 5), (0, 2), (12, 7), (1,9)]

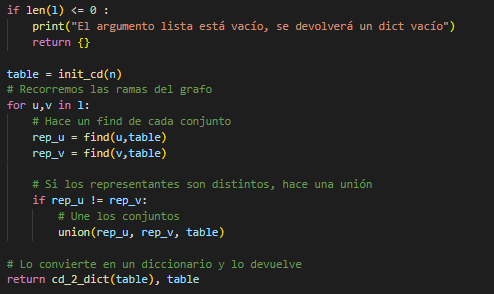
Hemos repetido (1,9) e invertido (12,7), resultados:

Resultado en ambos casos:

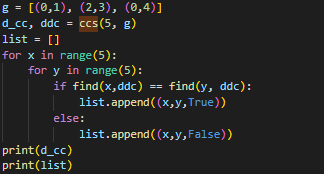


* **2. Argumentar que nuestro algoritmo de encontrar componentes conexas es correcto, esto es, que a su final en los distintos subconjuntos disjuntos se encuentran los vértices de las distintas componentes del grafo dado.**

Podemos comprobar esto modificando varias cosas, primero modificamos css para que nos devuelva el array sin transformar a diccionario:



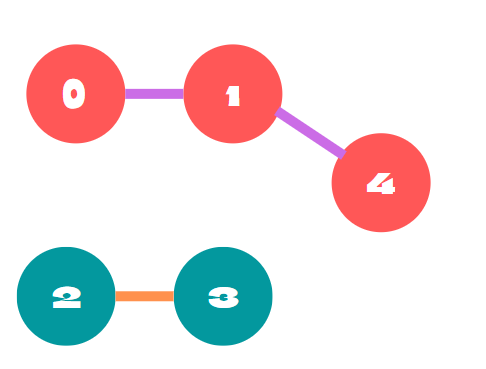
Y creamos código para comprobar los representantes de cada nodo:



Esto nos mostrará las conexiones existentes:



Como se puede observar 0 está conectado consigo mismo, con 1 y con 4, 1 está conectado con 0, consigo mismo, y con 4, 2 está conectado consgo mismo y con 3 y por último 3 está conectado con 2 y consigo mismo, una representación visual sería:



* **3. El tamaño de un grafo no dirigido viene determinado por el número n de nodos y la longitud de la lista l de ramas. Estimar razonadamente en función de ambos el coste del algoritmo de encontrar las componentes conexas mediante conjuntos disjuntos.**

En nuestro código realizamos L uniones, entonces el coste sería de: O(L + M lg∗ N) teniendo en cuenta que M = Ω(N) finds con compresión de caminos, en nuestro código son dos finds con compresión, por lo tanto el coste será de O(L + 2N lg∗ N)