2076. Process Restricted Friend Requests

- input: n, restrictions, requests
  - → 0 to n-1 personas
  - → vector restrictions[i] = [x_i, y_i]
    - ↳ persona x_i y persona y_i no pueden ser amigos
  - → vector requests[j] = [u_j, v_j]
    - ↳ request is successful si u_j y v_j pueden ser amigos directamente

ej

restrictions = [[0,1], [1,2], [2,3]]

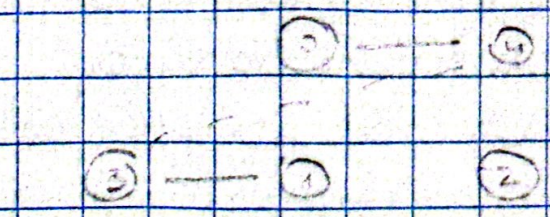
requests = [[0,4], [1,2], [3,1], [3,4]]

[0,4] si

[1,2] no

[3,1] si

[3,4] no xq 0 y 1 no pueden ser amigos (indirectamente)



0 1 2 3 4

0,4 → 0 1 2 3
4

1,2 → estan directly restricted
así que no

0 1 2 3
4

1,3 → no estan

0 1 2
4 3

3,4 → no pueden, los padres (1 y 0) estan restricted, así

0 → 1	0 → 1
1 → 0, 2	1 → 0, 2
2 → 1, 3	2 → 1, 3
3 → 2	3 → 2
4 →	4 → 1
0 → 1	0 → 1
1 → 0, 2	1 → 0, 2
2 → 1, 3 →	2 → 1
3 → 2, 0	3 → 0, 2
4 → 1	4 → 1

1. DSU para las conexiones y una lista para las restricciones de cada user
2. 1.1 Se procesa cada restriction, se agrega la restricción para cada user
(ej. [3,1], a 1 se le agregaría 3 como restricción y a 3 se le agrega 1)
1.2 El DSU se inicializa con cada uno como sus padres
2. Se procesa cada requests Se verifica si los users en el requests están restricted
2.1 Si los users no están restricted, ~~se hace unionset~~ se revisa que no estén unidos (no tengan el mismo padre), si no se hace unionset las restricciones de los users que se acaban de unir se deben combinar.
(ej. unimos 2 y 4, 2 tiene 1 y 3 de restricción 4 tiene 5 y 3. Ahora 2 tendrá 1, 3 y 5 y 4 también tendrá 1, 3 y 5
2.2 Si los users sí están restricted, continue
↳ pushback false al vector de respuesta
pushback true al vector de respuesta

→ Se debe verificar que los friends de ambas partes del request no tenga restricciones con cada uno