AYUDANTÍA 11

DFS, topSort y Kosaraju

Ignacio Bascuñán

PROBLEMA 1

- topSort: Orden topológico.

$$\boldsymbol{topSort}(\boldsymbol{G})$$

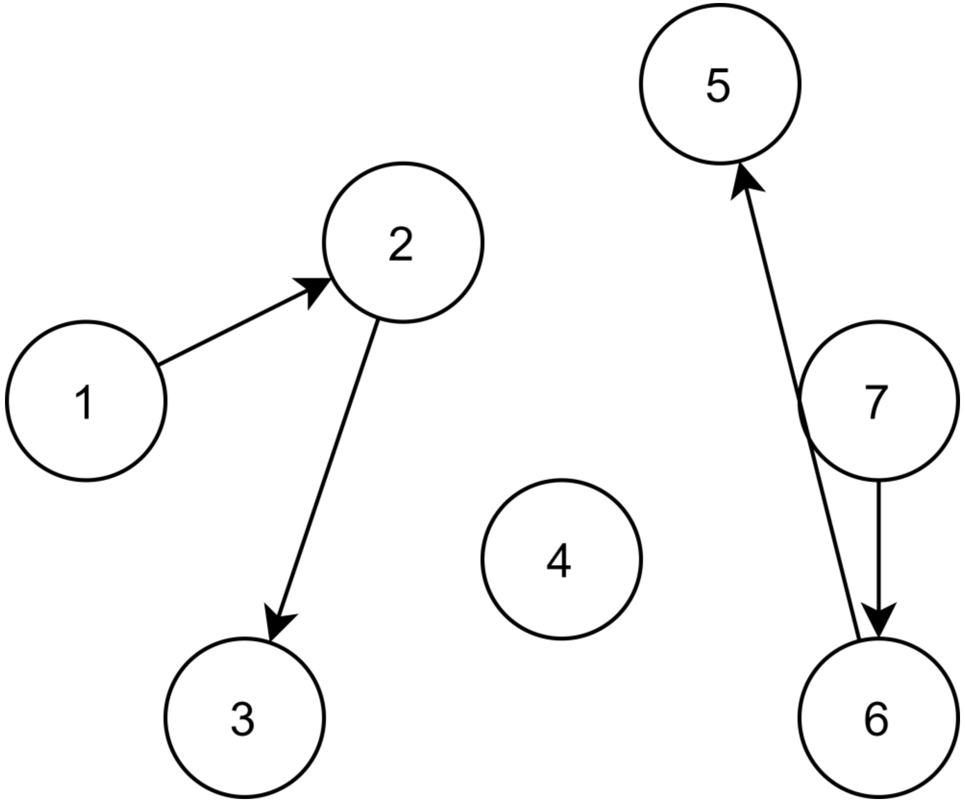
Crear lista $\boldsymbol{L}$ vacía

Ejecutar $\boldsymbol{dfs}(\boldsymbol{G})$ con tiempos:

- cada vez que calculamos el tiempo $\boldsymbol{end}$ para un nodo, insertamos ese nodo al frente de $\boldsymbol{L}$

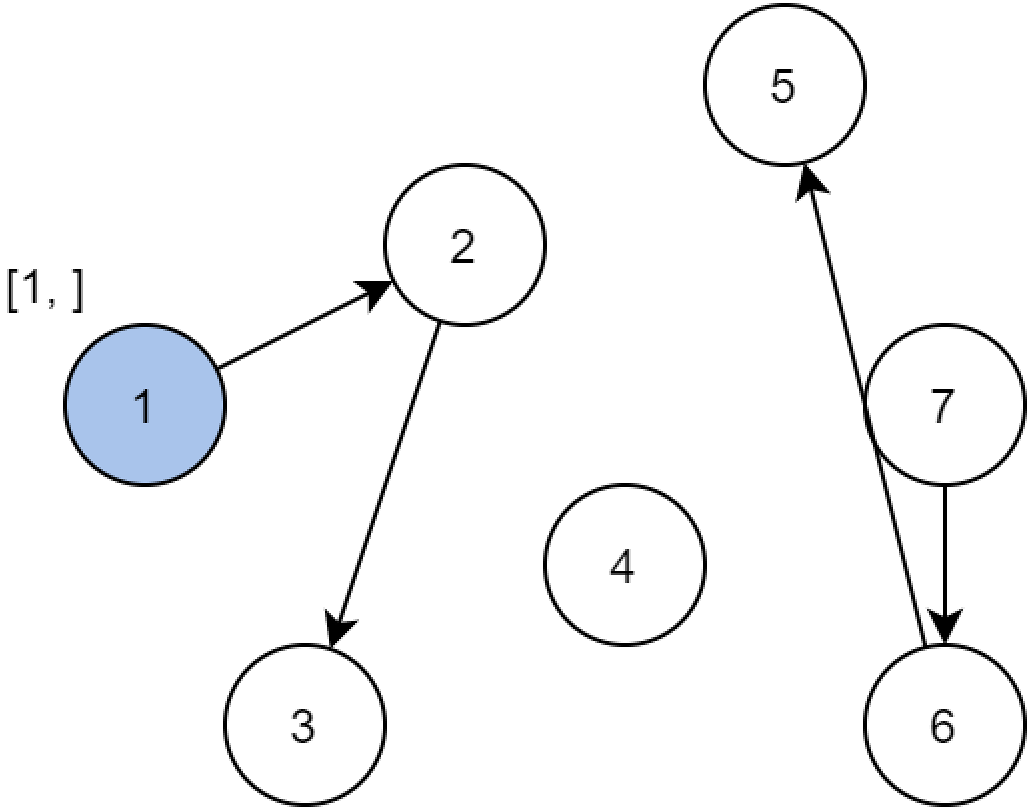
$\boldsymbol{return}\ \boldsymbol{L}$

PROBLEMA 1



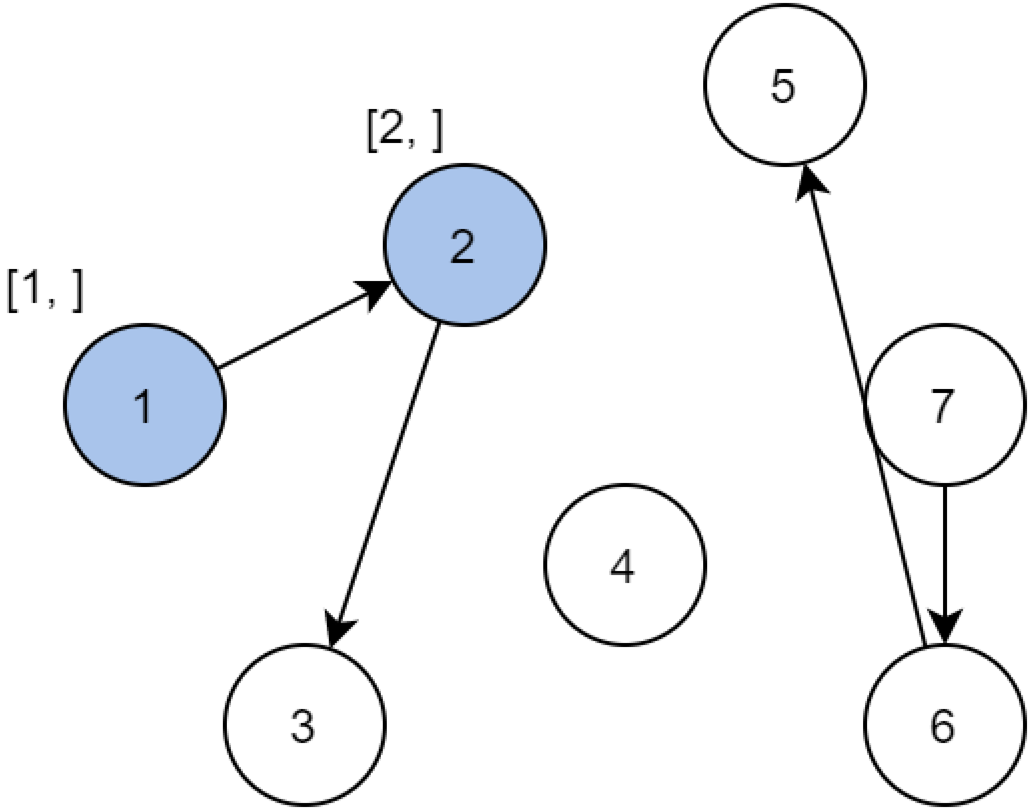
L = [ ]

PROBLEMA 1



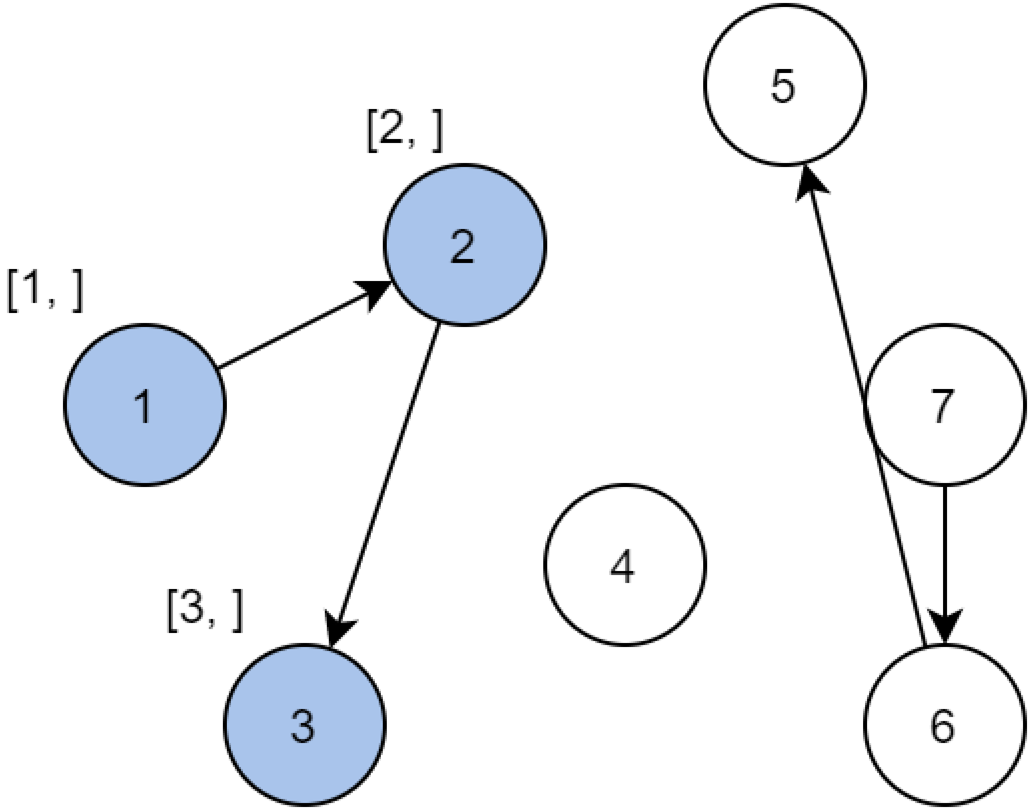
L = [ ]

PROBLEMA 1



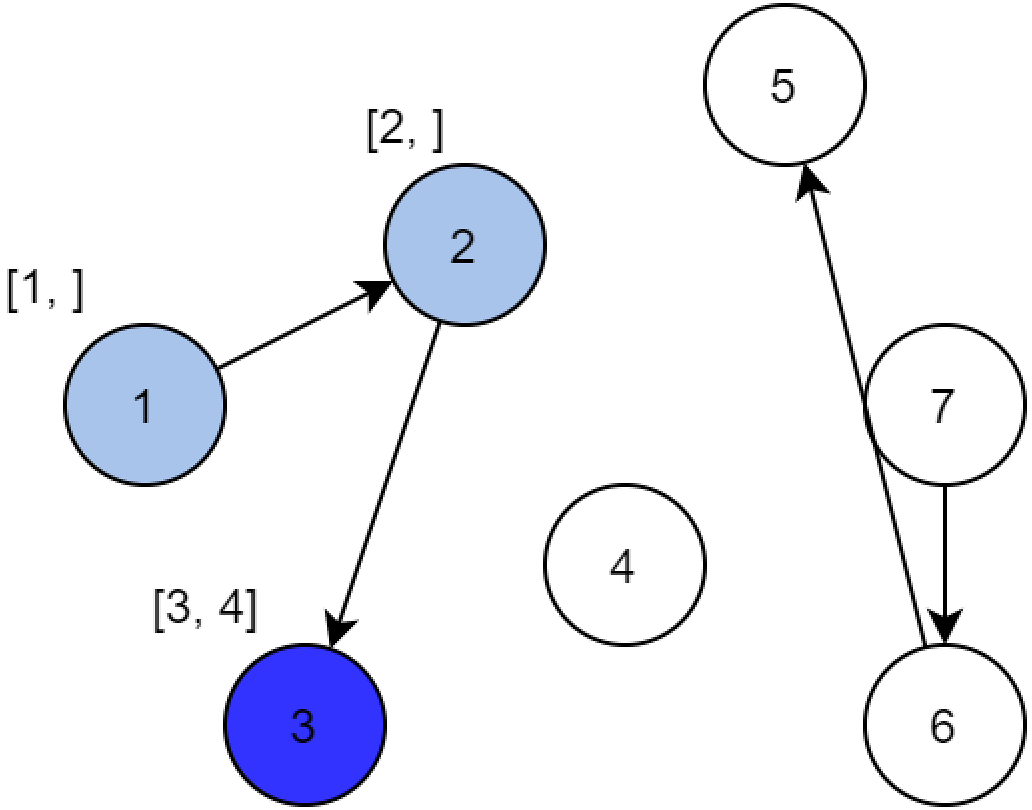
L = [ ]

PROBLEMA 1



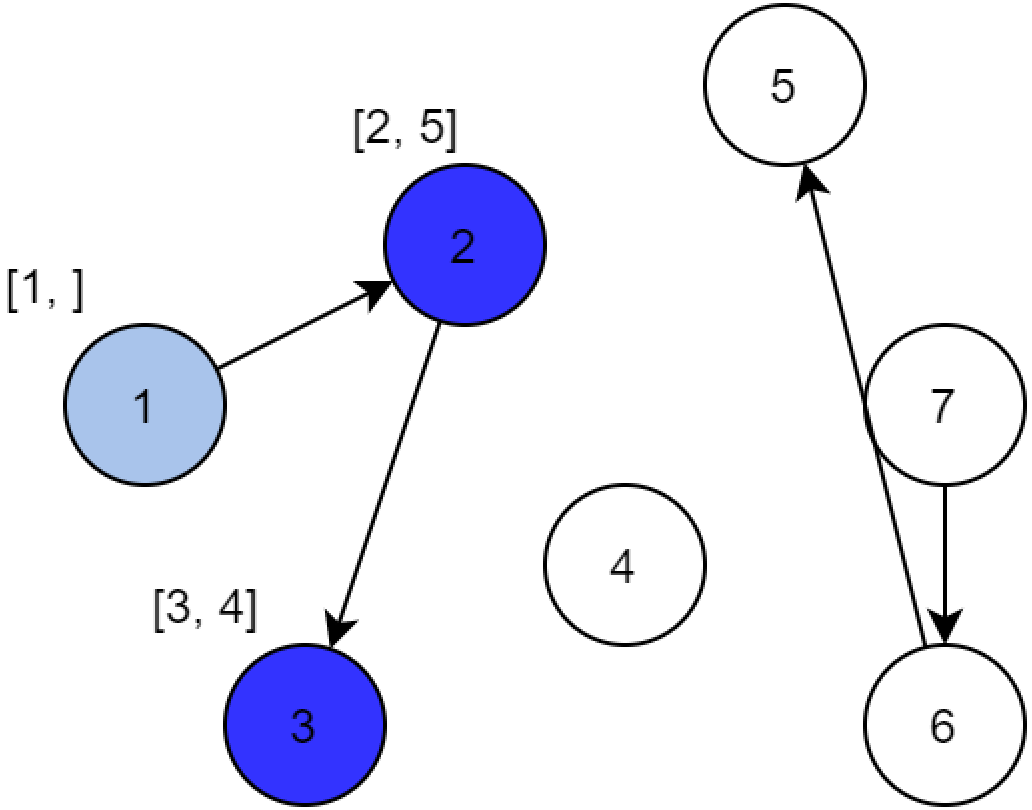
L = [ ]

PROBLEMA 1



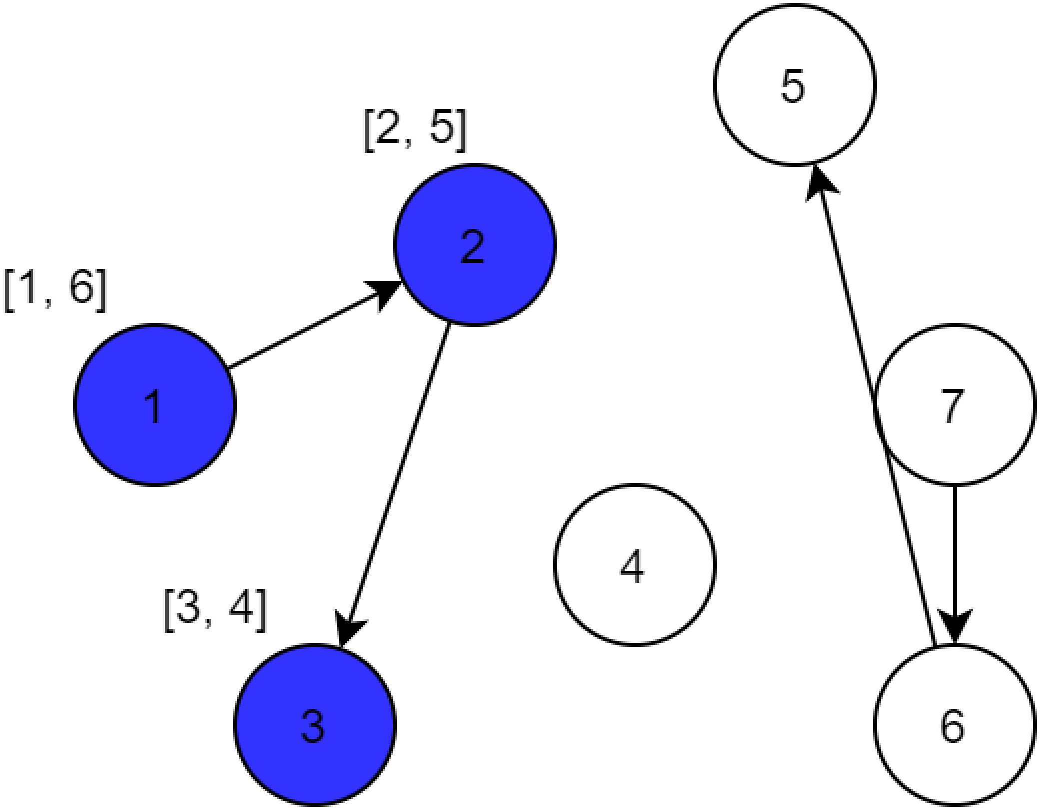
L = [3]

PROBLEMA 1



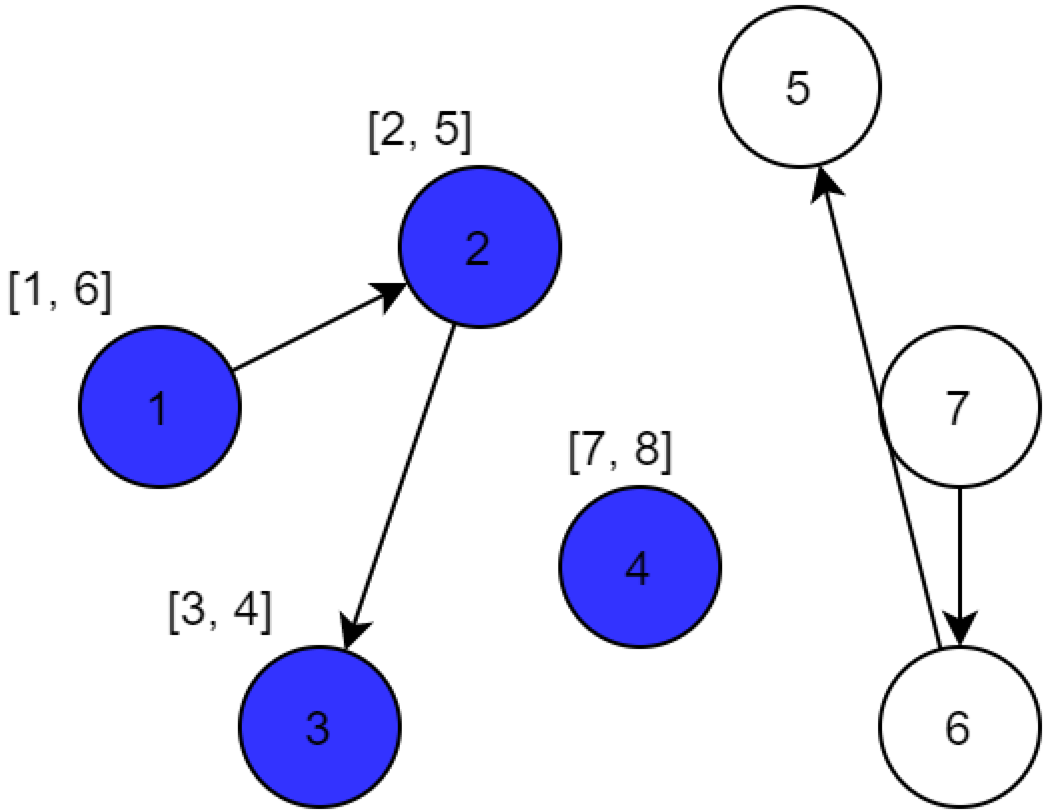
L = [2, 3]

PROBLEMA 1



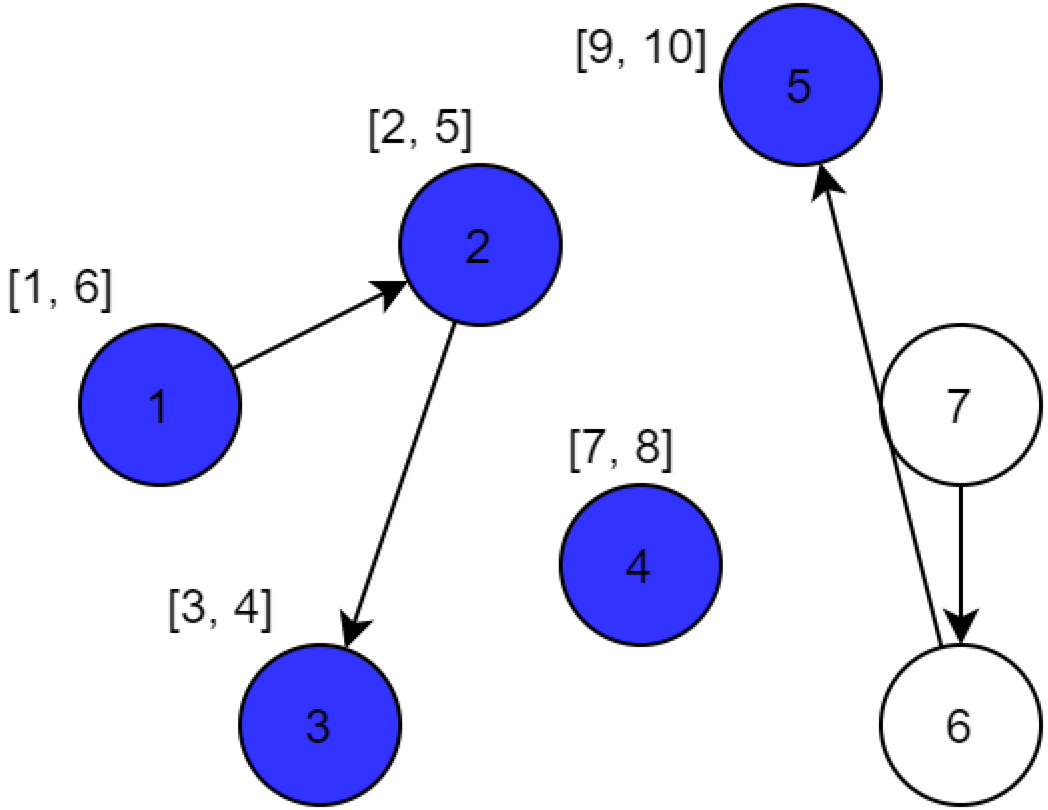
L = [1, 2, 3]

PROBLEMA 1



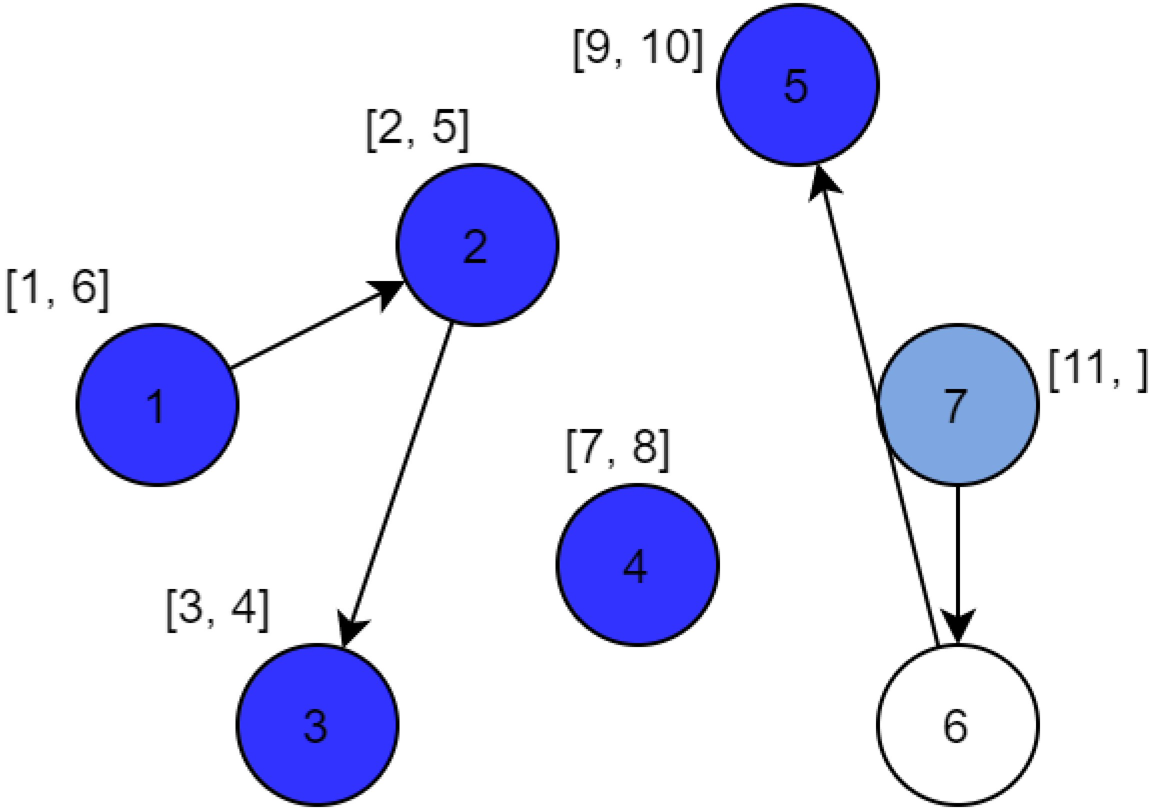
L = [4, 1, 2, 3]

PROBLEMA 1



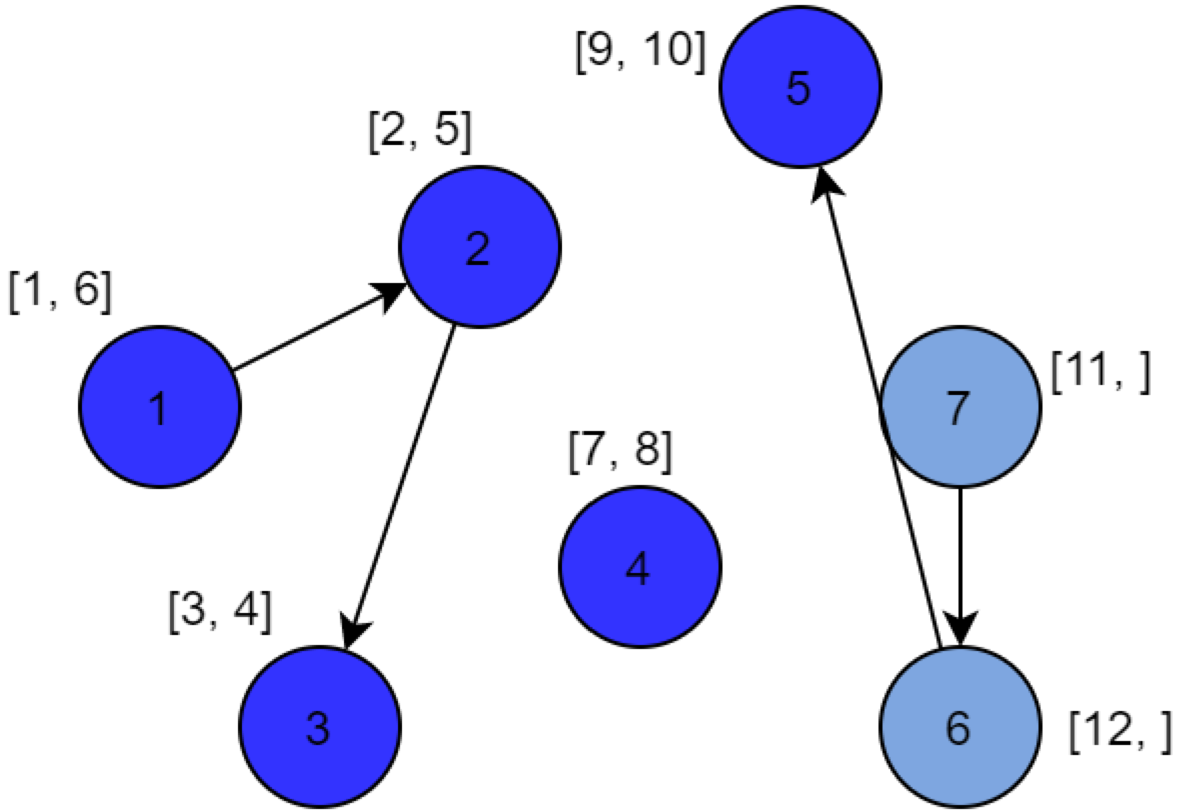
L = [5, 4, 1, 2, 3]

PROBLEMA 1



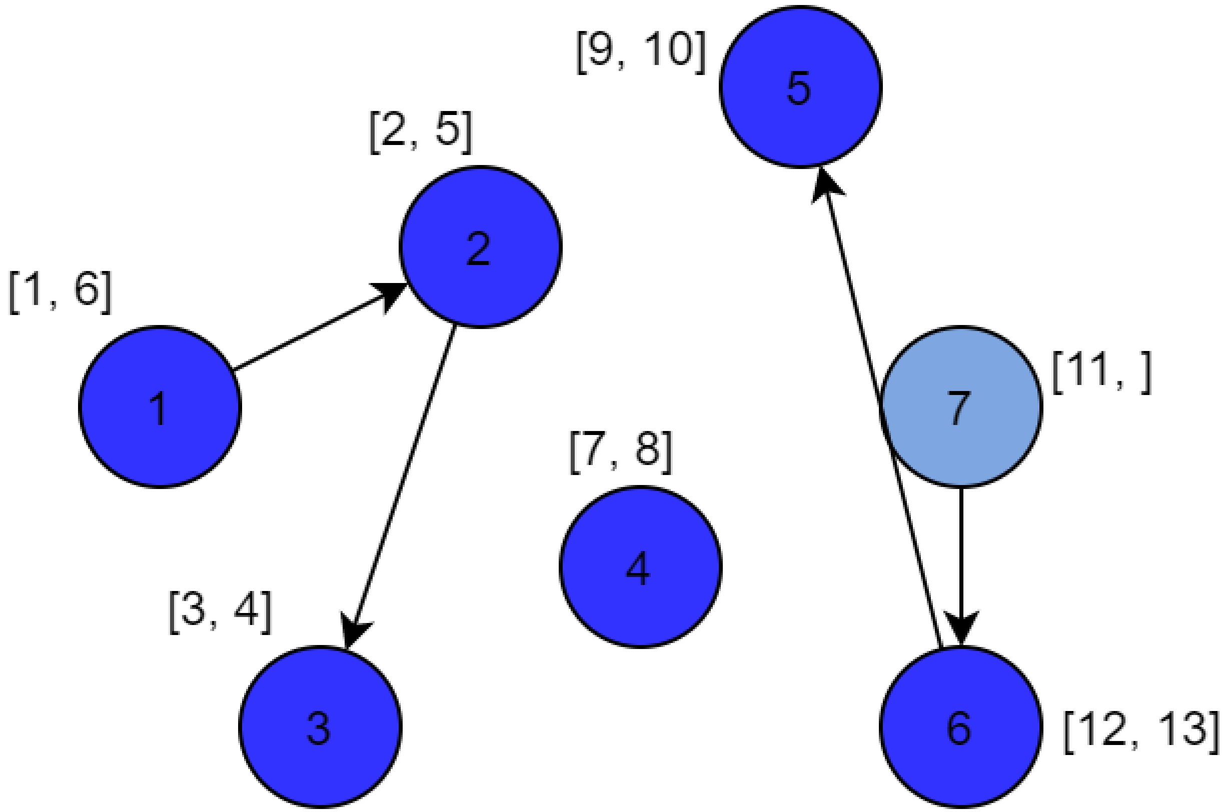
L = [5, 4, 1, 2, 3]

PROBLEMA 1



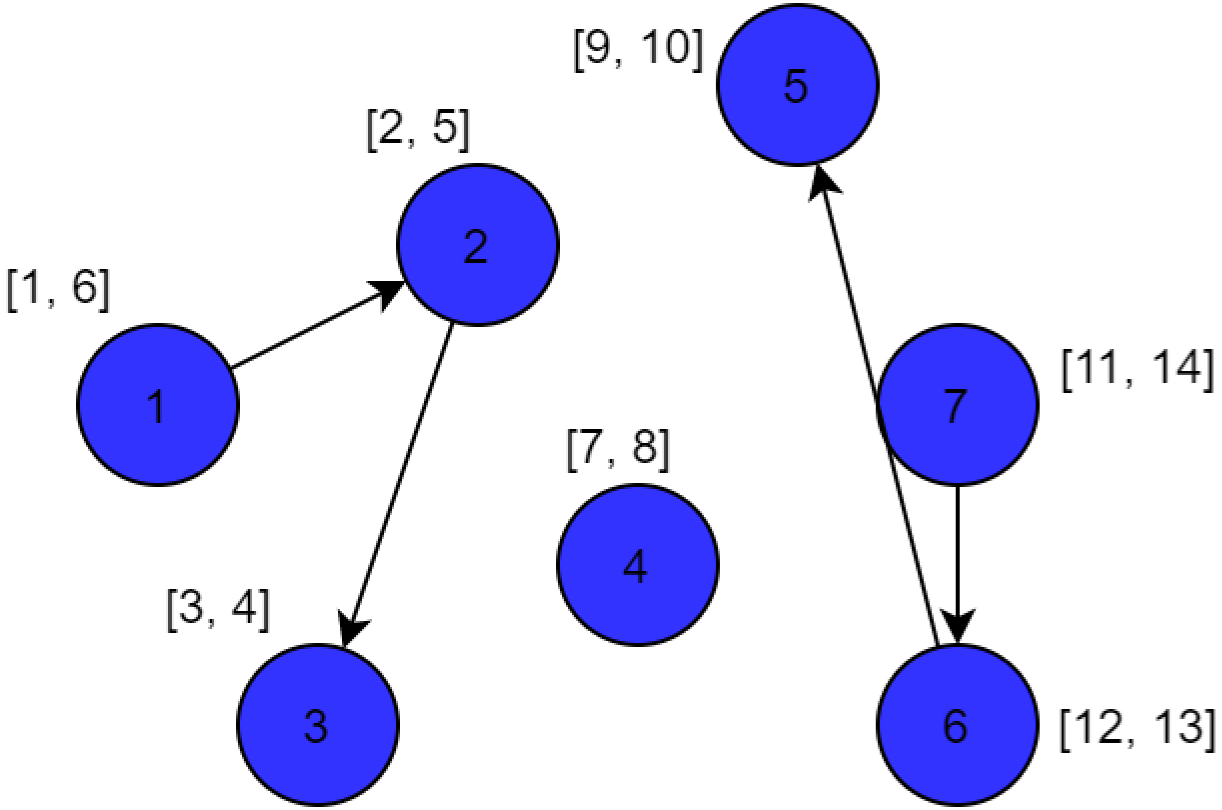
L = [5, 4, 1, 2, 3]

PROBLEMA 1



L = [6, 5, 4, 1, 2, 3]

PROBLEMA 1



L = [7, 6, 5, 4, 1, 2, 3]

PROBLEMA 1: COMPLEJIDAD

- Hay C cursos y cada curso puede tener un máximo de A actividades, por lo que el grafo tiene un máximo de $C \cdot A$ vértices. Además, hay un máximo de R requisitos por actividad, por lo que hay un máximo de $C \cdot A \cdot R$ aristas.

- La complejidad de DFS, y por ende de topSort, es O(|E| + |V|).

- La complejidad del problema será $O(C \cdot A + C \cdot A \cdot R)$

PROBLEMA 2:

$\boldsymbol{kosaraju}(\boldsymbol{G})$

Crear lista $\boldsymbol{L}$ vacía

Ejecutar $\boldsymbol{dfs}(\boldsymbol{G})$ con tiempos

Insertar vértices en $\boldsymbol{L}$ en orden descendiente de tiempos $\boldsymbol{f}$

$\boldsymbol{for\ each\ u\ in\ L}$:

$\quad \boldsymbol{assign}(\boldsymbol{u}, \boldsymbol{u})$

$\boldsymbol{assign}(\boldsymbol{u}, \boldsymbol{rep})$:

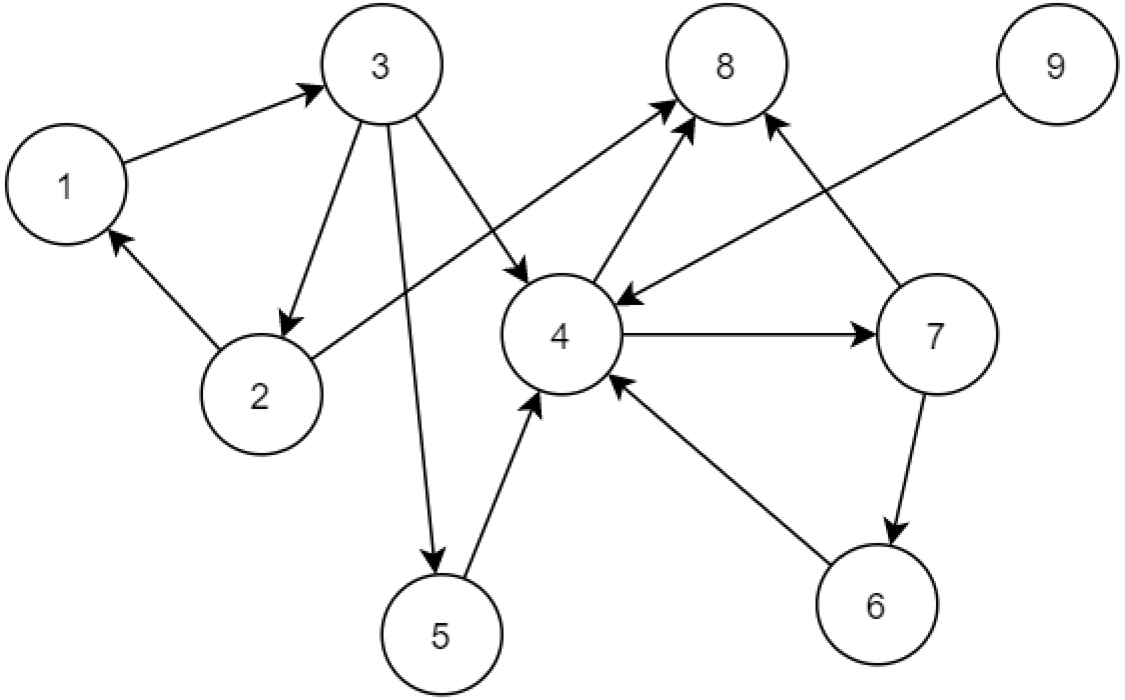
$\quad \boldsymbol{if\ u.rep} = \emptyset$:

$\quad\quad \boldsymbol{u.rep} = \boldsymbol{rep}$

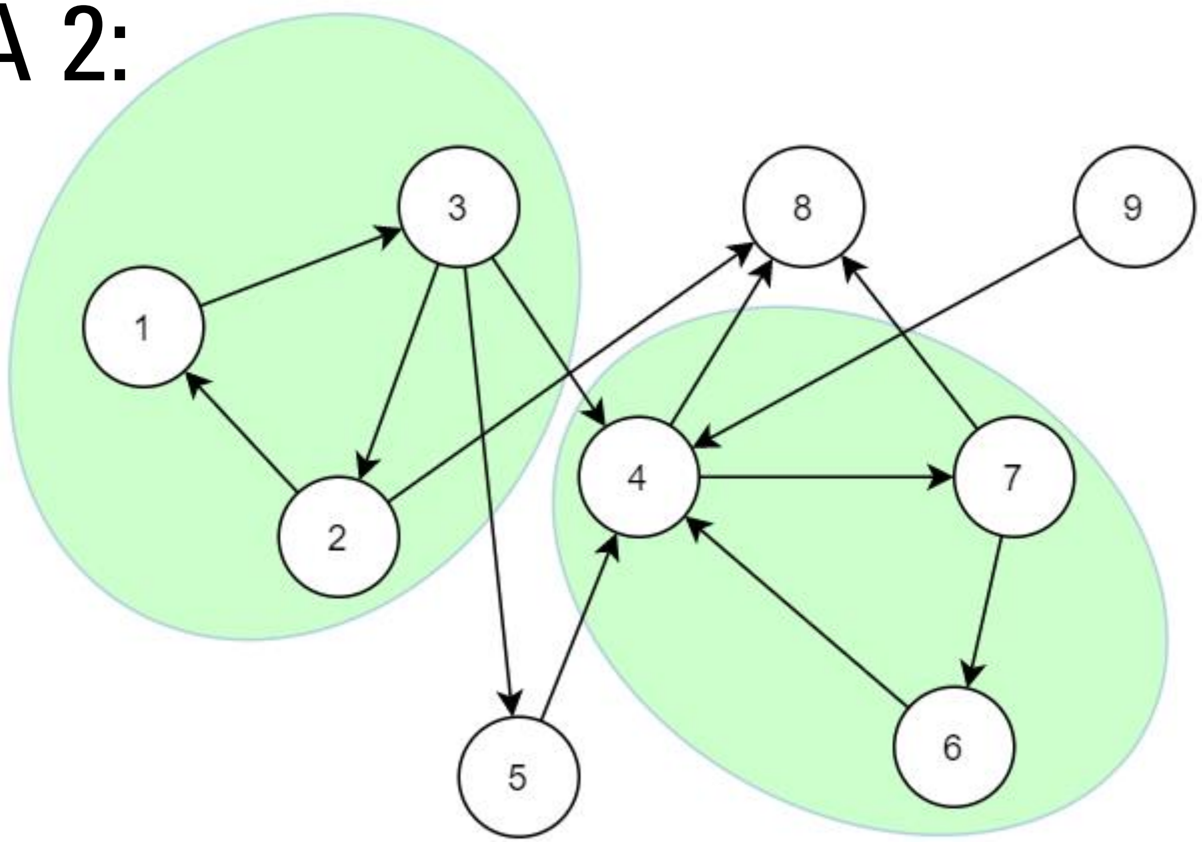
$\quad\quad \boldsymbol{foreach\ v\ in}\ \alpha'[\boldsymbol{u}]$:

$\quad\quad\quad \boldsymbol{assign}(\boldsymbol{v}, \boldsymbol{rep})$

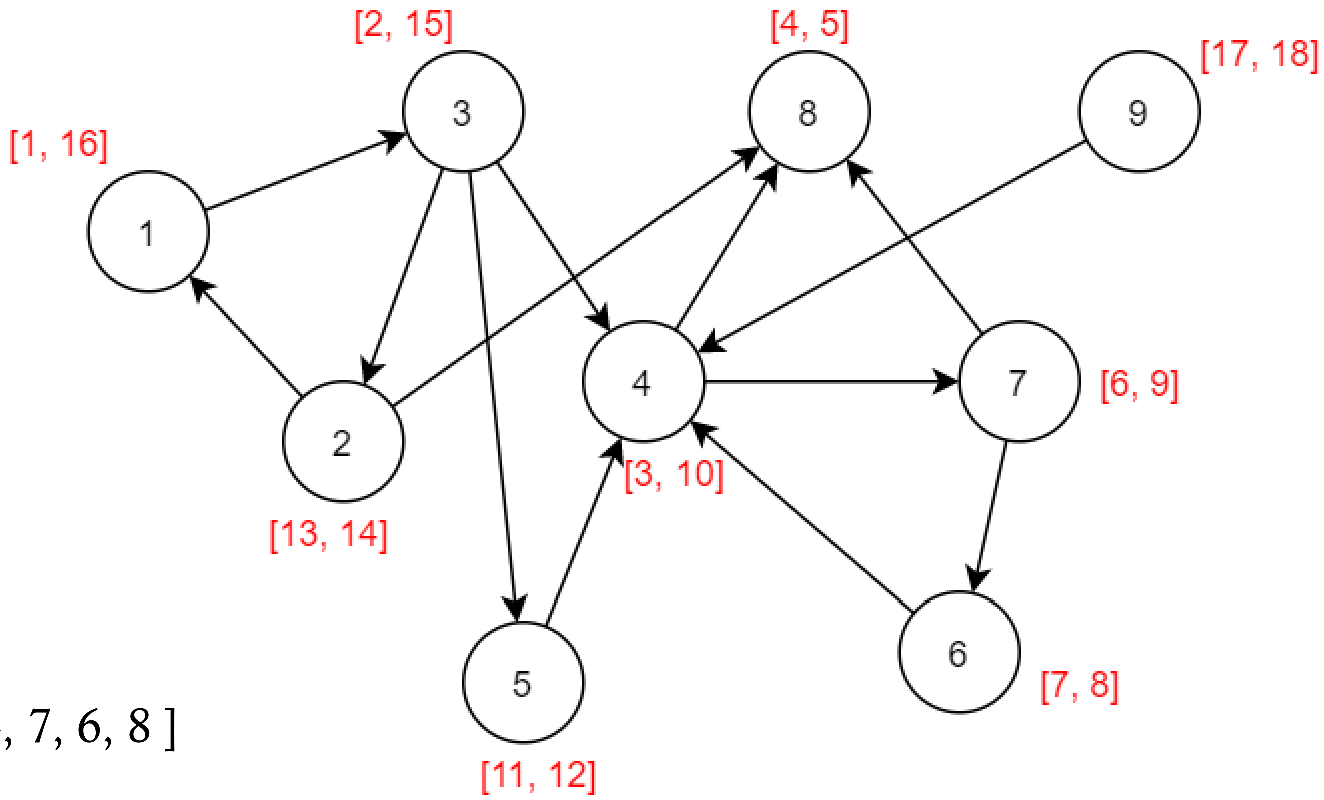
PROBLEMA 2:



PROBLEMA 2:



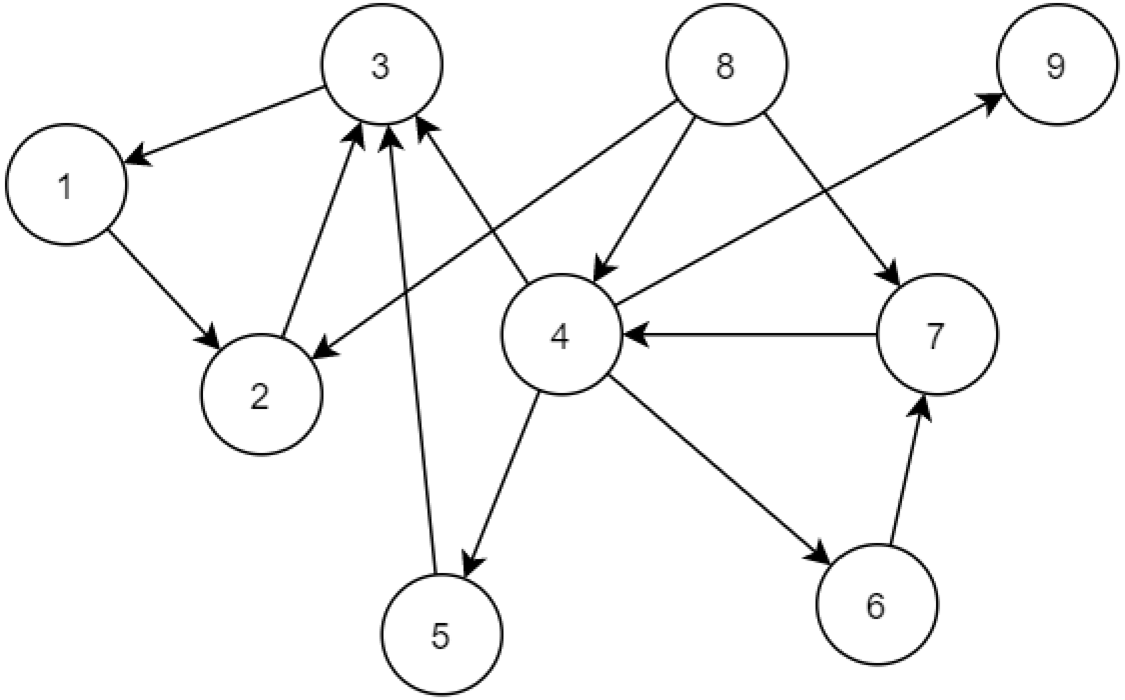
PROBLEMA 2:



L = [9, 1, 3, 2, 5, 4, 7, 6, 8]

PROBLEMA 2:

Grafo Transpuesto



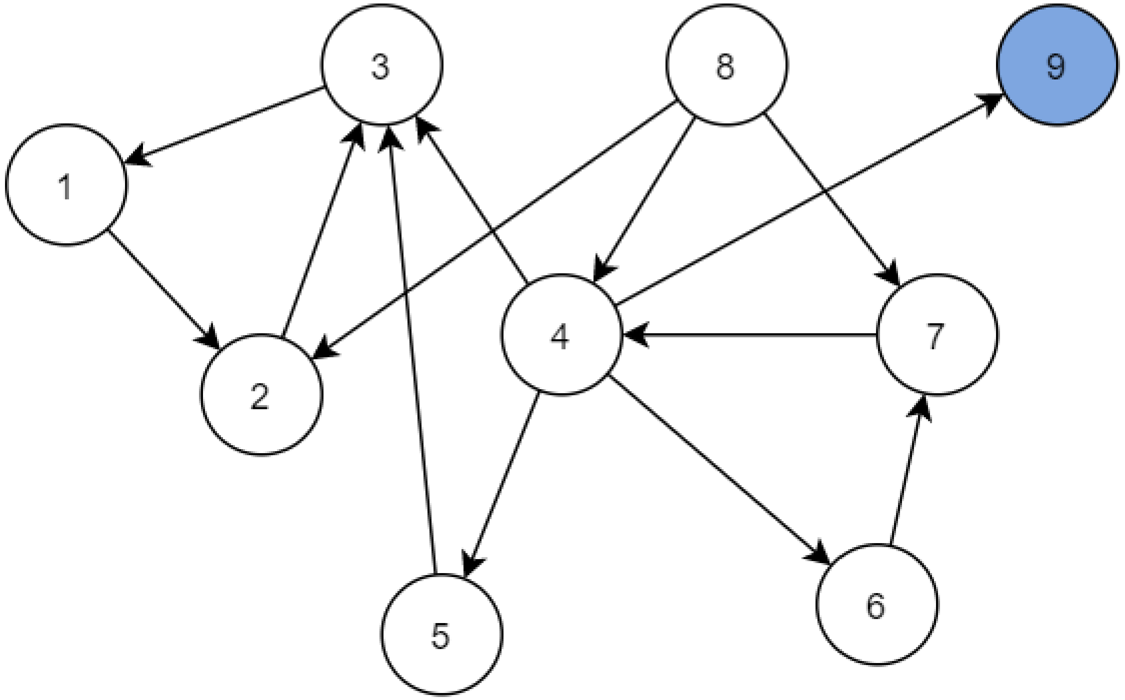
L = [9, 1, 3, 2, 5, 4, 7, 6, 8]

Sets:

1 →
2 →
3 →
4 →
5 →
6 →
7 →
8 →
9 →

PROBLEMA 2:

Grafo Transpuesto



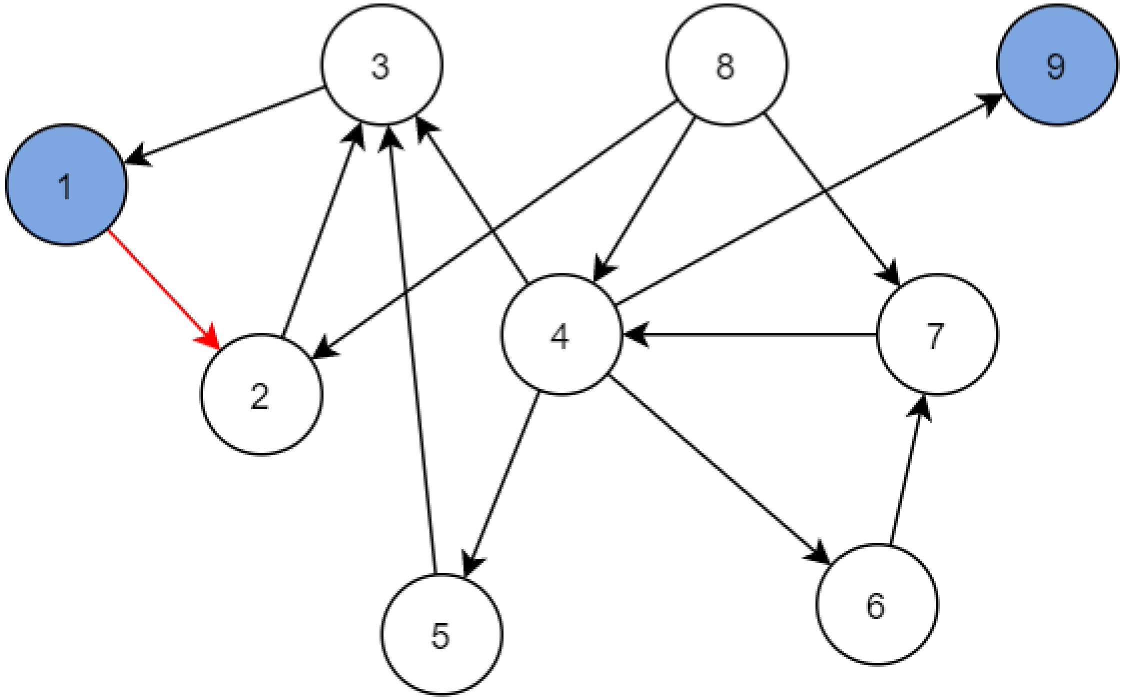
Sets:

1 →
2 →
3 →
4 →
5 →
6 →
7 →
8 →
9 → 9

L = [9, 1, 3, 2, 5, 4, 7, 6, 8]

PROBLEMA 2:

Grafo Transpuesto



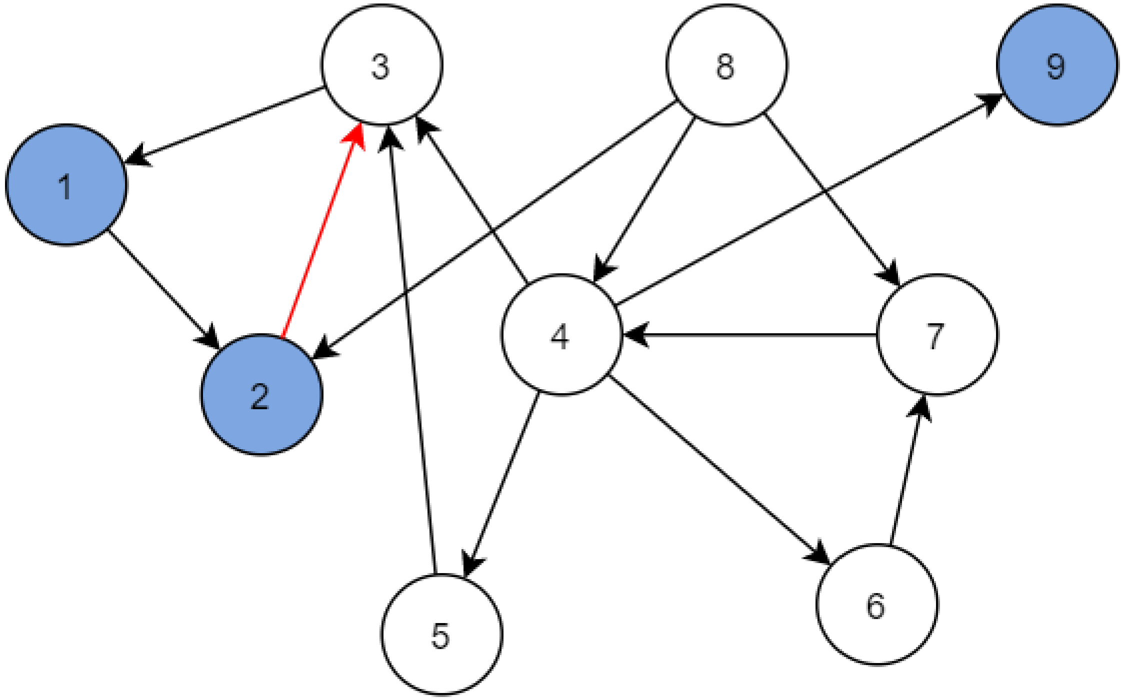
L = [9, 1, 3, 2, 5, 4, 7, 6, 8]

Sets:

1 → 1
2 →
3 →
4 →
5 →
6 →
7 →
8 →
9 → 9

PROBLEMA 2:

Grafo Transpuesto



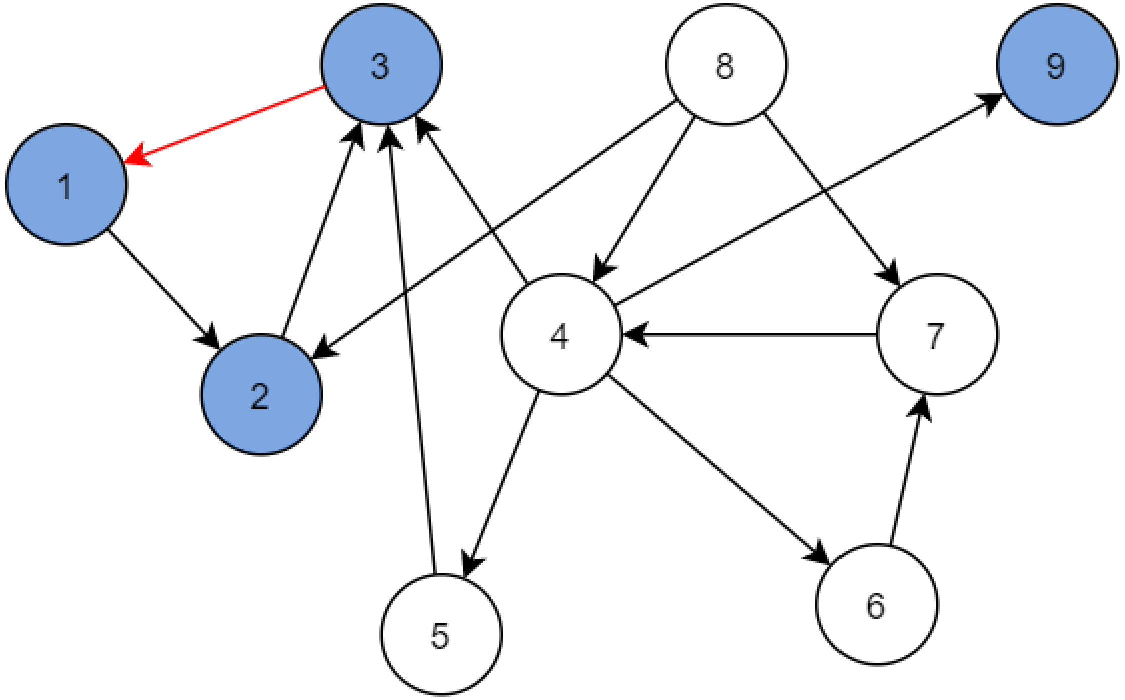
Sets:

1 → 1
2 → 1
3 →
4 →
5 →
6 →
7 →
8 →
9 → 9

L = [9, 1, 3, 2, 5, 4, 7, 6, 8]

PROBLEMA 2:

Grafo Transpuesto



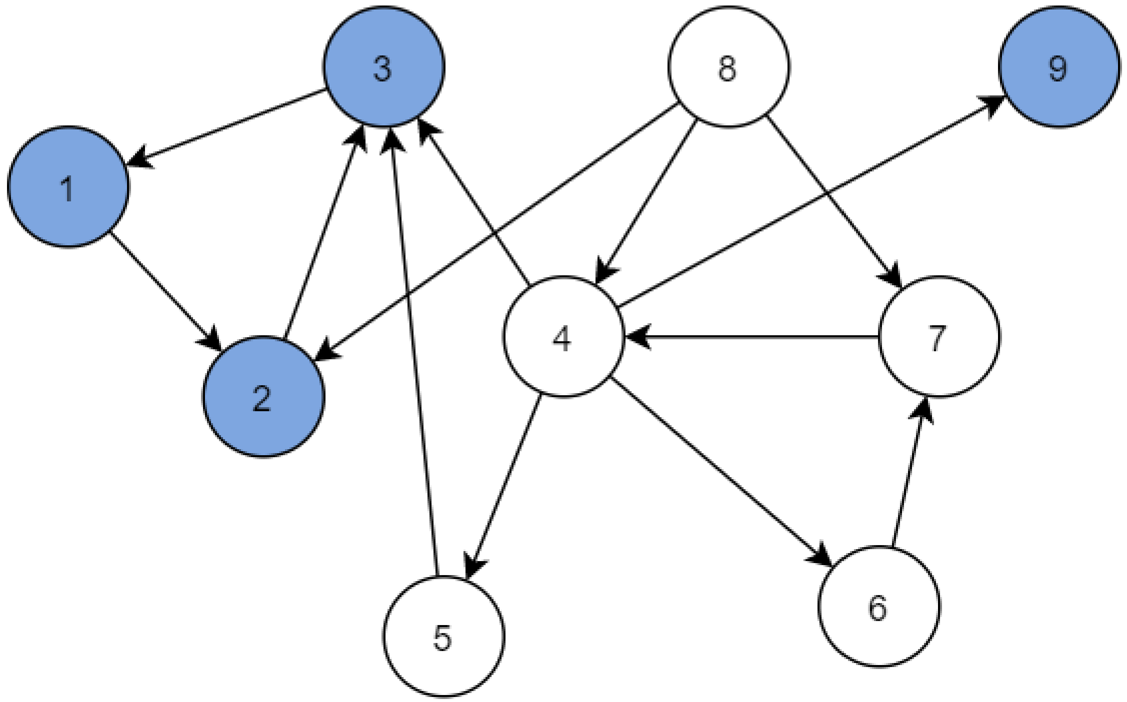
L = [9, 1, 3, 2, 5, 4, 7, 6, 8]

Sets:

1 → 1
2 → 1
3 → 1
4 →
5 →
6 →
7 →
8 →
9 → 9

PROBLEMA 2:

Grafo Transpuesto



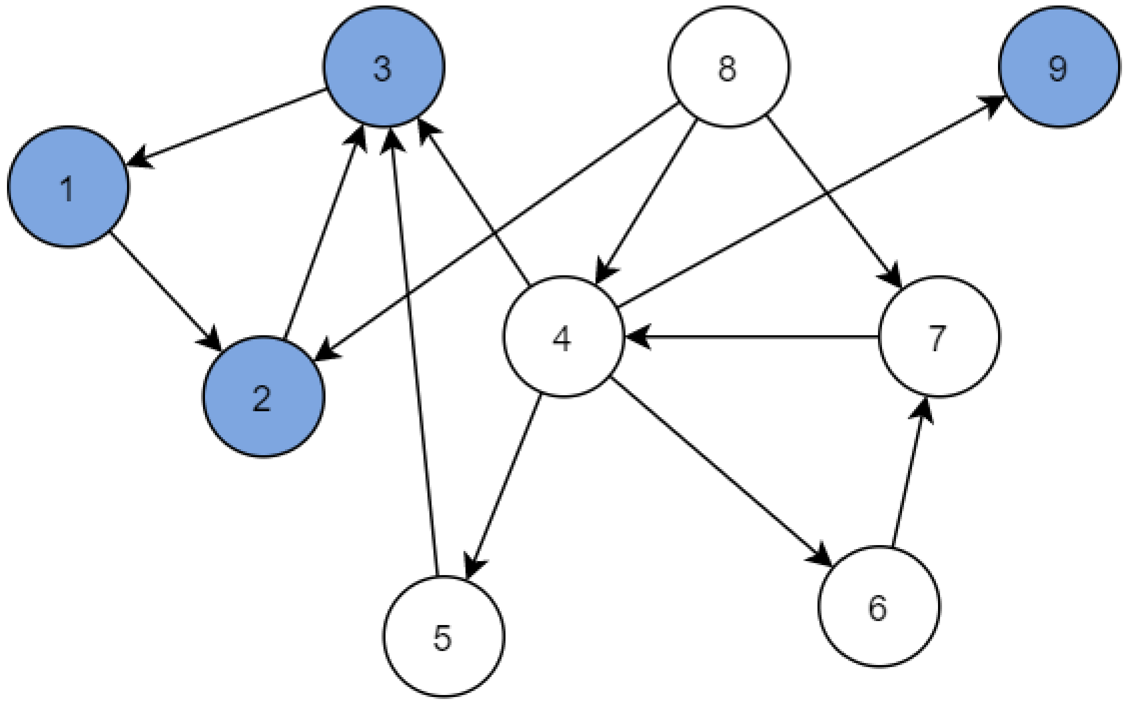
L = [9, 1, 3, 2, 5, 4, 7, 6, 8]

Sets:

1 → 1
2 → 1
3 → 1
4 →
5 →
6 →
7 →
8 →
9 → 9

PROBLEMA 2:

Grafo Transpuesto



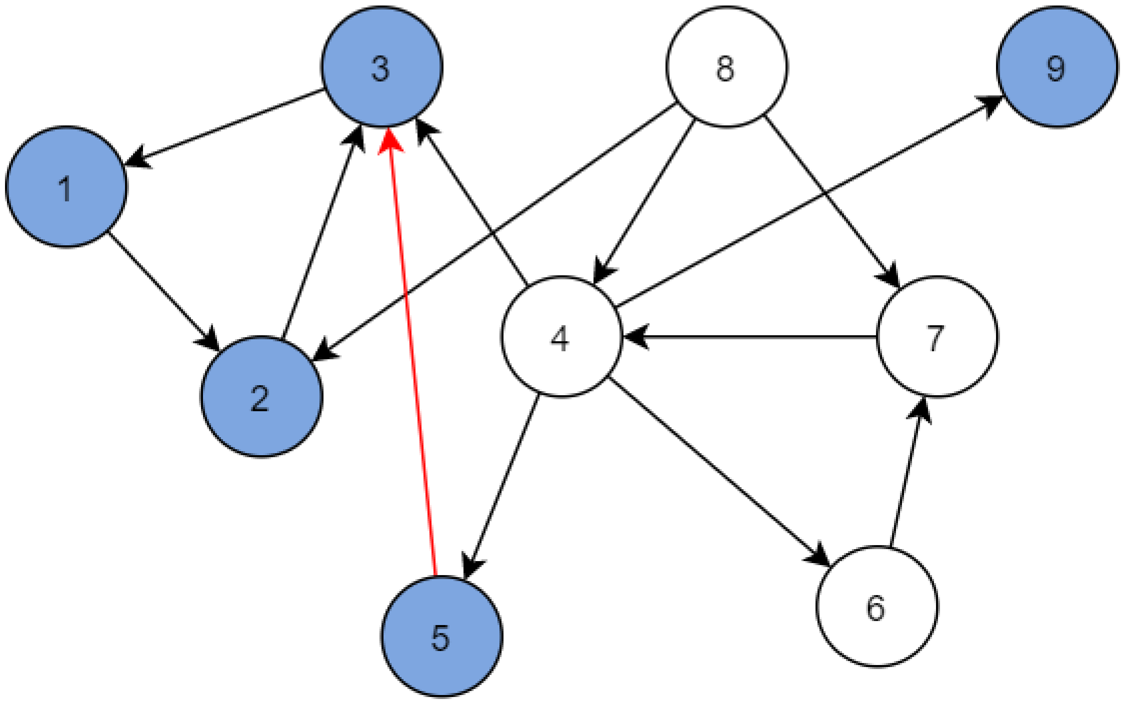
Sets:

1 → 1
2 → 1
3 → 1
4 →
5 →
6 →
7 →
8 →
9 → 9

L = [9, 1, 3, 2, 5, 4, 7, 6, 8]

PROBLEMA 2:

Grafo Transpuesto



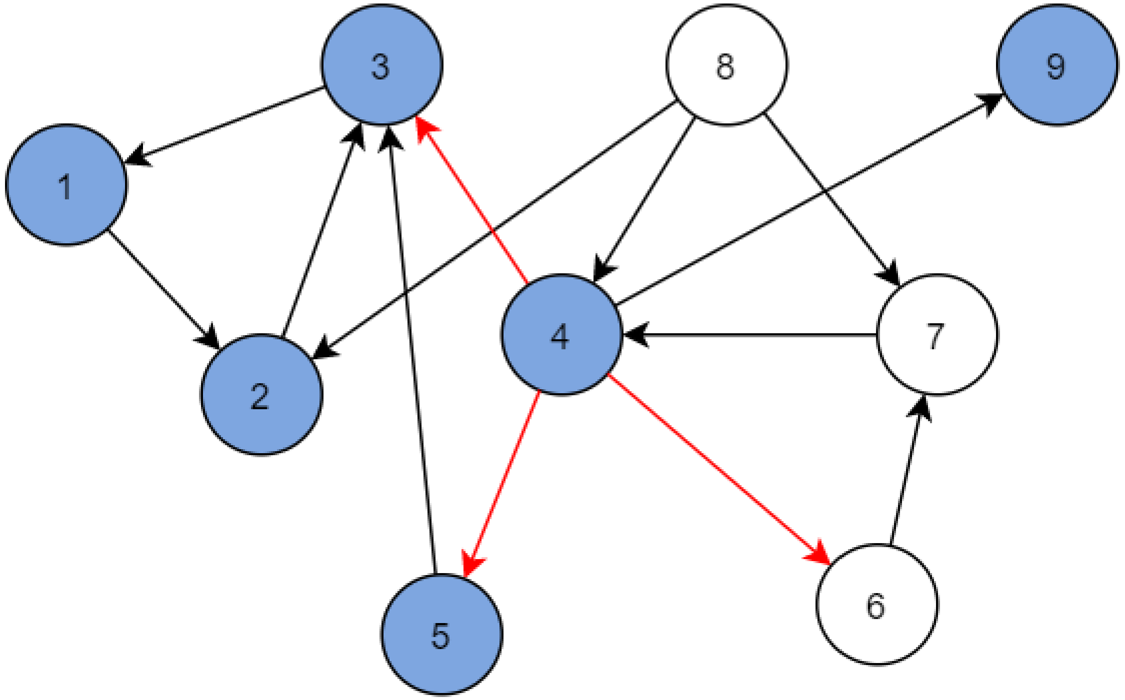
L = [9, 1, 3, 2, 5, 4, 7, 6, 8]

Sets:

1 → 1
2 → 1
3 → 1
4 →
5 → 5
6 →
7 →
8 →
9 → 9

PROBLEMA 2:

Grafo Transpuesto



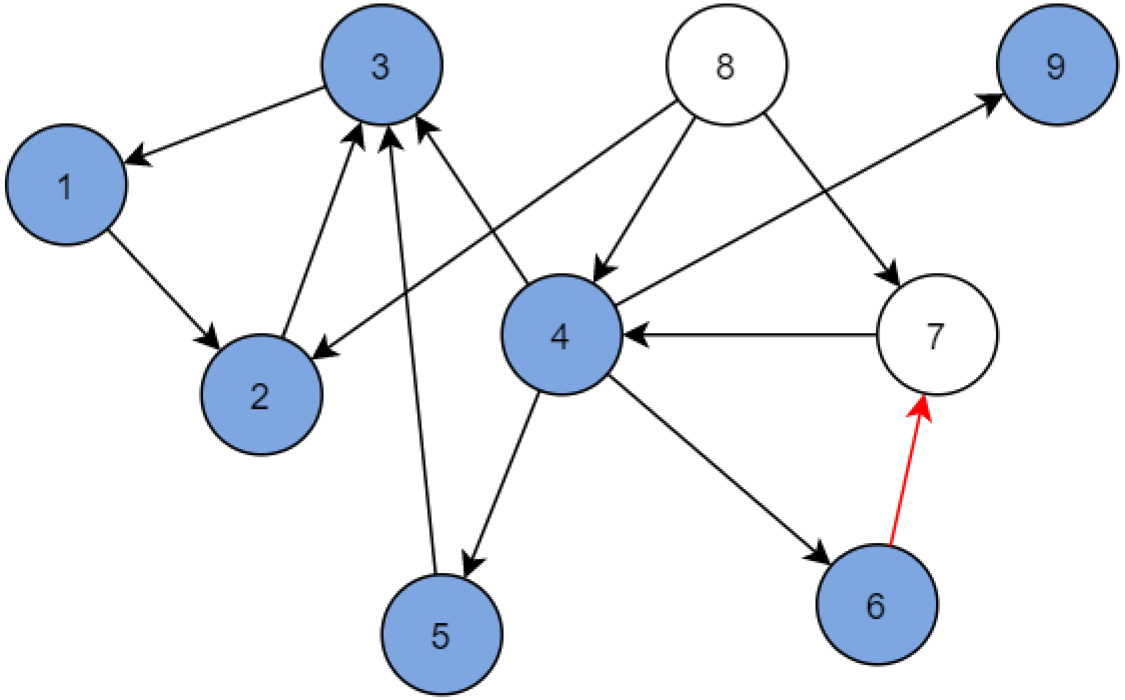
L = [9, 1, 3, 2, 5, 4, 7, 6, 8]

Sets:

1 → 1
2 → 1
3 → 1
4 → 4
5 → 5
6 →
7 →
8 →
9 → 9

PROBLEMA 2:

Grafo Transpuesto



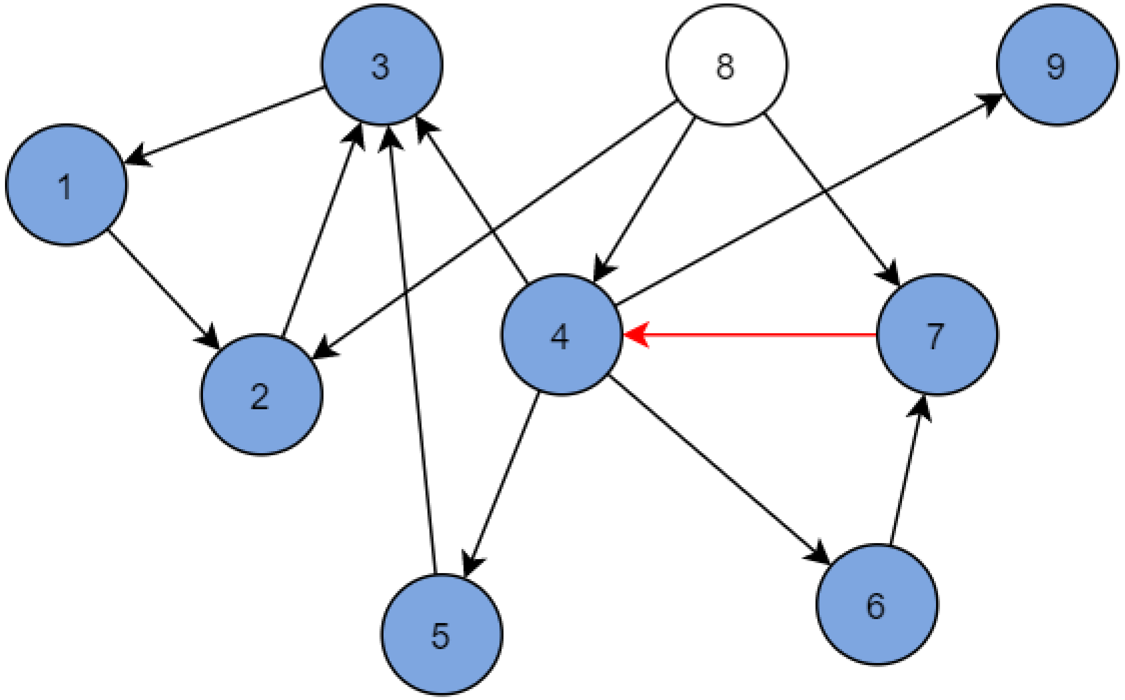
L = [9, 1, 3, 2, 5, 4, 7, 6, 8]

Sets:

1 → 1
2 → 1
3 → 1
4 → 4
5 → 5
6 → 4
7 →
8 →
9 → 9

PROBLEMA 2:

Grafo Transpuesto



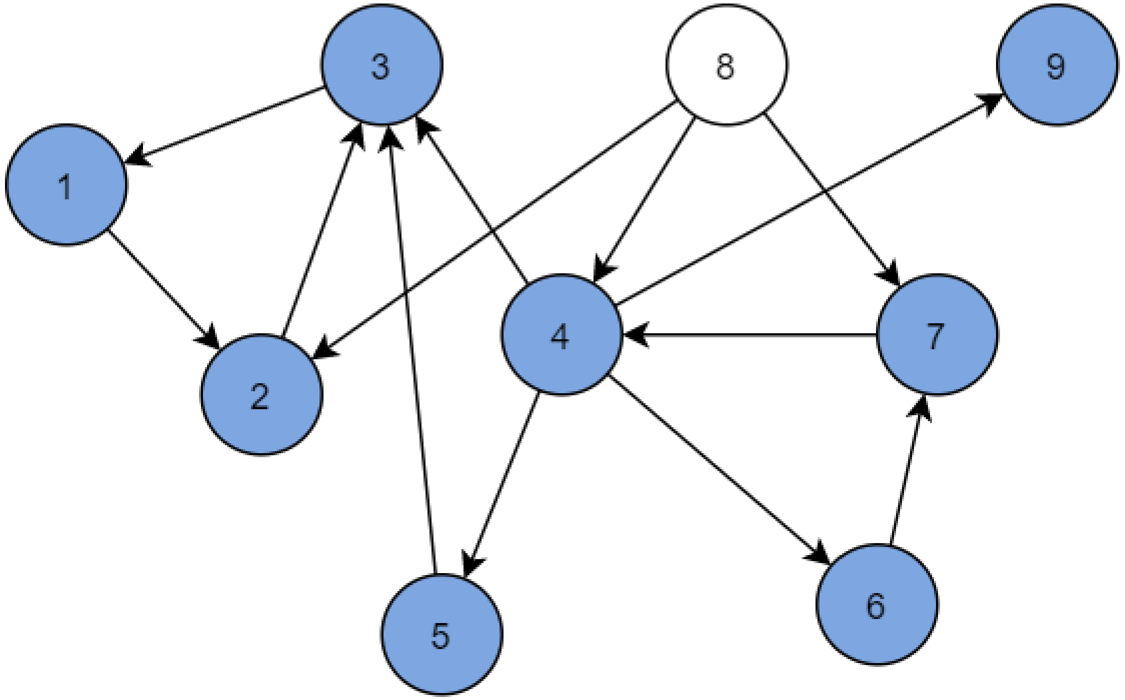
L = [9, 1, 3, 2, 5, 4, 7, 6, 8]

Sets:

1 → 1
2 → 1
3 → 1
4 → 4
5 → 5
6 → 4
7 → 4
8 →
9 → 9

PROBLEMA 2:

Grafo Transpuesto



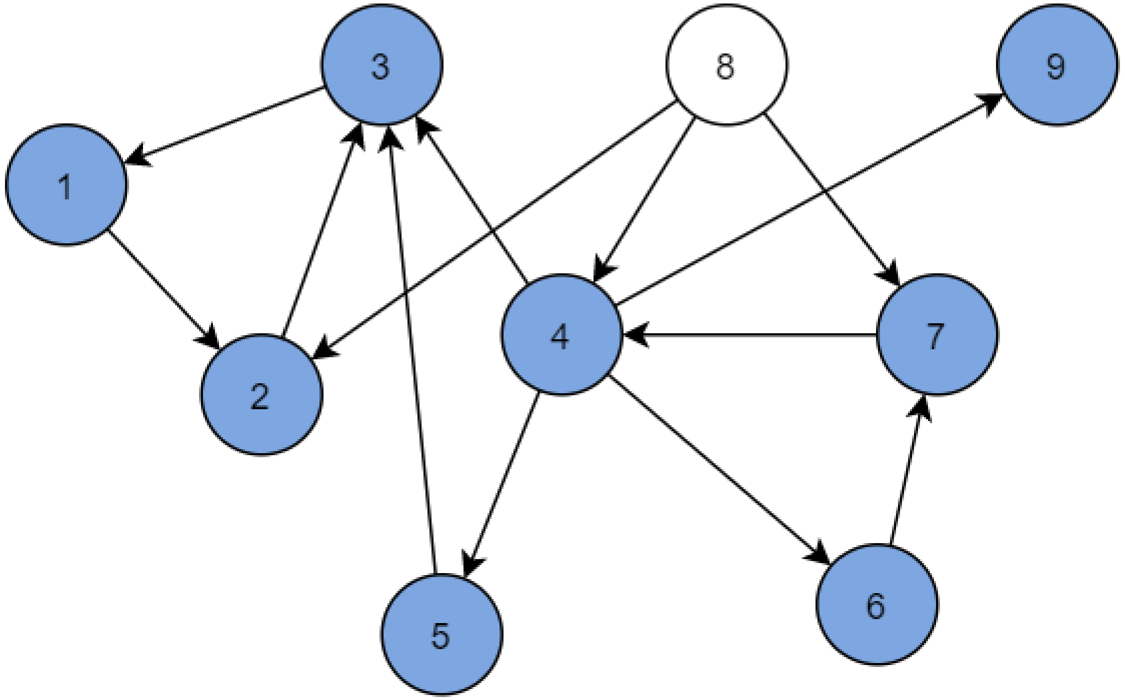
Sets:

1 → 1
2 → 1
3 → 1
4 → 4
5 → 5
6 → 4
7 → 4
8 →
9 → 9

L = [9, 1, 3, 2, 5, 4, 7, 6, 8]

PROBLEMA 2:

Grafo Transpuesto



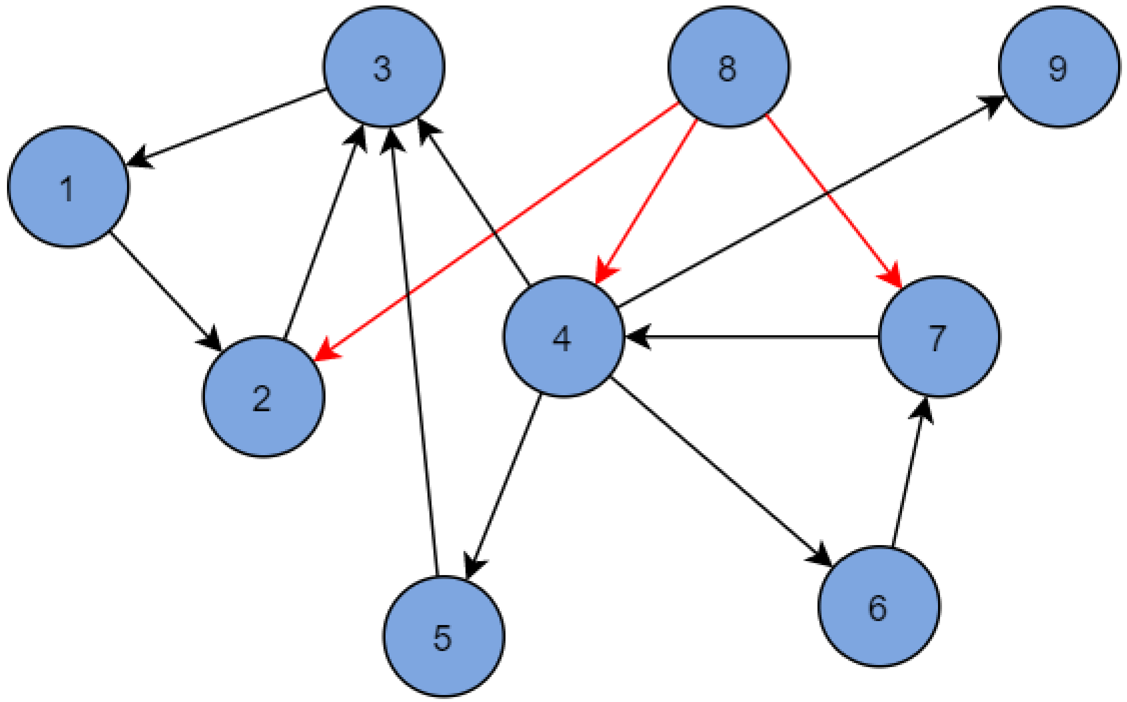
L = [9, 1, 3, 2, 5, 4, 7, 6, 8]

Sets:

1 → 1
2 → 1
3 → 1
4 → 4
5 → 5
6 → 4
7 → 4
8 →
9 → 9

PROBLEMA 2:

Grafo Transpuesto



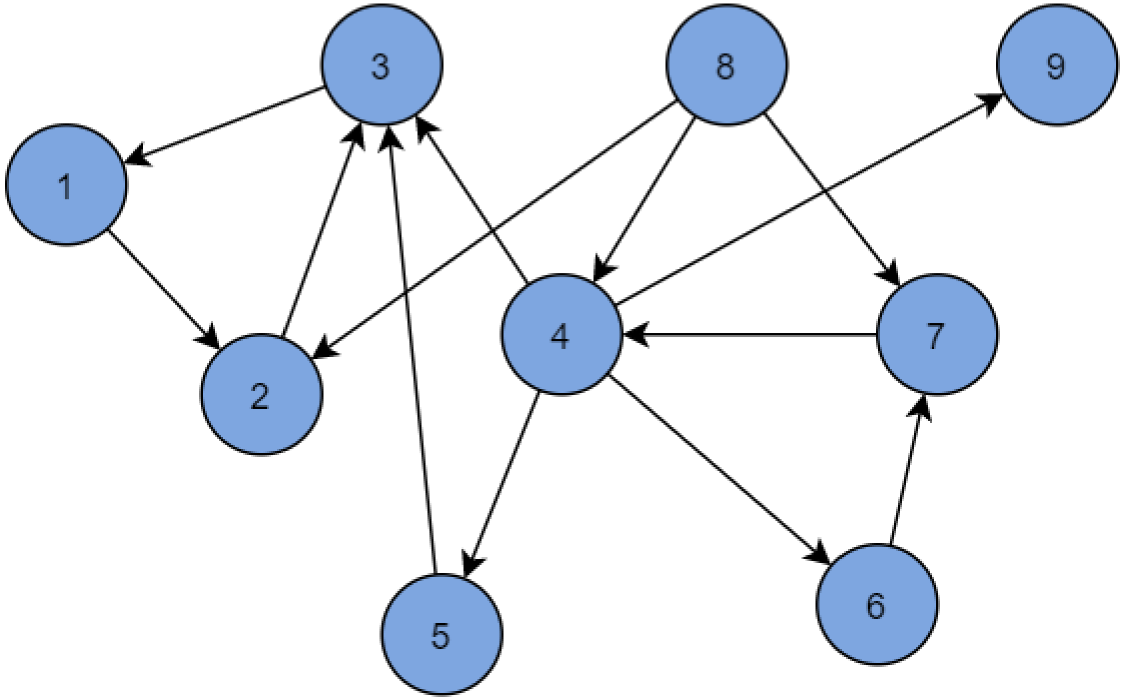
Sets:

1 → 1
2 → 1
3 → 1
4 → 4
5 → 5
6 → 4
7 → 4
8 → 8
9 → 9

L = [9, 1, 3, 2, 5, 4, 7, 6, 8]

PROBLEMA 2:

Grafo Transpuesto



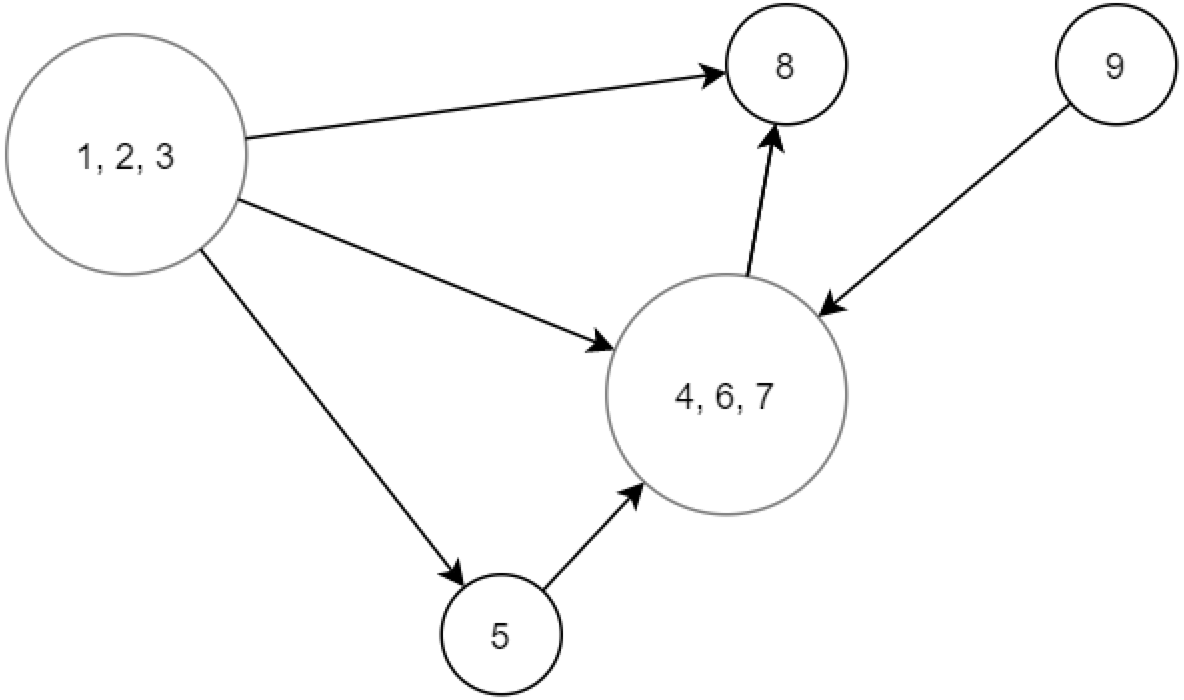
Sets:

1 → 1
2 → 1
3 → 1
4 → 4
5 → 5
6 → 4
7 → 4
8 → 8
9 → 9

L = [9, 1, 3, 2, 5, 4, 7, 6, 8]

PROBLEMA 2:

Grafo Original
Colapsado

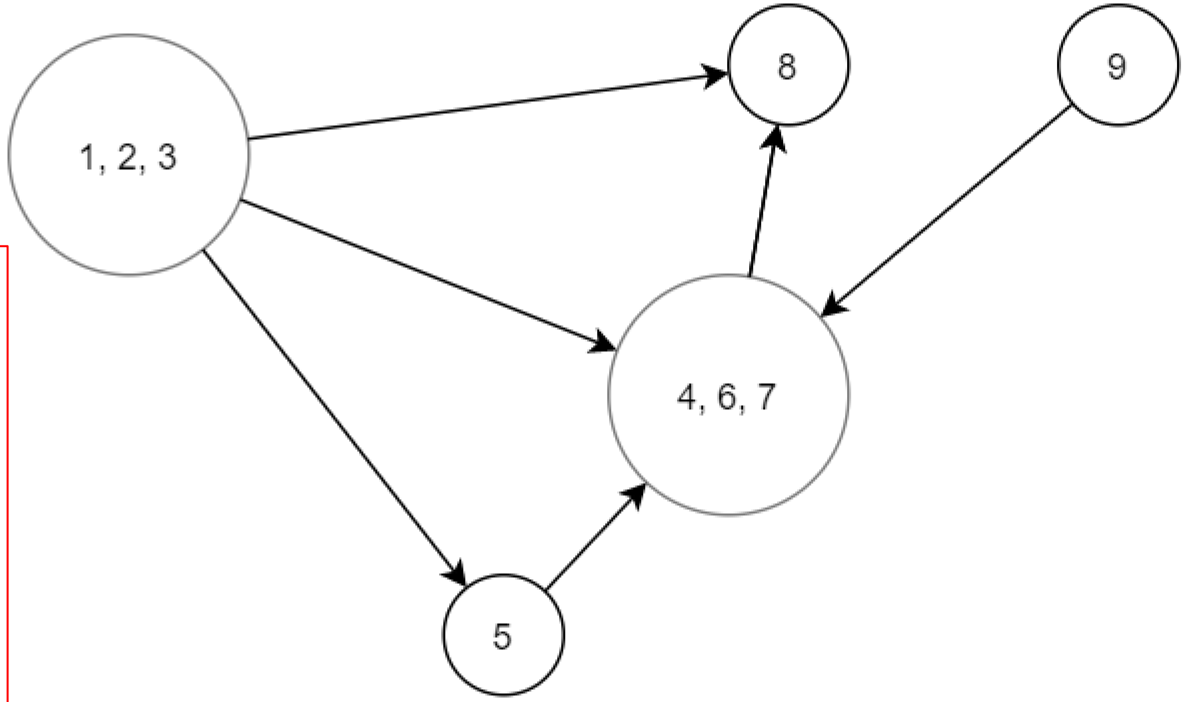


Sets:

1 → 1
2 → 1
3 → 1
4 → 4
5 → 5
6 → 4
7 → 4
8 → 8
9 → 9

PROBLEMA 2:

Grafo Original
Colapsado



Solo los puntos 1, 2, 3, 4, 6 y 7 pertenecen a componentes fuertemente conexas de una cantidad mayor a 1, por lo que solo desde estos puntos existirá un camino especial.

Sets:

1 → 1
2 → 1
3 → 1
4 → 4
5 → 5
6 → 4
7 → 4
8 → 8
9 → 9

PROBLEMA 2: COMPLEJIDAD

- Kosaraju tiene una complejidad de $O(|E| + |V|)$.
- Se tienen P vertices y C aristas.
- El observer la cantidad de elementos de los sets es O(1) si se guarda la información mientras se crean, sino, hay que contar todos los vertices lo que es O(P).
- La complejidad es O(P + C)