1st problem

We assume that for given a diredcted graph G, each edge is labeled by an element of some closed semiring and there is special vertices ‘u’ and ‘v’.

The labels come from S where (S, +, •, 0, 1).

This is the algorithm that compute c(u,v) for all possible pair of vertices u and v that u, v ∈ V[G]

```
COMPUTE-PATH(G, u, v)
c(u, v)=0 //this “0” is an element of some closed semiring
do if u = v
  c(u, v) ← 1 //this “1” is an element of some closed semiring
for each vertex k ∈ V[G] - {u}
  do color[k] ← WHITE
      path[k] ← 0 //this “0” is an element of some closed semiring
color[u] ← GRAY
Q ← ∅
ENQUEUE(Q, u)
while Q ≠ ∅
do k ← DEQUEUE(Q)
   for each a ∈ Adj[k]
       do if color[a] = WHITE
             then color[a] ← GRAY
             if k = u
                then path[a] ← Label[(k, a)]
             else
                then if path[a] = 0 //this “0” is an element of some closed semiring
                        then path[a] ← •(path[k], Label[(k, a)])
                       else
                        then path[a] ← +(path[a], •(path[k], Label[(k, a)]))
            if a ≠ v
              then ENQUEUE(Q, a)
   color[k] ← BLACK
if c(u, v) = 1
  then c(u, v) ← +(c(u, v), path[v])
else
  then c(u, v) ← path[v]
RERUTN c(u, v)
```

We can compute **c(u, v) by using** this algorithm

2nd problem

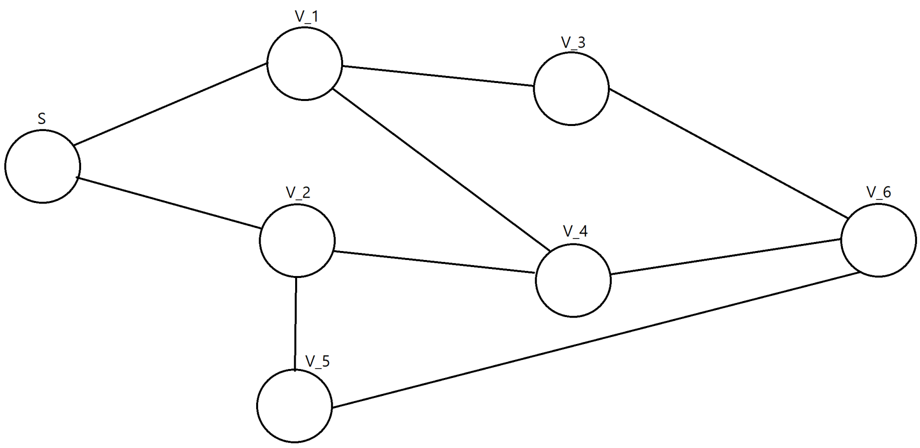
```
COMPUTE-2-COLORING(G, s)
for each vertex u ∈ V[G]
  do color[u] ← WHITE
color[s] ← RED
Q ← ∅
```

```
5 ENQUEUE(Q, u)
6 while Q ≠ ∅
7 do u ← DEQUEUE(Q)
8    for each v ∈ Adj[u]
9       do if color[v] = WHITE
10               then if color[u] = RED
11                       then color[v] ← BLUE
12                    else
13                       then color[v] ← RED
14                    ENQUEUE(Q, v)
15          else if color[v] = color[u]
16                    then RETURN NO
17 RETURN YES
```

The algorithm COMPUTE-2-COLORING is made by applying BFS.

We can solve 2-coloring problem by using this algorithm.

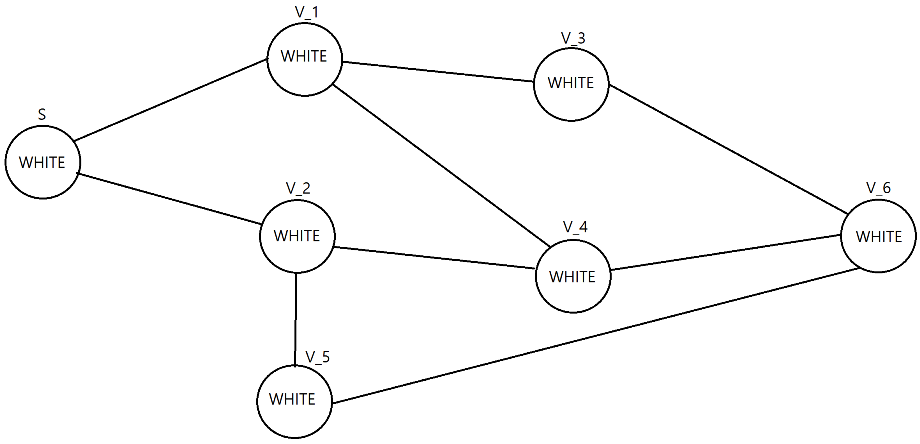
I`m going to explain how this algorithm is done with example.



The picture above is a undirected graph G that is $|V| \geq 2$, $|E| \geq 1$

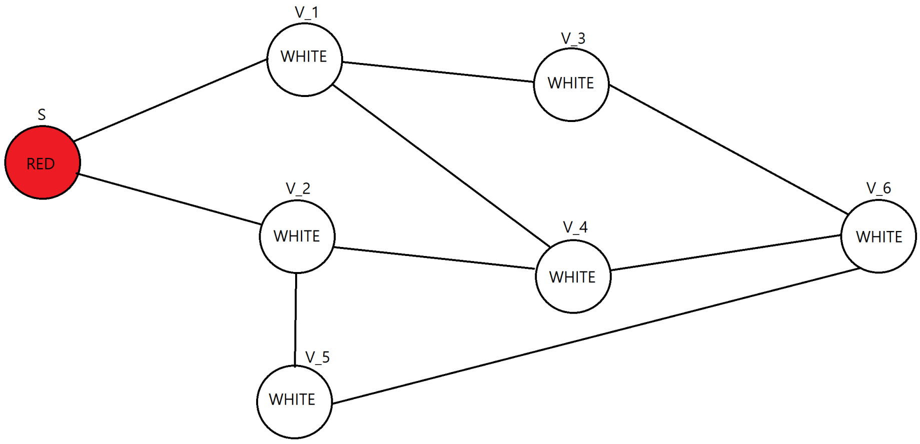
After running line 1-2, all vertices of G(S, v_1, v_2, v_3, v_4, v_5, v_6) are colored by WHITE

Graph G is as follows.



After running line 3-5, vertex S is colored by RED, QUEUE Q is initialized, and the vertex S is enqueued. So Q=(S)

Graph G is as follows.



We are in the while loop at 6-16 because Q is not empty.

Dequeue S of Q.

Adjacent vertices of S are v_1, v_2

v_1 is colored with WHITE and S is colored with RED

→ Color v_1 with BLUE

→ Enqueue v_1

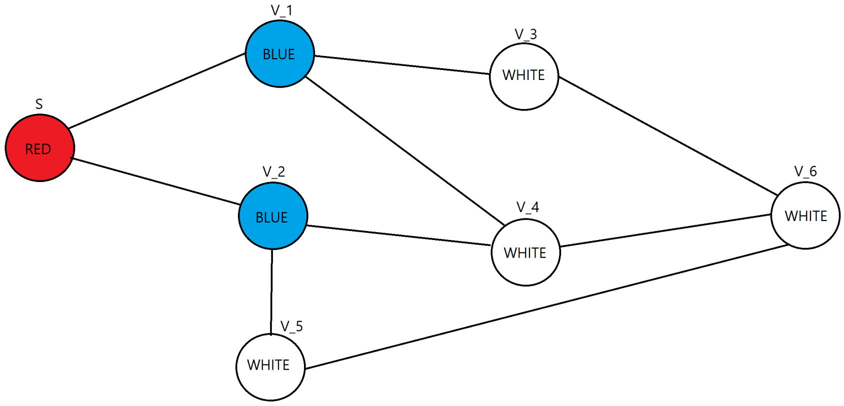
v_2 is colored with WHITE and S is colored with RED

→ Color v_2 with BLUE

→ Enqueue v_2

Q is (v_1, v_2)

Graph G is as follows.



We are in the while loop at 6-16 because Q is not empty.

Dequeue v_1 of Q.

Adjacent vertices of S are S, v_3, v_4

S is not colored with WHITE and colors of S and v_1 are not same

v_3 is colored with WHITE and v_1 is colored with BLUE

→ Color v_3 with RED

→ Enqueue v_3

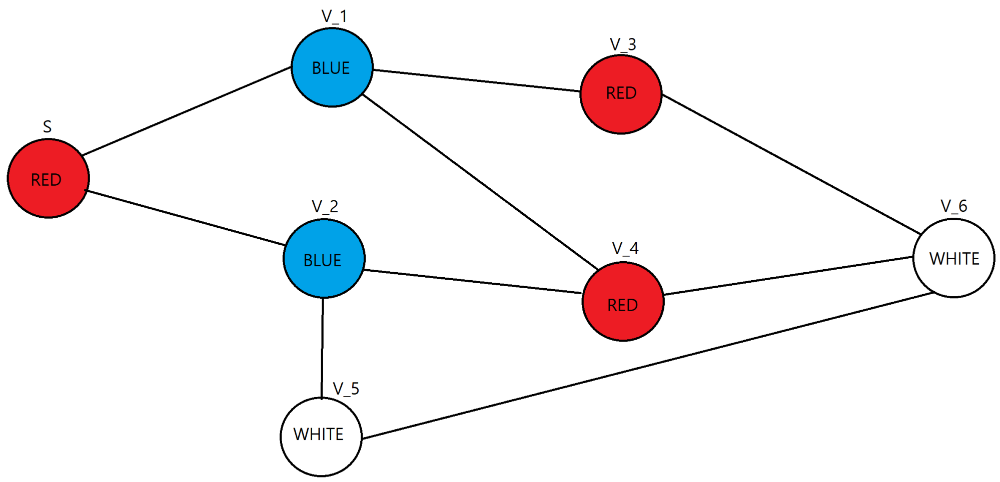
v_4 is colored with WHITE and v_1 is colored with BLUE

→ Color v_4 with RED

→ Enqueue v_4

Q is (v_2, v_3, v_4)

Graph G is as follows.



We are in the while loop at 6-16 because Q is not empty.

Dequeue v_2 of Q.

Adjacent vertices of S are S, v_4, v_5

S is not colored with WHITE and colors of S and v_2 are not same

v_4 is not colored with WHITE and colors of v_4 and v_2 are not same

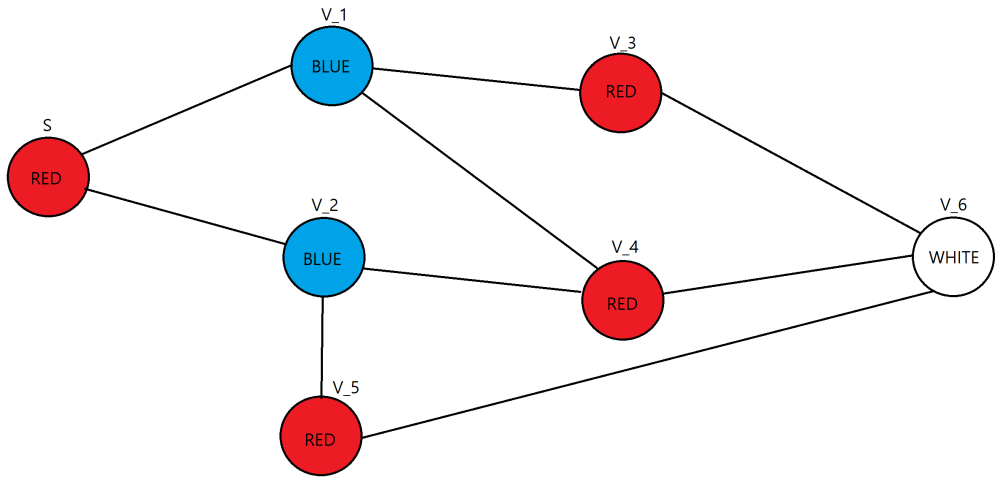
v_5 is colored with WHITE and v_2 is colored with BLUE

→ Color v_5 with RED

→ Enqueue v_5

Q is (v_3, v_4, v_5)

Graph G is as follows.



We are in the while loop at 6-16 because Q is not empty.

Dequeue v_3 of Q.

Adjacent vertices of S are v_1, v_6

v_1 is not colored with WHITE and colors of v_1 and v_3 are not same

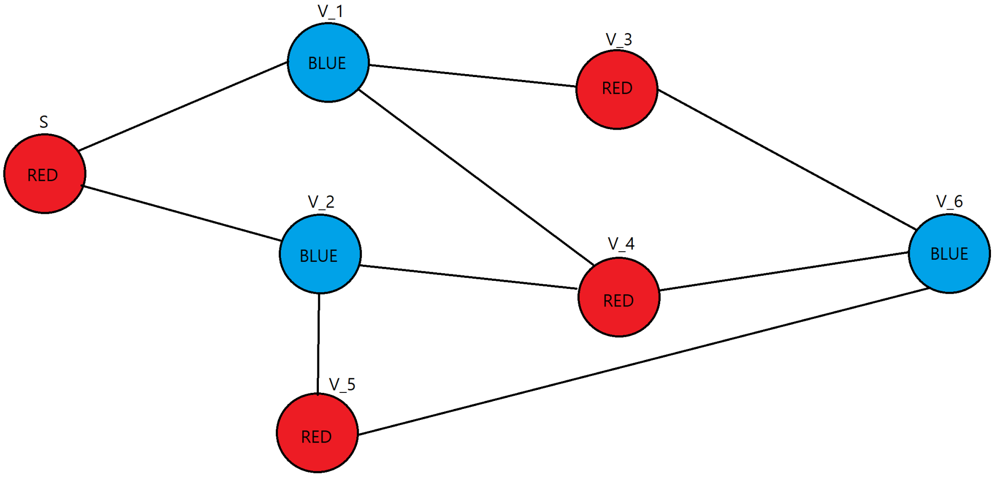
v_6 is colored with WHITE and v_3 is colored with RED

→ Color v_6 with BLUE

→ Enqueue v_6

Q is (v_4, v_5, v_6)

Graph G is as follows.



We are in the while loop at 6-16 because Q is not empty.

Dequeue v_4 of Q.

Adjacent vertices of S are v_1, v_2, v_6

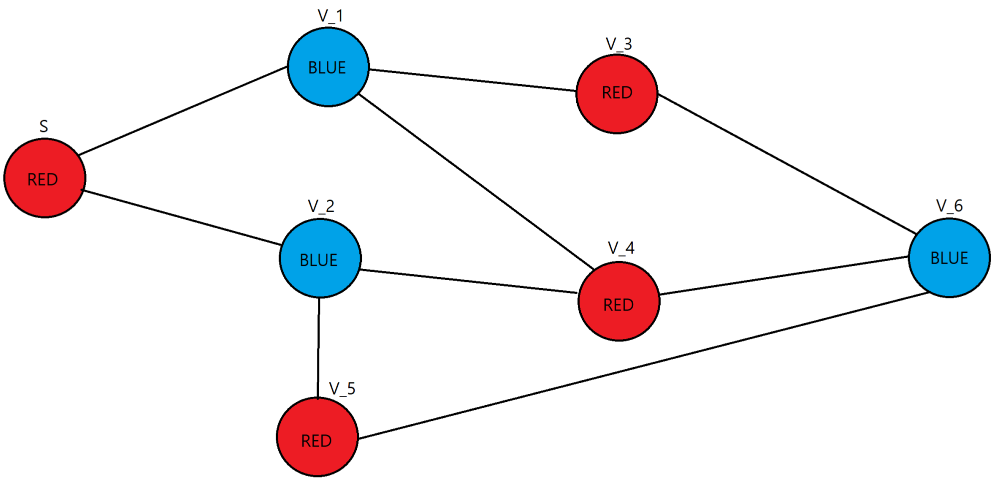
v_1 is not colored with WHITE and colors of v_1 and v_4 are not same

v_2 is not colored with WHITE and colors of v_2 and v_4 are not same

v_6 is not colored with WHITE and colors of v_6 and v_4 are not same

Q is (v_5, v_6)

Graph G is as follows.



We are in the while loop at 6-16 because Q is not empty.

Dequeue v_5 of Q.

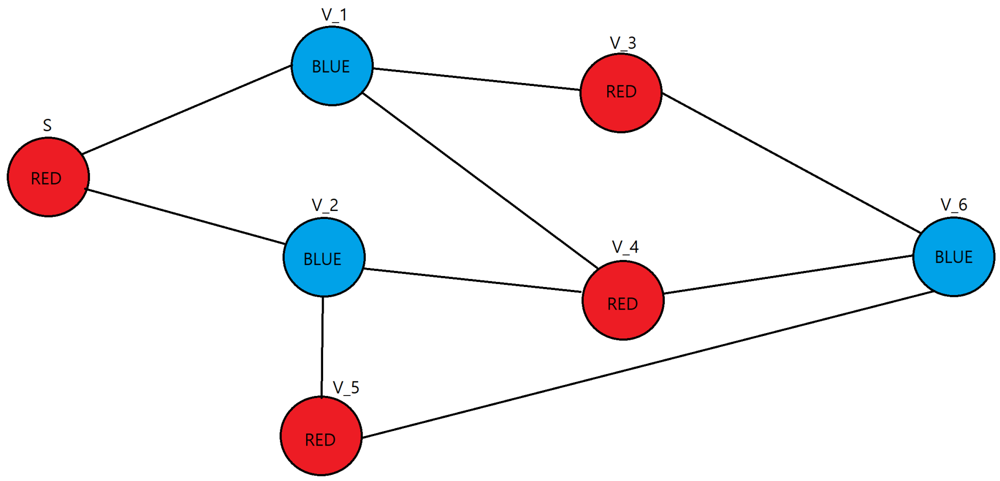
Adjacent vertices of S are v_2, v_6

v_2 is not colored with WHITE and colors of v_2 and v_4 are not same

v_6 is not colored with WHITE and colors of v_6 and v_4 are not same

Q is (v_6)

Graph G is as follows.



We are in the while loop at 6-16 because Q is not empty.

Dequeue v_6 of Q.

Adjacent vertices of S are v_3, v_4, v_5

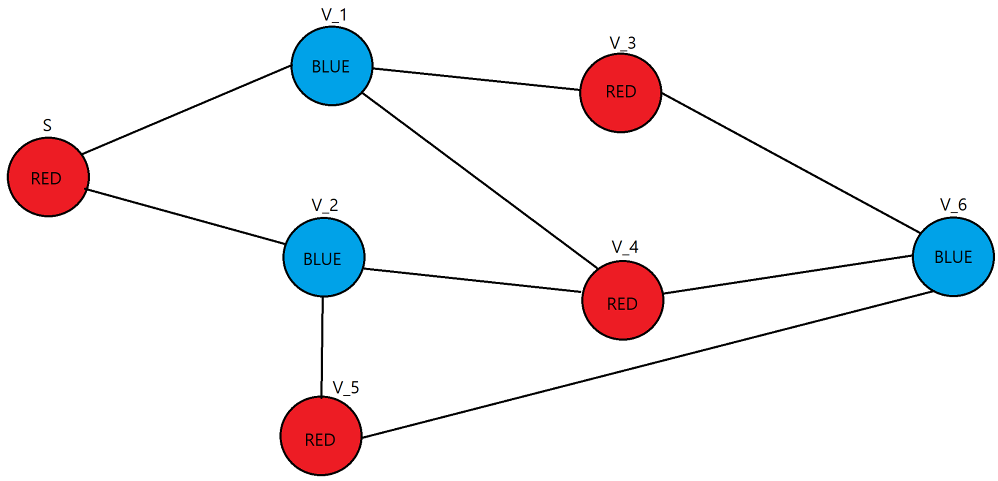
v_3 is not colored with WHITE and colors of v_3 and v_6 are not same

v_4 is not colored with WHITE and colors of v_4 and v_6 are not same

v_5 is not colored with WHITE and colors of v_5 and v_6 are not same

Q is ()

Graph G is as follows.



We are out of while loop at 6-16 because Q is empty.

After running line 17, the algorithm gives us the anwser "YES"

3rd prblem