```java
class BellmanFordAlog
{
    public int isNegativeWeightCycle(int v, int[][] edges)
    {
        //code here

       int inf =  Integer.MAX_VALUE ;
       long [] distance = new long[v];// to keep track of distance from the origin
       int[] path = new int path[];
       Array.fill(path,-1);
       Arrays.fill(distance,inf);
       distance[0]=0; // try checking for every edge from src + weight < directDitanceformOrigin[dst]
       for(int i=0;i<v-1;i++){//you need to repeat above step for V-1 steps to make sure changes propagate to all edges;
           for(int[] edge : edges){
               if(distance[edge[0]]+edge[2]<distance[edge[1]]){
                   distance[edge[1]]=distance[edge[0]] + edge[2];
                   path[edge[1]]=edge[0];  // later to backtrace to find the path
               }
       }
    }

     // if the distance array changes in the below iteration then there is negative weight loop
     for(int i=0;i<v-1;i++){
           for(int[] edge : edges){
               if(distance[edge[0]]+edge[2]<distance[edge[1]]){
                  distance[edge[1]]=Integer.MIN_VALUE;
                  path[edge[1]]=-2; // indicating parth of negetive weight loop.// dont know how to diffrentiate
               }                    // between actual part of loop and affected by negative loops
       }



     return 0;
}

}
```