```java
/**
 * @author Bishoy A Abdelmalik
 * @class COMP 282 meeting at 2:00 PM
 * @Assigment Program #4
 * @DateTurnedIn Dec 9, 2019
 * @description this file includes the code to do the topological sort
 * uses a queue to save the nodes with no predecessors  *
 */

class GraphTopSort extends Graph {

    // set all predecessor counts to 0
    public void initPredCounts() {
        Vertex_Node currentLocation=head;
        while (currentLocation!=null){
            currentLocation.setPredCt(0);
            currentLocation=currentLocation.getNext();
        }


    }
    private void setPredCount(Vertex_Node currentLocation) {
        while (currentLocation!=null) {
            Edge_Node edge=currentLocation.getNbrList();
            while (edge!=null){
                edge.getTarget().setPredCt(edge.getTarget().getPredCt()+1);
                edge=edge.getNext();
            }
            currentLocation=currentLocation.getNext();
        }

    }

    public void outputTopSort() {
        Vertex_Node[] zeroPredecessorQueue = new Vertex_Node[this.size];
        Vertex_Node[] topSort = new Vertex_Node[this.size];

        int queueFront = 0, queueBack = 0, outputCt = 0;
        Vertex_Node v = this.head;
        //initialize all vertices pred count to 0
        initPredCounts();
        // set the predecessor counts by visiting all the edges and
        // incrementing the target predecessor counts of these edges

        // implement as simple nested while loops
        setPredCount(v);

        // find vertices with predecessor counts of 0 and put them on the queue
        // this will be a while loop
        while (v!=null){
            if (v.getPredCt()==0) {
                zeroPredecessorQueue[queueBack++]=v;
            }
            v=v.getNext();
        }

        // the main loop
        // remove a vertex from the zero queue, add it to the topSort array,
        // and traverse its edges to update target predecessor counts -- if any
        // become 0, add to the zero queue
        while (queueBack != queueFront) {  // while the queue is not empty
            Vertex_Node zeroVertex=zeroPredecessorQueue[queueFront++];
            zeroVertex.setPredCt(-1);
            Edge_Node edge=zeroVertex.getNbrList();
            while (edge!=null){
                edge.getTarget().setPredCt(edge.getTarget().getPredCt()-1);
                if (edge.getTarget().getPredCt()==0){
                    zeroPredecessorQueue[queueBack++]=edge.getTarget();
                }
                edge=edge.getNext();
            }
            topSort[outputCt++]=zeroVertex;
        }

        // check if every vertex has been put in the topSort array and either output it
        // on one line or report "loop"
        v=this.head;
        boolean loop=false;
        while (v!=null){
            if (v.getPredCt()!=-1){
                loop=true;
            }
            v=v.getNext();
        }
        if (loop){
            System.out.println("Loop");
        }else {
            String out="";
            for (Vertex_Node vertix:topSort) {
                out+=vertix.getName()+" ";
            }
            System.out.println(out);
        }

        // Your output should match mine exactly


    }



    public static String myName() {
        return "Bishoy Abdelmalik";
    }
}


```