**Program :**

import java.util.\*;

import static SPCC.FirstAndFollow.\*;

public class LL1 {

    public static Character start = 'a';

    public static boolean flag = true;

    public static void main(String[] args) {

        Scanner scn = new Scanner(System.in);

        HashMap<Character, List<String>> map = new HashMap<>();

        System.out.println("Enter no of rules");

        int n = scn.nextInt();

        while (n-- > 0) {

            System.out.println("Enter head");

            Character c = scn.next().charAt(0);

            List<String> prod = new ArrayList<>();

            System.out.println("Enter no of productions");

            int p = scn.nextInt();

            p++;

            while (p-- > 0) {

                prod.add(scn.nextLine());

            }

            map.put(c, prod);

        }

        System.out.println("Enter the start symbol");

        start = scn.next().charAt(0);

        List<Character> left = new ArrayList<>();

        printProd(map, left);

        HashMap<Character, List<Character>> first = new HashMap<>();

        for (int i = 0; i < left.size(); i++) {

            if (!first.containsKey(left.get(i))) {

                printFirst(left.get(i), map, first, left);

            }

        }

        System.out.println("FIRST IS ----\n");

        for (Map.Entry<Character, List<Character>> m : first.entrySet()) {

            System.out.print(m.getKey() + " -->");

            for (int i = 0; i < m.getValue().size(); i++) {

                System.out.print(m.getValue().get(i) + " ");

            }

            System.out.println();

        }

        HashMap<Character, List<Character>> follow = new HashMap<>();

        for (int i = 0; i < left.size(); i++) {

            if (!follow.containsKey(left.get(i))) {

                printFollow(start, left.get(i), map, follow, left, first);

            }

        }

        System.out.println("FOLLOW IS ----\n");

        for (Map.Entry<Character, List<Character>> m : follow.entrySet()) {

            System.out.print(m.getKey() + " -->");

            for (int i = 0; i < m.getValue().size(); i++) {

                System.out.print(m.getValue().get(i) + " ");

            }

            System.out.println();

        }

        System.out.println("Numbers given to grammar ...");

        List<Character> right = new ArrayList<>();

        for (int i = 0; i < left.size(); i++) {

            List<String> curr = map.get(left.get(i));

            for (int j = 0; j < curr.size(); j++) {

                for (int k = 0; k < curr.get(j).length(); k++) {

                    if (!map.containsKey(curr.get(j).charAt(k))) {

                        if (!right.contains(curr.get(j).charAt(k))) {

                            right.add(curr.get(j).charAt(k));

                        }

                    }

                }

            }

        }

        List<Pair> p = new ArrayList<>();

        int val = 1;

        for (Map.Entry<Character, List<String>> m : map.entrySet()) {

            int r = left.indexOf(m.getKey());

            int c = 0;

            for (int i = 1; i < m.getValue().size(); i++) {

                System.out.println(m.getValue().get(i) + " -> " + val);

                if (first.containsKey(m.getValue().get(i).charAt(0))) {

                    List<Character> fl = first.get(m.getValue().get(i).charAt(0));

                    for (int k = 0; k < fl.size(); k++) {

                        // System.out.println(fl.get(k));

                        p.add(new Pair(r, right.indexOf(fl.get(k)), val));

                    }

                } else {

                    c = right.indexOf(m.getValue().get(i).charAt(0));

                    p.add(new Pair(r, c, val));

                    // System.out.println(r+" "+m.getValue().get(i).charAt(0));

                }

                val++;

            }

        }

        String[][] ll1 = new String[left.size()][right.size()];

        for (String[] j : ll1) {

            Arrays.fill(j, "");

        }

        for (Pair pq : p) {

            if (ll1[pq.r][pq.c] != "") {

                flag = false;

            }

            ll1[pq.r][pq.c] += String.valueOf(pq.val);

        }

        System.out.println("THE LL1 PARSER TABLE IS :");

        System.out.print(" ");

        for (int i = 0; i < right.size(); i++) {

            System.out.print(right.get(i) + " ");

        }

        System.out.println();

        for (int i = 0; i < ll1.length; i++) {

            System.out.print(left.get(i) + " ");

            for (int j = 0; j < ll1[0].length; j++) {

                System.out.print(ll1[i][j] + " ");

            }

            System.out.println();

        }

        if (!flag) {

            System.out.println("The grammer is not LL1");

        } else {

            System.out.println("The grammer is LL1");

        }

    }

}

class Pair {

    int r, c, val;

    Pair(int r, int c, int val) {

        this.r = r;

        this.c = c;

        this.val = val;

    }

}

**Output:**