```
package ass2macroPass2;

import java.io.BufferedReader;
import java.io.FileInputStream;
import java.io.FileOutputStream;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.LinkedList;
import java.util.List;
import java.util.Map;
import java.util.StringTokenizer;

public class ass2macroPass2 {

    static List<MntTuple> mnt; //MNT List
    static List<String> mdt; //MDT List
    static int mntc; //Initialized to 1
    static int mdtc; //Initialized to 1
    static int mdtp; //used in Pass 2
    static BufferedReader input; //reading Files
    static List<List<String>> ala; //Prepare Argument List Array
    static Map<String, Integer> ala_macro_binding; //used for binding ALA

    public static void main(String args[]) throws Exception {
        initializeTables(); //Initializing everything

        //mnt touple initializing

        String s;
        BufferedReader br;
        br = new BufferedReader(new InputStreamReader(new FileInputStream("/home/student/workspace/
SPOSL/src/out_mnt.txt")));
        //reading Symbol table
        while ((s = br.readLine()) != null) {
            StringTokenizer st = new StringTokenizer(s, " ", false); //convert line into tokens
            mnt.add(new MntTuple(Integer.parseInt(st.nextToken()), st.nextToken(),Integer.parseInt(st.nextTo
ken()))); //adding token into list
        }

        //mdt initializing
        br = new BufferedReader(new InputStreamReader(new FileInputStream("/home/student/workspace/
SPOSL/src/out_mdt.txt")));
        while ((s = br.readLine()) != null) {
            mdt.add(s);
        }

        mntc = 3;
        mdtc = 9;
        mdtp = 0;

        br = new BufferedReader(new InputStreamReader(new FileInputStream("/home/student/workspace/
SPOSL/src/out_ala_pass1.txt")));
        while ((s = br.readLine()) != null) {
            StringTokenizer st = new StringTokenizer(s, " ", false); //convert line into tokens
            List<String> temp1 = new ArrayList<String>();
            temp1.add(st.nextToken());
            temp1.add(st.nextToken());
            ala.add(temp1);
        }

        ala_macro_binding.put("INCR1",0);
        ala_macro_binding.put("INCR2",1);
        System.out.println("\n===== PASS 2 =====\n");
        pass2();
    }

    static void initializeTables() {
        mnt = new LinkedList<>();
        mdt = new ArrayList<>();
        ala = new LinkedList<>();
        mntc = 1;
        mdtc = 1;
        ala_macro_binding = new HashMap<>();
    }

    static void pass2() throws Exception {
        input = new BufferedReader(new InputStreamReader(new FileInputStream("/home/student/workspa
ce/SPOSL/src/output_pass1.txt")));
        //pass 1 as INPUT
        PrintWriter output = new PrintWriter(new FileOutputStream("/home/student/workspace/SPOSL/src/o
utput_pass2.txt"), true);
        //used as MACRO Output expansion
        String token = new String();
        String s;
        while ((s = input.readLine()) != null) { //while reading all lines
            StringTokenizer st = new StringTokenizer(s, " ", false); //convert line into tokens
            while (st.hasMoreTokens()) { //till all tokens are reached
                token = st.nextToken();
                if (st.countTokens() > 2) {
                    token = st.nextToken();
                }
                MntTuple x = null;
                for (MntTuple m : mnt) {
                    if (m.name.equalsIgnoreCase(token)) { //check the MACRO call
                        x = m; //take MACRO_NAME into x
                        break;
                    }
                }
                if (x != null) {
                    mdtp = x.index; //update MDT Index in MDTP
                    List<String> l = pass2Ala(s); //call to Pass 2 ALA and storing them in l (SET UP ALA2)
                    mdtp++; //update MDTP
                    String temp = new String();
                    while (!(temp = mdt.get(mdtp)).trim().equalsIgnoreCase("MEND")) { //reach until MEND recei
ves in code
                        String line = new String();
                        StringTokenizer st2 = new StringTokenizer(temp, " ,", false); //divide line into tokens
                        for (int i = 0; i < 12; i++) { //check argument length
                            line += " ";
                        }
                        String opcode = st2.nextToken();
                        line += opcode;
                        for (int i = opcode.length(); i < 24; i++) { //get actual macro expansion over the call
                            line += " ";
                        }
                        line += st2.nextToken(); //append the macro expansion
                        while (st2.hasMoreTokens()) { //check further tokens and arguments
                            String token2 = st2.nextToken();
                            int index;
                            if ((index = token2.indexOf("#")) != -1) { //if MDT gets ’#’
                                line += "," + l.get(Integer.parseInt(token2.substring(index + 1, index + 2))); //append a
ctual argument
                            }
                        }
                        mdtp++; //now update the pointer
                        output.println(line); //write to file
                        System.out.println(line); //print everything
                    }
                    break;
                } else {
                    output.println(s);
                    System.out.println(s);
                    break;
                }
            }
        }
        System.out.println("\nALA:");
        showAla(2); //print ALA of pass 2 Over here
    }

    static List<String> pass2Ala(String s) {
        StringTokenizer st = new StringTokenizer(s, " ", false); //convert line into tokens
        int num_tokens = st.countTokens(); //count of tokens/arguments
        String macro_name = st.nextToken(); //save macro name of these arguments
        int ala_no = ala_macro_binding.get(macro_name); //get complete key value macro binding
        List<String> l = ala.get(ala_no); //take complete ala binding in l
        int ctr = 0;
        StringTokenizer st2 = null;
        try {
            st2 = new StringTokenizer(st.nextToken(), ",", false);
            while (st2.hasMoreTokens()) {
                l.set(ctr, st2.nextToken()); //set all the tokens to l
                ctr++;
            }
        } catch (Exception e) {
            // do nothing
        }
        if (ctr < num_tokens) {
            String s2 = mdt.get(mdtp); //get complete line from MDT and store it in s2
            StringTokenizer st3 = new StringTokenizer(s2, " ,", false);
            String token = new String();
            int index = 0;
            while (st3.hasMoreTokens()) {
                token = st3.nextToken();
                if ((index = token.indexOf("=")) != -1) {
                    try {
                        l.set(ctr++, token.substring(index + 1, token.length())); //Again, forget after '=' part
                    } catch (Exception e) {
                        // do nothing
                    }
                }
            }
        }
        ala.set(ala_no, l); //substitute all the actual arguments over here (in Pass 2 ALA)
        return l;
    }

    static void showAla(int pass) throws Exception {
        PrintWriter out = new PrintWriter(new FileOutputStream("/home/student/workspace/SPOSL/src/out_
ala_pass" + pass + ".txt"), true); //write in this file
        for (List l : ala) { //till all Arguments reached
            System.out.println(l); //print
            out.println(l); //write to file
        }
    }
}
```