

```

1 import java.util.*;
2 import java.io.*;
3
4 class MntTuple {
5     String name;
6     int index;
7
8     MntTuple(String s, int i) {
9         name = s;
10        index = i;
11    }
12
13    public String toString() {
14        return "[" + name + ", " + index + "]";
15    }
16 }
17
18 class MacroProcessor{
19     static List<MntTuple> mnt;
20     static List<String> mdt;
21     static int mntc;
22     static int mdtc;
23     static int mdtp;
24     static BufferedReader input;
25     static List<List<String>> ala;
26     static Map<String, Integer> ala_macro_binding;
27
28     public static void main(String args[]) throws Exception {
29         initializeTables();
30         System.out.println("==== PASS 1 =====\n");
31         pass1();
32     }
33
34     static void pass1() throws Exception {
35         String s = new String();
36         input = new BufferedReader(new InputStreamReader(new
FileInputStream("input.txt")));
37         PrintWriter output = new PrintWriter(new
FileOutputStream("output_pass1.txt"), true);
38         while((s = input.readLine()) != null) {
39             if(s.equalsIgnoreCase("MACRO")) {
40                 processMacroDefinition();
41             } else {
42                 output.println(s);
43             }
44         }
45         System.out.println("ALA:");
46         showAla(1);
47         System.out.println("\nMNT:");
48         showMnt();
49         System.out.println("\nMDT:");
50         showMdt();
51     }
52
53     static void processMacroDefinition() throws Exception {
54         String s = input.readLine();
55         String macro_name = s.substring(0, s.indexOf(" "));
56         mnt.add(new MntTuple(macro_name, mdtp));
57         mdtp++;
58         pass1Ala(s);

```

```

59 StringTokenizer st = new StringTokenizer(s, " ,", false);
60 String x = st.nextToken();
61 for(int i=x.length() ; i<12 ; i++) {
62     x += " ";
63 }
64 String token = new String();
65 int index;
66 token = st.nextToken();
67 x += token;
68 while(st.hasMoreTokens()) {
69     token = st.nextToken();
70     x += "," + token;
71 }
72 mdt.add(x);
73 mdtc++;
74 addIntoMdt(ala.size()-1);
75 }
76
77 static void pass1Ala(String s) {
78     StringTokenizer st = new StringTokenizer(s, " ,", false);
79     String macro_name = st.nextToken();
80     List<String> l = new ArrayList<>();
81     int index;
82     while(st.hasMoreTokens()) {
83         String x = st.nextToken();
84         if((index = x.indexOf("=")) != -1) {
85             x = x.substring(0, index);
86         }
87         l.add(x);
88     }
89     ala.add(l);
90     ala_macro_binding.put(macro_name, ala_macro_binding.size());
91 }
92
93 static void addIntoMdt(int ala_number) throws Exception {
94     String temp = new String();
95     String s = new String();
96     List l = ala.get(ala_number);
97     boolean isFirst;
98     while(!s.equalsIgnoreCase("MEND")) {
99         isFirst = true;
100         s = input.readLine();
101         String line = new String();
102         StringTokenizer st = new StringTokenizer(s, " ,", false);
103         temp = st.nextToken();
104         for(int i=temp.length() ; i<12 ; i++) {
105             temp += " ";
106         }
107         line += temp;
108         while(st.hasMoreTokens()) {
109             temp = st.nextToken();
110             if(temp.startsWith("&")) {
111                 int x = l.indexOf(temp);
112                 temp = ",#" + x;
113                 isFirst = false;
114             } else if(!isFirst) {
115                 temp = "," + temp;
116             }
117             line += temp;
118         }

```

```
119     mdt.add(line);
120     mdtc++;
121 }
122 }
123
124 static void showAla(int pass) throws Exception {
125     PrintWriter out = new PrintWriter(new FileOutputStream("out_ala_pass" +
126 pass + ".txt"), true);
127     for(List l : ala) {
128         System.out.println(l);
129         out.println(l);
130     }
131 }
132
133 static void showMnt() throws Exception {
134     PrintWriter out = new PrintWriter(new FileOutputStream("out_mnt.txt"),
135 true);
136     for(MntTuple l : mnt) {
137         System.out.println(l);
138         out.println(l);
139     }
140 }
141
142 static void showMdt() throws Exception {
143     PrintWriter out = new PrintWriter(new FileOutputStream("out_mdt.txt"),
144 true);
145     for(String l : mdt) {
146         System.out.println(l);
147         out.println(l);
148     }
149 }
150
151 static void initializeTables() {
152     mnt = new LinkedList<>();
153     mdt = new ArrayList<>();
154     ala = new LinkedList<>();
155     mntc = 0;
156     mdtc = 0;
157     ala_macro_binding = new HashMap<>();
158 }
159 }
```