```
import java.util.*;
import java.io.*;

class Macro {
    String name;
    List<String> parameters;
    List<String> definition;

    public Macro(String name, List<String> parameters, List<String> definition) {
        this.name = name;
        this.parameters = parameters;
        this.definition = definition;
    }
}

public class TwoPassMacroProcessor {
    public static void main(String[] args) throws IOException {
        List<String> inputCode = new ArrayList<>();
        List<Macro> macros = new ArrayList<>();
        Map<String, Integer> MNT = new HashMap<>();
        Map<String, Integer> ALA = new HashMap<>();
        List<String> MDT = new ArrayList<>();
        List<String> intermediateCode = new ArrayList<>();

        // Read input code from a file or source
        try (BufferedReader reader = new BufferedReader(new FileReader("C:\\Users\\DELL\\Documents\\in
put.txt"))) {
            String line;
            while ((line = reader.readLine()) != null) {
                inputCode.add(line);
            }
        }

        int currentMacroIndex = -1;

        for (int i = 0; i < inputCode.size(); i++) {
            String line = inputCode.get(i);
            String[] tokens = line.split("\\s+");

            if (tokens[0].equals("MACRO")) {
                String macroName = tokens[1];
                List<String> parameters = Arrays.asList(tokens).subList(2, tokens.length - 1);
                Macro macro = new Macro(macroName, parameters, new ArrayList<>());
                macros.add(macro);
                currentMacroIndex = macros.indexOf(macro);
                MNT.put(macroName, currentMacroIndex);

                // Skip macro definition lines
                while (!inputCode.get(i).equals("MEND")) {
                    i++;
                }
            } else {
                if (currentMacroIndex != -1) {
                    MDT.add(line);
                    // Replace formal parameters with positional parameters
                    String[] macroTokens = line.split("\\s+");
                    for (int j = 1; j < macroTokens.length; j++) {
                        if (macros.get(currentMacroIndex).parameters.contains(macroTokens[j])) {
                            ALA.put(macroTokens[j], j);
                        }
                    }

                    // Generate intermediate code
                    StringBuilder intermediateLine = new StringBuilder(macroTokens[0]);
                    for (int j = 1; j < macroTokens.length; j++) {
                        if (ALA.containsKey(macroTokens[j])) {
                            intermediateLine.append(" #").append(ALA.get(macroTokens[j]));
                        } else {
                            intermediateLine.append(" ").append(macroTokens[j]);
                        }
                    }
                    intermediateCode.add(intermediateLine.toString());
                } else {
                    intermediateCode.add(line);
                }
            }
        }

        // Print MNT, ALA, MDT, and intermediate code
        System.out.println("Macro Name Table (MNT):");
        MNT.forEach((name, index) -> System.out.println(name + ": " + index));

        System.out.println("\nArgument List Array (ALA):");
        ALA.forEach((parameter, position) -> System.out.println(parameter + ": " + position));

        System.out.println("\nMacro Definition Table (MDT):");
        for (int i = 0; i < MDT.size(); i++) {
            System.out.println(i + ": " + MDT.get(i));
        }

        System.out.println("\nIntermediate Code:");
        for (String code : intermediateCode) {
            System.out.println(code);
        }
    }
}
```

Output:

```
Macro Name Table (MNT):
ADD: 0

Argument List Array (ALA):

Macro Definition Table (MDT):
0:
1: START
2: ADD X, Y
3: SUB Z, W
```

```
4: END
```

Intermediate Code:

```
START
ADD X, Y
SUB Z, W
END
```