Text

Description automatically generated

**T.Y.B.Tech (CSE)**

[System Software and Compilers(SSC)](https://mitwpu.instructure.com/courses/3210)

**Lab Assignment No – 4**

**Name: Aniruddha Shende**

**Roll number: PD-05**

**Batch: D1**

**Panel: D**

Text, letter

Description automatically generated

A picture containing text, document, receipt, screenshot

Description automatically generated

Diagram

Description automatically generated

A piece of paper with writing

Description automatically generated with medium confidence

**Code and Output on Next page:**

**INPUT file  without macro definition (from Pass 1 of macro processor):**

Text

Description automatically generated

**Java Code:**

1. **Main.java file**

package com.lab1;

public class Main {

public static void main(String[] args) {

Pass1\_MACRO.printPass1\_MACRO();

Pass2\_MACRO.printPass2\_MACRO();

}

}

1. **MDT Table file**

package com.lab1;

import java.util.LinkedHashMap;

public class MDTtable {

private static int location\_counter = 0;

private static LinkedHashMap<String, String> MDT = new LinkedHashMap<String, String>();

public static int getLocation\_counter() {

return location\_counter;

}

public static void add(String instructions) {

location\_counter += 1;

MDT.put(Integer.toString(location\_counter), instructions);

}

public static void printMDT() {

for (String key : MDT.keySet()) {

System.out.println(key + " " + MDT.get(key));

}

}

}

1. **MNT Table file**

package com.lab1;

import java.util.HashSet;

import java.util.LinkedHashMap;

import java.util.Set;

public class MNTtable {

private static LinkedHashMap<String, String> MNT = new LinkedHashMap<String, String>();

private static Set<String> all\_macros = new HashSet<String>();

public static void add\_to\_MNT(String macro\_name, int index) {

all\_macros.add(macro\_name);

MNT.put(macro\_name,Integer.toString(index));

}

public static void printMNT() {

int mnt\_index = 0;

for (String key : MNT.keySet()) {

mnt\_index++;

System.out.println(mnt\_index + " " + key + " " + MNT.get(key));

}

}

public static boolean isMacro\_present(String macro\_name) {

return all\_macros.contains(macro\_name);

}

}

1. **ALA Table file**

package com.lab1;

import java.util.LinkedHashMap;

public class ALAtable {

private static LinkedHashMap<String, String> ALA = new LinkedHashMap<String, String>();

private static int index = 0;

public static void add(String arguments) {

index++;

ALA.put(Integer.toString(index), arguments);

}

public static void printALA() {

for (String key : ALA.keySet()) {

System.out.println(key + " " + ALA.get(key));

}

}

public static LinkedHashMap<String, String> getALA() {

return ALA;

}

}

**5. Pass2\_MACRO.java file**

package com.lab1;

import java.io.BufferedWriter;

import java.io.File;

import java.io.FileNotFoundException;

import java.io.FileWriter;

import java.io.IOException;

import java.util.ArrayList;

import java.util.LinkedHashMap;

import java.util.Scanner;

import java.util.StringTokenizer;

public class Pass2\_MACRO {

public static void printPass2\_MACRO() {

System.out.println("Pass2 MACRO :\n");

File file = new File("/Users/ani/Desktop/Tri-9/SSC/Lab/Lab Assign 2/Assign 2 Lab/src/com/lab1/output\_file.txt");

Scanner sc = null;

try {

sc = new Scanner(file);

} catch (FileNotFoundException e) {

e.printStackTrace();

}

FileWriter xyz;

try

{

xyz = new FileWriter("/Users/ani/Desktop/Tri-9/SSC/Lab/Lab Assign 2/Assign 2 Lab/src/com/lab1/new\_output\_pass2.txt");

BufferedWriter writer2 = new BufferedWriter(xyz);

while (sc.hasNextLine()) {

String line = sc.nextLine();

StringTokenizer st = new StringTokenizer(line, " ");

String opcode = st.nextToken();

if (MNTtable.isMacro\_present(opcode)) {

String str = line;

String[] list1 = str.split(" ");

//fetching the arguemnts of the macro

ArrayList<String> arguments\_of\_macro = Pass1\_MACRO.getMacro\_with\_their\_params(list1[0]);

int j = 0;

LinkedHashMap<String, String> temp\_map = new LinkedHashMap<String, String>();

for (int i = 1; i < list1.length; i++) {

temp\_map.put(arguments\_of\_macro.get(j), list1[i]);

j++;

}

; //replacing the arguments with the values

int j1 = 0;

for (int i = 1; i < list1.length; i++) {

for (String key : ALAtable.getALA().keySet()) {

if (ALAtable.getALA().get(key).equals(arguments\_of\_macro.get(j1))) {

ALAtable.getALA().put(key, list1[i]);

j1++;

break;

}

}

}

//reading the code for extracting the macro

File file1 = new File("/Users/ani/Desktop/Tri-9/SSC/Lab/Lab Assign 2/Assign 2 Lab/src/com/lab1/input1.txt");

Scanner sc1 = null;

try {

sc1 = new Scanner(file1);

} catch (FileNotFoundException e) {

e.printStackTrace();

}

while (sc1.hasNextLine()) {

String line1 = sc1.nextLine();

StringTokenizer st1 = new StringTokenizer(line1, " ");

String opcode1 = st1.nextToken();

if (opcode1.equals("MACRO")) {

String str1 = sc1.nextLine();

String[] list4 = str1.split(" ");

if (MNTtable.isMacro\_present(opcode) && list4[0].equals(opcode)) {

str1 = sc1.nextLine();

while (!str1.equals("MEND")) {

String[] list2 = str1.split(" ");

for (int i = 0; i < list2.length; i++) {

if (temp\_map.containsKey(list2[i])) {

list2[i] = temp\_map.get(list2[i]);

}

}

String fina = "+";

for (int i = 0; i < list2.length; i++) {

fina = fina + list2[i] + " ";

}

//System.out.println(fina);

writer2.write(fina);

writer2.newLine();

str1 = sc1.nextLine();

}

}

}

}

// sc.nextLine();

} else {

writer2.write(line);

writer2.newLine();

}

}

writer2.close();

}

catch (IOException except)

{

except.printStackTrace();

}

System.out.println("\n\nUpdated ALA Table : \n");

ALAtable.printALA();

sc.close();

}

}

**Output:**

**Graphical user interface, text

Description automatically generated with medium confidence**

**Expanded Macro:**

**Text

Description automatically generated**