T.Y.B.Tech (CSE)

System Software and Compilers(SSC)

Lab Assignment No – 4

Name: Aniruddha Shende

Roll number: PD-05

Batch: D1

Panel: D

Name:- Aniruddha Arun Shende
Roll no:- PD-05
Batch:- D1
Panel:- D

MIT-WPU MIT WORLD PEACE UNIVERSITY

Subject:- SSC

Lab Assignment No. - 4

Assignment Title: Design of Pass II of 2 pass Macroprocessor.

Aim :- Design suitable data structure & implement pass II of 2 Pass Macroprocessor

Objective: Design suitable data structure & implement pass II of 2 pass Macroprocessor. Input should consist of a one macro definition & one macro call & few assembly language instructions.

Theory:- Write about

1) Algorithm for Pass II

Ans 1) (a) Read one line of i/p program at a time.
(b) For each line it checks if operate of that line matches any of MNT entry.
(c) The initial value of MDTP is obtained from MDT index field of MNT entry.
(d) The macro expander prepares the ALA consisting of a table of dummy arguments

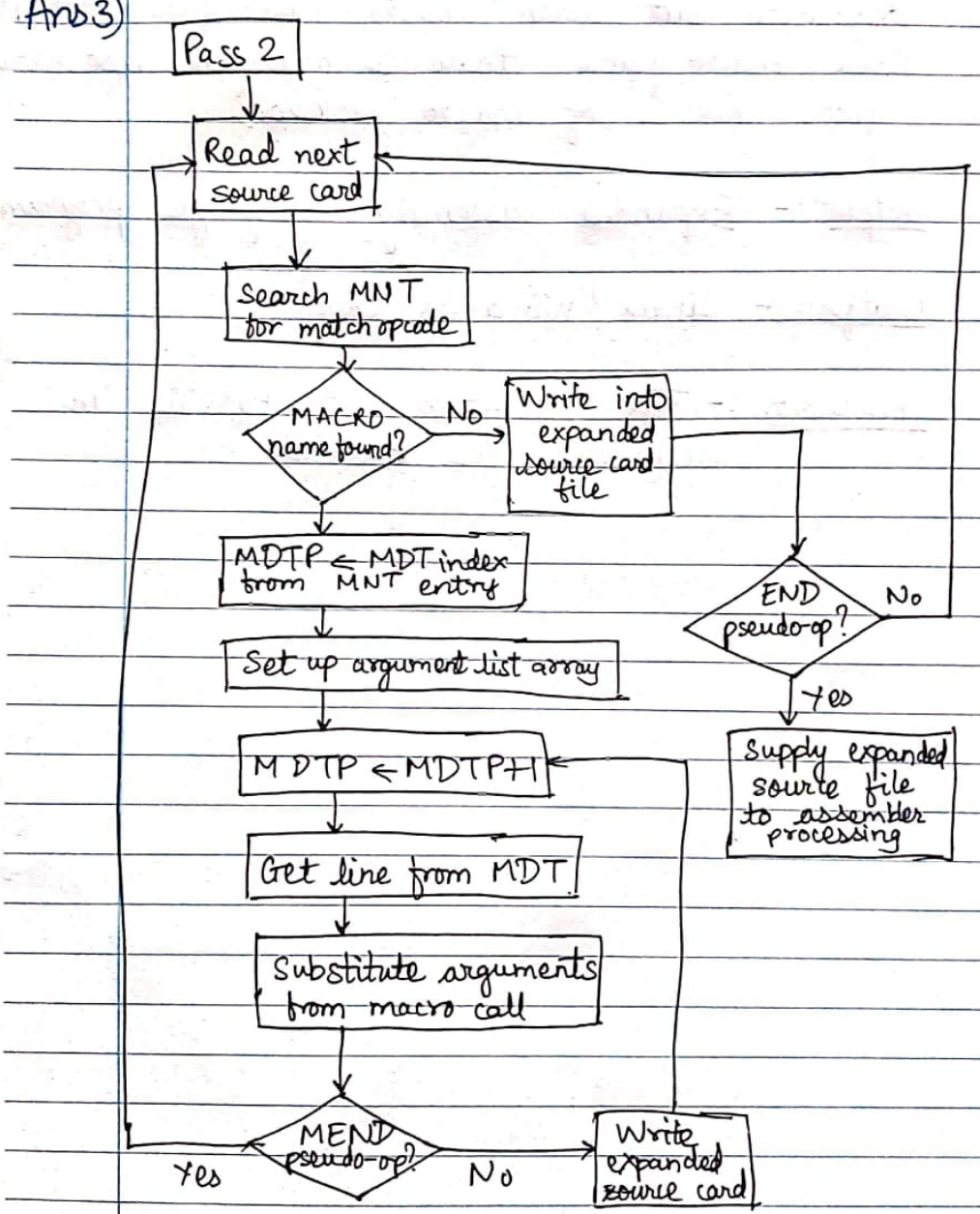
indices & corresponding argument to call.
(e) The value from the argument list is substituted for dummy arguments indices in the macro definitions table.
(f) Reading MEND line in MDT terminates expansion of macro & scanning continues in the i/p files.
(g) When END pseudo-op is encountered, the expanded source program is given to the assembler.

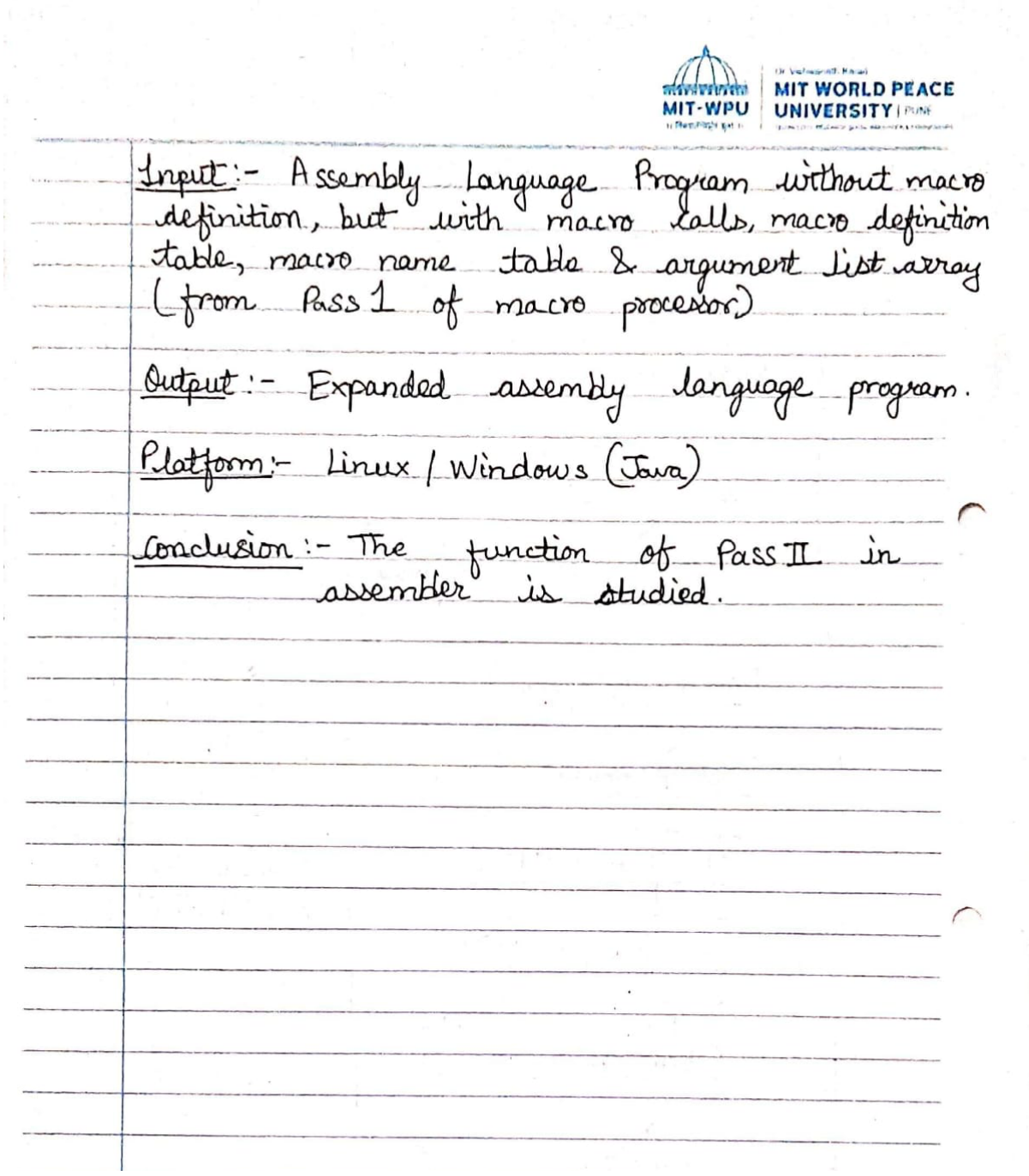
2) Data structures required for 2 pass macroprocessor.

Ans 2) File: Contains output file given from pass I.
MNT: It is used for recognizing macro name
MDT: It is used to perform macro expansion.
MDTP: It is used to point the index of MDT.
ALA: It is used to replace the index notation by the actual value.

3) Flowchart for pass-II

Ans 3)





Input :- Assembly Language Program without macro definition, but with macro calls, macro definition table, macro name table & argument list array (from Pass 1 of macro processor)

Output :- Expanded assembly language program.

Platform :- Linux / Windows (Java)

Conclusion :- The function of Pass II in assembler is studied.

Code and Output on Next page:

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

```
1   START
2   MOVER AREG S1
3   MOVER BREG S1
4   INCR D1 D2
5   DECR D3 D4
6   S1 DC 5
7   D1 DC 2
8   D2 DC 3
9   D3 DC 4
10  D4 DC 5
11  END
```

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();
    }
}
```

2. 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));
        }
    }
}
```

3. 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);
    }
}
```

4. 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:

```
Pass2 MACRO :


Updated ALA Table :

Index   Formal Arguments
-----------------------------------------
1       D1
2       D2
3       D3
4       D4

ani@Aniruddhas-MacBook-Pro Lab Assign 2 %
```

Expanded Macro:

Assign 2 Lab > src > com > lab1 > new_output_pass2.txt

```
START
MOVER AREG S1
MOVER BREG S1
+ADD AREG D1
+ADD BREG D2
+SUB AREG D3
+SUB BREG D4
S1 DC 5
D1 DC 2
D2 DC 3
D3 DC 4
D4 DC 5
END
```