EXPERIMENT NO : 2

**Problem Statement:** Design suitable data structures and implement Pass-I and Pass- II of a two-pass macro-processor. The output of Pass-I (MNT, MDT and intermediate code file without any macro definitions) should be input for Pass-II.

# Macro Processor Pass I:

**Source code**

package com.spos;

import java.io.BufferedReader; import java.io.FileReader; import java.io.FileWriter; import java.io.IOException; import java.util.Iterator;

import java.util.LinkedHashMap; public class LabTwo {

public static void main(String[] args) throws IOException { BufferedReader br= new BufferedReader(new

FileReader("C:\\Eclipse\\SposPracs\\src\\com\\spos\\input.asm")); FileWriter mnt= new FileWriter("mnt.txt"); FileWriter mdt= new FileWriter("mdt.txt"); FileWriter kpdt= new FileWriter("kpdt.txt"); FileWriter pnt= new FileWriter("pntab.txt");

FileWriter ir= new FileWriter("intermediate.txt"); LinkedHashMap<String, Integer> pntab=new LinkedHashMap<>(); String line;

String Macroname = null;

int mdtp=1,kpdtp=0,paramNo=1,pp=0,kp=0,flag=0;

while((line=br.readLine())!=null)

{

String parts[]=line.split("\\s+"); if(parts[0].equalsIgnoreCase("MACRO"))

{

flag=1; line=br.readLine(); parts=line.split("\\s+"); Macroname=parts[0]; if(parts.length<=1)

{

mnt.write(parts[0]+"\t"+pp+"\t"+kp+"\t"+mdtp+"\t"+(kp==0?kpdtp:(kpdtp+1))+"\n"); System.***out***.println(parts[0]+"\t"+pp+"\t"+kp+"\t"+mdtp+"\t"+(kp==0?kpdtp:(kpdtp+1))

+"\n");

continue;

}

for(int i=1;i<parts.length;i++) //processing of parameters

{

parts[i]=parts[i].replaceAll("[&,]", "");

//System.out.println(parts[i]); if(parts[i].contains("="))

{

++kp;

String keywordParam[]=parts[i].split("="); pntab.put(keywordParam[0], paramNo++); if(keywordParam.length==2)

{

kpdt.write(keywordParam[0]+"\t"+keywordParam[1]+"\n"); System.***out***.println(keywordParam[0]+"\t"+keywordParam[1]+"\n");

}

else

{

kpdt.write(keywordParam[0]+"\t-\n");

System.***out***.println(keywordParam[0]+"\t-\n");

}

}

else

{

pntab.put(parts[i], paramNo++); pp++;

}

}

mnt.write(parts[0]+"\t"+pp+"\t"+kp+"\t"+mdtp+"\t"+(kp==0?kpdtp:(kpdtp+1))+"\n");

System.***out***.println(parts[0]+"\t"+pp+"\t"+kp+"\t"+mdtp+"\t"+(kp==0?kpdtp:(kpdtp+1))

+"\n");

kpdtp=kpdtp+kp;

//System.out.println("KP="+kp);

}

else if(parts[0].equalsIgnoreCase("MEND"))

{

mdt.write(line+"\n"); System.***out***.println(line+"\n"); flag=kp=pp=0;

mdtp++; paramNo=1;

pnt.write(Macroname+":\t"); Iterator<String> itr=pntab.keySet().iterator(); while(itr.hasNext())

{

pnt.write(itr.next()+"\t"); System.***out***.println(itr.next()+"\t");

}

pnt.write("\n"); System.***out***.println("\n"); pntab.clear();

}

else if(flag==1)

{

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

{

if(parts[i].contains("&"))

{

parts[i]=parts[i].replaceAll("[&,]", "");

}

else

{

mdt.write("(P,"+pntab.get(parts[i])+")\t");

System.***out***.println("(P,"+pntab.get(parts[i])+")\t");

mdt.write(parts[i]+"\t"); System.***out***.println(parts[i]+"\t");

}

}

}

else

{

}

}

mdt.write("\n"); System.***out***.println("\n"); mdtp++;

ir.write(line+"\n"); System.***out***.println(line+"\n");

br.close();

mdt.close();

mnt.close();

ir.close();

pnt.close();

kpdt.close();

System.***out***.println("Macro Pass1 Processing done. :)");

}

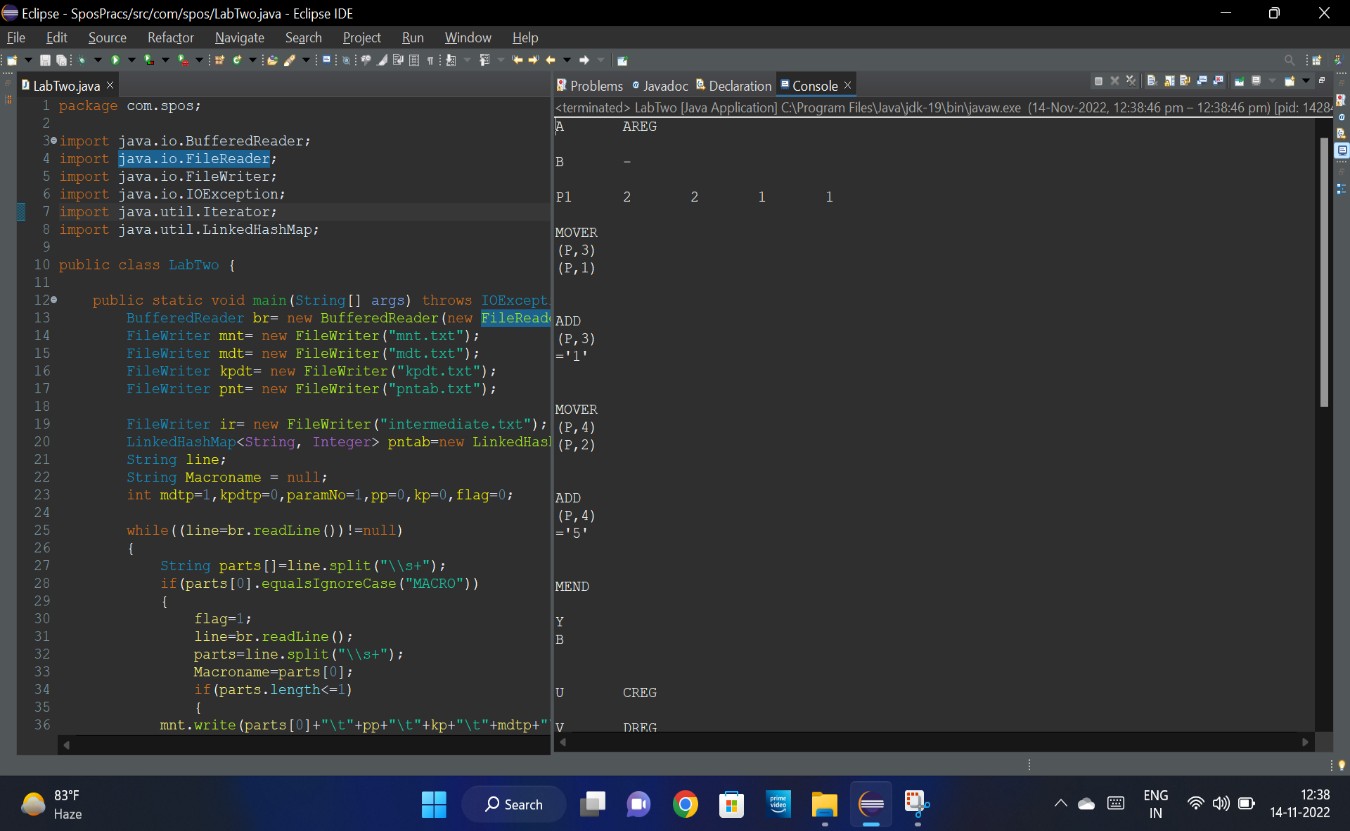
}

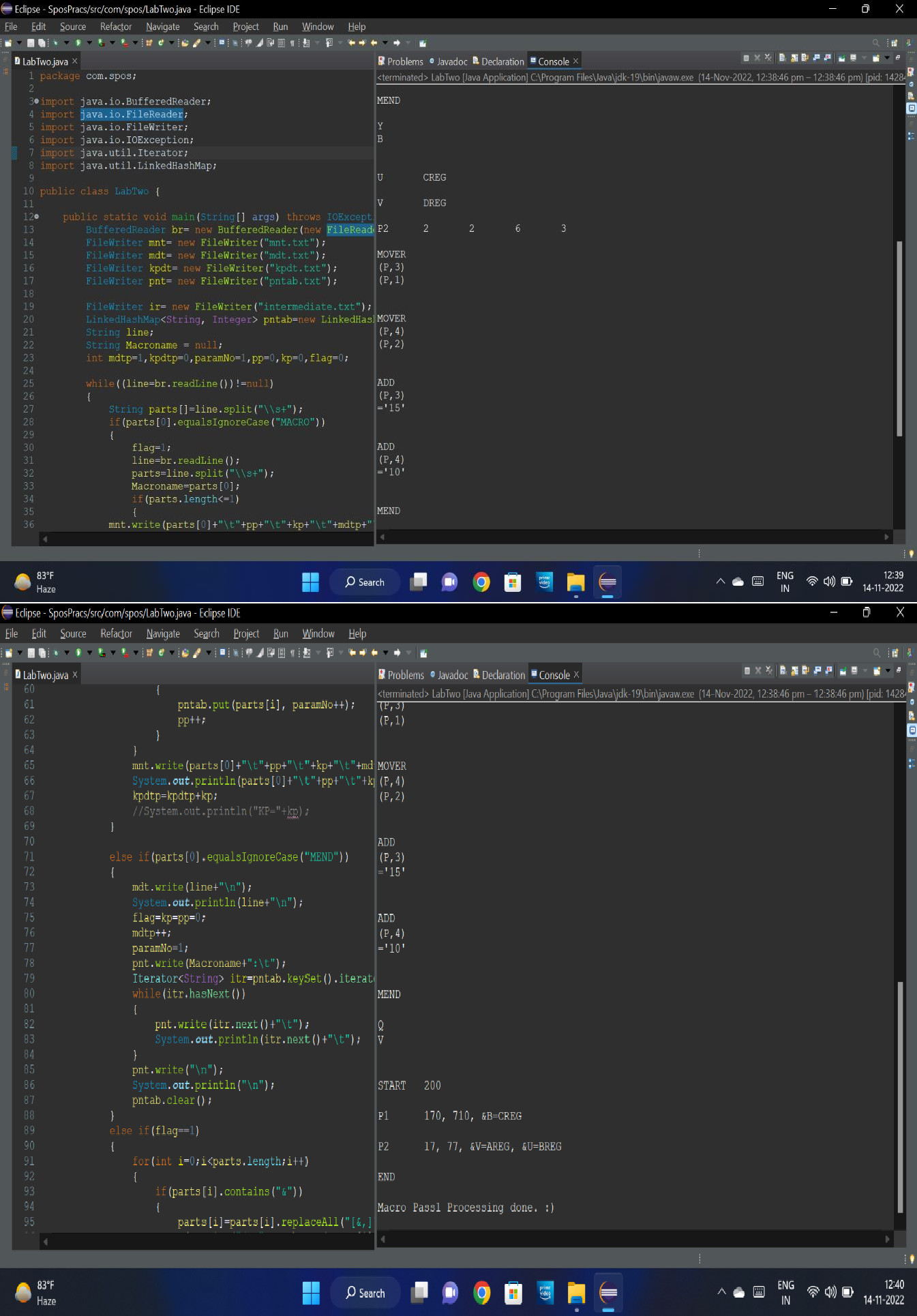
# Input Assembly file:

|  |  |  |
| --- | --- | --- |
| MACRO |  | |
| M1 | &X, | &Y, &A=AREG, &B= |
| MOVER | &A, | &X |
| ADD | &A, | ='1' |
| MOVER | &B, | &Y |
| ADD | &B, | ='5' |
| MEND |  |  |
| MACRO |  |  |
| M2 | &P, | &Q, &U=CREG, &V=DREG |
| MOVER | &U, | &P |
| MOVER | &V, | &Q |
| ADD | &U, | ='15' |
| ADD | &V, | ='10' |
| MEND |  |  |
| START | 100 |  |
| M1 | 10, | 20, &B=CREG |

M2 100, 200, &V=AREG, &U=BREG END

**Output:**





1. **Macro Processor Pass I:**

# Source code

package com.spos;

import java.io.BufferedReader; import java.io.FileReader; import java.io.FileWriter; import java.io.IOException; import java.util.HashMap; import java.util.Vector;

public class LabTwoMacroPassTwo {

public static void main(String[] args) throws IOException {

// **TODO** Auto-generated method stub

BufferedReader irb=new BufferedReader(new FileReader("intermediate.txt")); BufferedReader mdtb=new BufferedReader(new FileReader("mdt.txt")); BufferedReader kpdtb=new BufferedReader(new FileReader("kpdt.txt")); BufferedReader mntb=new BufferedReader(new FileReader("mnt.txt")); FileWriter fr=new FileWriter("pass2.txt");

HashMap<String, MNTEntry> mnt=new HashMap<>(); HashMap<Integer, String> aptab=new HashMap<>(); HashMap<String,Integer> aptabInverse=new HashMap<>(); Vector<String>mdt=new Vector<String>(); Vector<String>kpdt=new Vector<String>();

int pp,kp,mdtp,kpdtp,paramNo; String line; while((line=mdtb.readLine())!=null)

{

mdt.addElement(line);

}

while((line=kpdtb.readLine())!=null)

{

kpdt.addElement(line);

}

while((line=mntb.readLine())!=null)

{

String parts[]=line.split("\\s+");

mnt.put(parts[0], new MNTEntry(parts[0], Integer.*parseInt*(parts[1]), Integer.*parseInt*(parts[2]), Integer.*parseInt*(parts[3]), Integer.*parseInt*(parts[4])));

//System.out.println();

}

while((line=irb.readLine())!=null)

{

String []parts=line.split("\\s+"); if(mnt.containsKey(parts[0]))

{

pp=mnt.get(parts[0]).getPp();

kp=mnt.get(parts[0]).getKp(); kpdtp=mnt.get(parts[0]).getKpdtp(); mdtp=mnt.get(parts[0]).getMdtp(); paramNo=1;

for(int i=0;i<pp;i++)

{

parts[paramNo]=parts[paramNo].replace(",", ""); aptab.put(paramNo, parts[paramNo]); aptabInverse.put(parts[paramNo], paramNo); paramNo++;

}

int j=kpdtp-1; for(int i=0;i<kp;i++)

{

String temp[]=kpdt.get(j).split("\t"); aptab.put(paramNo,temp[1]); aptabInverse.put(temp[0],paramNo); j++;

paramNo++;

}

for(int i=pp+1;i<parts.length;i++)

{

parts[i]=parts[i].replace(",", ""); String splits[]=parts[i].split("=");

String name=splits[0].replaceAll("&", ""); aptab.put(aptabInverse.get(name),splits[1]);

}

int i=mdtp-1; while(!mdt.get(i).equalsIgnoreCase("MEND"))

{

String splits[]=mdt.get(i).split("\\s+"); fr.write("+"); System.***out***.println("+");

for(int k=0;k<splits.length;k++)

{

if(splits[k].contains("(P,"))

{

containing number value=aptab.get(Integer.*parseInt*(splits[k]));

}

else

{

}

splits[k]=splits[k].replaceAll("[^0-9]", "");//not String

fr.write(value+"\t"); System.***out***.println(value+"\t");

fr.write(splits[k]+"\t"); System.***out***.println(splits[k]+"\t");

}

fr.write("\n"); System.***out***.println("\n"); i++;

}

}

fr.close();

}

else

{

}

aptab.clear(); aptabInverse.clear();

fr.write(line+"\n"); System.***out***.println(line+"\n");

mntb.close();

mdtb.close(); kpdtb.close(); irb.close();

System.***out***.print("Macro Pass2 Processing done");

}

}

}

**Output:**

