**Practical Assignment 4:**

**Write a Java program for pass-II of a two-pass macro-processor. The output of assignment 3 (MNT, MDT, and file without any macro definitions) should be input for this assignment.**

import java.io.\*;

import java.util.HashMap;

import java.util.Vector;

public class macroPass2 {

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

BufferedReader b1 = new BufferedReader(new FileReader("intermediate.txt"));

BufferedReader b2 = new BufferedReader(new FileReader("mnt.txt"));

BufferedReader b3 = new BufferedReader(new FileReader("mdt.txt"));

BufferedReader b4 = new BufferedReader(new FileReader("kpdt.txt"));

FileWriter f1 = new FileWriter("Pass2.txt");

HashMap<Integer,String> aptab=new HashMap<Integer,String>();

HashMap<String,Integer> aptabInverse=new HashMap<String,Integer>();

HashMap<String,Integer> mdtpHash=new HashMap<String,Integer>();

HashMap<String,Integer> kpdtpHash=new HashMap<String,Integer>();

HashMap<String,Integer> kpHash=new HashMap<String,Integer>();

HashMap<String,Integer> macroNameHash=new HashMap<String,Integer>();

Vector<String>mdt=new Vector<String>();

Vector<String>kpdt=new Vector<String>();

String s,s1;

int i,pp,kp,kpdtp,mdtp,paramNo;

while((s=b3.readLine())!=null)

mdt.addElement(s);

while((s=b4.readLine())!=null)

kpdt.addElement(s);

while((s=b2.readLine())!=null){

String word[]=s.split("\t");

s1=word[0]+word[1];

macroNameHash.put(word[0],1);

kpHash.put(s1,Integer.parseInt(word[2]));

mdtpHash.put(s1,Integer.parseInt(word[3]));

kpdtpHash.put(s1,Integer.parseInt(word[4]));

}

while((s=b1.readLine())!=null){

String b1Split[]=s.split("\\s");

if(macroNameHash.containsKey(b1Split[0])){

pp= b1Split[1].split(",").length-b1Split[1].split("=").length+1;

kp=kpHash.get(b1Split[0]+Integer.toString(pp));

mdtp=mdtpHash.get(b1Split[0]+Integer.toString(pp));

kpdtp=kpdtpHash.get(b1Split[0]+Integer.toString(pp));

String actualParams[]=b1Split[1].split(",");

paramNo=1;

for(int j=0;j<pp;j++){

aptab.put(paramNo, actualParams[paramNo-1]);

aptabInverse.put(actualParams[paramNo-1],paramNo);

paramNo++;

}

i=kpdtp-1;

for(int j=0;j<kp;j++){

String temp[]=kpdt.get(i).split("\t");

aptab.put(paramNo,temp[1]);

aptabInverse.put(temp[0],paramNo);

i++;

paramNo++;

}

i=pp+1;

while(i<=actualParams.length){

String initializedParams[]=actualParams[i-1].split("=");

aptab.put(aptabInverse.get(initializedParams[0].substring(1,initializedParams[0].length())),initializedParams[1].substring(0,initializedParams[1].length()));

i++;

}

i=mdtp-1;

while(mdt.get(i).compareToIgnoreCase("MEND")!=0){

f1.write("+ ");

for(int j=0;j<mdt.get(i).length();j++){

if(mdt.get(i).charAt(j)=='#')

f1.write(aptab.get(Integer.parseInt("" + mdt.get(i).charAt(++j))));

else

f1.write(mdt.get(i).charAt(j));

}

f1.write("\n");

i++;

}

aptab.clear();

aptabInverse.clear();

}

else

f1.write("+ "+s+"\n");

}

b1.close();

b2.close();

b3.close();

b4.close();

f1.close();

}

}

**OUTPUT:**

+ MOVE AREG,10

+ ADD AREG,='1'

+ MOVER AREG,20

+ ADD AREG,='5'

+ MOVER &AREG,100

+ MOVER &BREG,200

+ ADD &AREG,='15'

+ ADD &BREG,='10'