PASS-1 MACROPROCESSOR :  
MAIN PROGRAM:  
import java.util.\*;  
import java.io.\*;  
class MACRO  
{  
static String mnt[][]=new String[5][3]; //assuming 5 macros in 1  
program  
static String ala[][]=new String[10][2]; //assuming 2 arguments in  
each macro  
static String mdt[][]=new String[20][1]; //assuming 4 LOC for each  
macro  
static int mntc=0,mdtc=0,alac=0;  
public static void main(String args[])  
{  
pass1();  
System.out.println("\n\*\*\*\*\*\*\*\*\*PASS-1 MACROPROCESSOR\*\*\*\*\*\*\*\*\*\*\*\n");  
System.out.println("MACRO NAME TABLE (MNT)\n");  
System.out.println("i macro loc\n");  
display(mnt,mntc,3);  
System.out.println("\n");  
System.out.println("ARGUMENT LIST ARRAY(ALA) for Pass1\n");  
display(ala,alac,2);  
System.out.println("\n");  
System.out.println("MACRO DEFINITION TABLE (MDT)\n");  
display(mdt,mdtc,1);  
System.out.println("\n");  
}  
static void pass1()  
{  
int index=0,i;  
String s,prev="",substring;  
try  
{  
BufferedReader inp = new BufferedReader(new FileReader("input.txt"));  
File op = new File("pass1\_output.txt");  
if (!op.exists())  
op.createNewFile();  
BufferedWriter output = new BufferedWriter(new  
FileWriter(op.getAbsoluteFile()));  
while((s=inp.readLine())!=null)  
{  
if(s.equalsIgnoreCase("MACRO"))  
{  
prev=s;  
for(;!(s=inp.readLine()).equalsIgnoreCase("MEND");mdtc++,prev=s)  
{

if(prev.equalsIgnoreCase("MACRO"))  
{  
StringTokenizer st=new StringTokenizer(s);  
String str[]=new String[st.countTokens()];  
for(i=0;i<str.length;i++)  
str[i]=st.nextToken();  
mnt[mntc][0]=(mntc+1)+""; //mnt formation  
mnt[mntc][1]=str[0];  
mnt[mntc++][2]=(++mdtc)+"";  
st=new StringTokenizer(str[1],","); //tokenizing the arguments  
String string[]=new String[st.countTokens()];  
for(i=0;i<string.length;i++)  
{  
string[i]=st.nextToken();  
ala[alac][0]=alac+""; //ala table formation  
index=string[i].indexOf("=");  
if(index!=-1)  
ala[alac++][1]=string[i].substring(0,index);  
else  
ala[alac++][1]=string[i];  
}  
}  
else //automatically eliminates tagging of arguments in definition  
{ //mdt formation  
index=s.indexOf("&");  
substring=s.substring(index);  
for(i=0;i<alac;i++)  
if(ala[i][1].equals(substring))  
s=s.replaceAll(substring,"#"+ala[i][0]);  
}  
mdt[mdtc-1][0]=s;  
}  
mdt[mdtc-1][0]=s;  
}  
else  
{  
output.write(s);  
output.newLine();  
}  
}  
output.close();  
}  
catch(FileNotFoundException ex)  
{  
System.out.println("UNABLE TO END FILE ");  
}  
catch(IOException e)  
{  
e.printStackTrace();  
}  
}  
static void display(String a[][],int n,int m)

{  
int i,j;  
for(i=0;i<n;i++)  
{  
for(j=0;j<m;j++)  
System.out.print(a[i][j]+" ");  
System.out.println();  
}  
}  
}  
/\* INPUT  
START  
MACRO  
INCR &ARG3 &ARG2  
ADD AREG &ARG1  
MOVER BREG &ARG1  
MEND  
MACRO  
PVG &ARG2 &ARG1  
SUB AREG &ARG2  
MOVER CREG & ARG1  
MEND  
INCR  
DECR  
DATA2  
END  
\*/  
/\* OUTPUT  
pvgcoen-3@pvgcoen3-ThinkCentre-M700:~/AA$ javac MACRO.java  
pvgcoen-3@pvgcoen3-ThinkCentre-M700:~/AA$ java MACRO  
\*\*\*\*\*\*\*\*\*PASS-1 MACROPROCESSOR\*\*\*\*\*\*\*\*\*\*\*  
MACRO NAME TABLE (MNT)  
i macro loc  
1 INCR 1  
2 PVG 5  
ARGUMENT LIST ARRAY(ALA) for Pass1  
0 &ARG3  
1 &ARG2

MACRO DEFINITION TABLE (MDT)  
INCR &ARG3 &ARG2  
ADD AREG &ARG1  
MOVER BREG &ARG1  
MEND  
PVG &ARG2 &ARG1  
SUB AREG #1  
MOVER CREG & ARG1  
MEND  
\*/

/\*

Problem Statement: Design suitable data structures and implement pass-I of a two-pass macro-processor using

OOP features in Java

\*/

import java.io.BufferedReader;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;

import java.util.HashMap;

public class macroPass1 {

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

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

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

FileWriter f2 = new FileWriter("mnt.txt");

FileWriter f3 = new FileWriter("mdt.txt");

FileWriter f4 = new FileWriter("kpdt.txt");

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

String s;

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

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

String word[]=s.split("\\s"); //separate by space

if(word[0].compareToIgnoreCase("MACRO")==0){

flag=1;

if(word.length<=2){

f2.write(word[1]+"\t"+pp+"\t"+kp+"\t"+mdtp+"\t"+(kp==0?kpdtp:(kpdtp+1))+"\n");

continue;

}

String params[]=word[2].split(",");

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

if(params[i].contains("=")){

kp++;

String keywordParam[]=params[i].split("=");

pntab.put(keywordParam[0].substring(1,keywordParam[0].length()),paramNo++);

if(keywordParam.length==2)

f4.write(keywordParam[0].substring(1,keywordParam[0].length())+"\t"+keywordParam[1]+"\n");

else

f4.write(keywordParam[0].substring(1,keywordParam[0].length())+"\t"+"-"+"\n");

}

else{

pntab.put(params[i].substring(1,params[i].length()),paramNo++);

pp++;

}

}

f2.write(word[1]+"\t"+pp+"\t"+kp+"\t"+mdtp+"\t"+(kp==0?kpdtp:(kpdtp+1))+"\n");

kpdtp+=kp;

}

else if(word[0].compareToIgnoreCase("MEND")==0){

f3.write(s+'\n');

flag=pp=kp=0;

mdtp++;

paramNo=1;

pntab.clear();

}

else if(flag==1){

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

if(s.charAt(i)=='&'){

i++;

String temp="";

while(!(s.charAt(i)==' '||s.charAt(i)==',')){

temp+=s.charAt(i++);

if(i==s.length())

break;

}

i--;

f3.write("#"+pntab.get(temp));

}

else

f3.write(s.charAt(i));

}

f3.write("\n");

mdtp++;

}

else{

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

}

}

b1.close();

f1.close();

f2.close();

f3.close();

f4.close();

}

}

/\*

OUTPUT:

neha@neha-1011PX:~/Desktop/neha\_SPOS/Turn1/A3$ javac macroPass1.java

neha@neha-1011PX:~/Desktop/neha\_SPOS/Turn1/A3$ java macroPass1

neha@neha-1011PX:~/Desktop/neha\_SPOS/Turn1/A3$ cat intermediate.txt

M1 10,20,&b=CREG

M2 100,200,&u=AREG,&v=BREG

neha@neha-1011PX:~/Desktop/neha\_SPOS/Turn1/A3$ cat mnt.txt

M1 2 2 1 1

M2 2 2 7 3

M3 2 0 13 4

neha@neha-1011PX:~/Desktop/neha\_SPOS/Turn1/A3$ cat mdt.txt

MOVE #3,#1

ADD #3,='1'

MOVER #3,#2

M2 69,169

ADD #3,='5'

MEND

MOVER #3,#1

MOVER #4,#2

M3 73,173

ADD #3,='15'

ADD #4,='10'

MEND

ADD #1,#2

MEND

neha@neha-1011PX:~/Desktop/neha\_SPOS/Turn1/A3$ cat kpdt.txt

a AREG

b -

u CREG

v DREG

\*/