4. Pass-2 Macroprocessor

Aim :- Design a MACRO PASS-2

Problem statement :- Write a Java program for pass-II of a 2-pass macro processor. The output of assignment-3 (MNT, MDT and file without any macro definitions) should be input for this assignment.

Theory :-

Basic tasks performed by Macro processor

a) Recognize macro def^n
b) Save the def^n
c) Recognize call
d) Expanded calls and substitute arguments.

- Pass I Macro def^n
- Pass 2 Macro calls and Expansion.

INPUT :-

```
MACRO
INCR1 &FIRST &SECOND = DATA9
A     1,&FIRST
BL    2,&SECOND
MEND  MACRO
INCR2 &ARG1, &ARG2 = DATA5
```

```
L     3, &ARG1
ST    4, &ARG2
MEND
PRG2  START
      USING *, BASE
      INCR1 DATA1
      INCR2 DATA3, DATA4
FOUR  DC    F'4'
FIVE  DC    F'5'
BASE  EQU   8
TEMP  DS    1F
      DROP  8
      END
```

Output :-

== Pass1 ==

ALA:

```
[&FIRST, &SECOND]
[&ARG1, &ARG2]
```

MNT:

```
[INCR1, 0]
[INCR2, 4]
```

MDT:

```
      &FIRST, &SECOND = DATA9
1     ~~ARG~~
A     1, #0
L     2, #1
MEN
```

```
D        &ARG1, &ARG2 = DATA5
INCR
  2
   L      3, #0
  ST      4, #1
  MEN
   D
```

== PASS 2 ==

MDT :

```
INCR     &FIRST, &SECOND =DATA9
  1
  A        1, #0
  L        2, #1
MEN
  D      &ARG1, &ARG2 = DATA5
INCR
  2
  L        3, #0
 ST        4, #1
MEN
  D
```

```
PRG2     STAR
           T        *, BASE
         USIN
           G
           A        1, DATA1
           L        2, DATA9
           L        3, DATA3
           ST       4, DATA4
```

FOUR	DC	F'4'
FIVE	DC	F'5'
BASE	EQU	8
TEMP	DS	1F
	DRO P	
	END	

ALA:

[DATA1, DATA9]

[DATA3, DATA4]

conclusion:-

Thus pass II of Macro processor is implemented and ALA file is generated.

Assignment No. 04 [PASS-2 Macroprocessor]

Problem Satement: 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.

---

1. Pass 2 Macro Code:

```
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:

OUTPUT:
Pritam-spos@Pritam-HP:~/SPOSL$ javac macroPass2.java
Pritam-spos@Pritam-HP:~/SPOSL$ java macroPass2
Pritam-spos@Pritam-HP:~/SPOSL$ cat Pass2.txt

Intermediate - -
M1 10,20,&b=CREG
M2 100,200,&u=&AREG,&v=&BREG



Kpdt--
a       AREG
b       -
u       CREG
v       DREG
```

pass2 --
+ MOVE AREG,10
+ ADD AREG,='1'
+ MOVER AREG,20
+ ADD AREG,='5'
+ MOVER &AREG,100
+ MOVER &BREG,200
+ ADD &AREG,='15'
+ ADD &BREG,='10'

MNT --

M1	2	2	1	1
M2	2	2	6	3

MDT --
MOVE #3,#1
ADD #3,='1'
MOVER #3,#2
ADD #3,='5'
MEND
MOVER #3,#1
MOVER #4,#2
ADD #3,='15'
ADD #4,='10'
MEND