3. Pass 1 - Macroprocessor

Aim:- To design Data Structure for Microprocessor

Problem statement:- Design suitable data structures and implement pass-I of a 2-pass macro processor using OOP features in Java.

Theory:-

1. Macro processor (Definition)
A macro processor is a program that reads a file (or files) and scans them for certain keywords. When a keyword is found it is replaced by some text. The keyword/text combination is called a Macro

Algorithm:- A one-pass macro processor the alternate between macro def & macro expansion algorithms.

Algorithm:

```
begin {macro processor}
    EXPANDING := FALSE
        while OPCODE <> 'END' do
        begin
            GETLINE
            PROCESSLINE
        end {while}
procedure end {macro processor}
procedure PROCESSLINE
```

```
begin
    search NAMTAB for OPCODE
    if found then
        EXPAND
    else if OPCODE = 'MACRO' then
        DEFINE
    else write source line to expand file
    end {PROCESSLINE}
```

Algorithm :-

```
procedure EXPAND
    begin
    EXPANDING := TRUE
    get first line of macro def^n {prototype} from DEFTAB
set up arguments from macro invocation in ARGTAB
write macro invocation to expanded file as a comment
while not end of macro def^n do
    begin
        GETLINE
        PROCESSLINE
        end {while}
        EXPANDING := FALSE
    end {EXPAND}

procedure GETLINE
    begin
    if EXPANDING then
begin get next line of macro def^n from DEFTAB
substitute arguments from ARGTAB for positional
notation
    end {if}
```

else
read next line from input file
end {GETLINE}

Input :-

```
MACRO  INCR &X &Y &REG1
       ADD REG &Y
       MOVEM &REG1 &X
MEND
       START 100
       READ N1
       READ N2
       INCR N1 N2
       STOP
       N1 DS1
       N2 DS2
       END
```

```
C:\ABC > javac macro.java
C:\ABC > java macro
MACRO  INCR  &X  &Y  &REG1
  MOVER      &REG1  &X
  ADD        &REG1  &Y
  MOVEM      &REG1  &X
MEND
     START   100
     READ    N1
     READ    N2
     INCR    N1   N2
     STOP
     N1      DS   1
     N2      DS   2
     END
```

PAGE :
DATE : / /

Conclusion :-

Thus pass I of macro processor is implemented and MNT, MDT & ALA file is generated.

Assignment No. 03 [PASS-1 Macroprocessor]

Problem Satement: Design suitable data structures and implement pass-I of a two-pass macro-processor using OOP features in Java.

1. Pass 1 Macro Code:

```
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:

Pritam-spos@Pritam-HP:~/SPOSL$ javac macroPass1.java
Pritam-spos@Pritam-HP:~/SPOSL$  java macroPass1
Pritam-spos@Pritam-HP:~/SPOSL$  cat intermediate.txt
M1 10,20,&b=CREG
M2 100,200,&u=AREG,&v=BREG

Pritam-spos@Pritam-HP:~/SPOSL$  cat mnt.txt
M1      2       2       1       1
M2      2       2       7       3
M3      2       0       13      4

Pritam-spos@Pritam-HP:~/SPOSL$  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

Pritam-spos@Pritam-HP:~/SPOSL$  cat kpdt.txt
a       AREG
b       -
u       CREG
v       DREG

*/
```