SRES's Sanjivani College of Engineering, Kopargaon
(*An Autonomous Institute*)
Department of Computer Engineering

SPOS Lab Manual

Assignment No. 03

Title: Implementation of Pass 1 of Two Pass Macroprocessor

Aim :
Design suitable data structures and implement pass-I of a two-pass macro-processor using OOP features in Java/Python

Inputs :
1. Assembly language program containing macro definitions and macro calls

Outputs :
1. Assembly Language Program without macro definiton but with macro calls
2. Macro Definition Table
3. Macro Name Table
4. Argument List Array

Theory :
Pass 1 of macro pocessor perform two important tasks
Recognize macro definitions – A macro processor must recognize macro definitions identified by the MACRO and MEND pseudo -opcodes
Save the definitions – The macro processor must store the macro instruction definitions, which it will use for expanding macro calls

Pass I data structures
1. **Input Assembly Language Program with macro definitions and macro calls**
   Input Assembly Language Program containing few assembly instructions, one or macro definitions and macro calls

2. **Macro Name Table (MNT)**
   used to store the names of defined macros
   Each MNT entry consists of a character string (the macro name) and a pointer(index) to the entry in the MDT that corresponds to the beginning of the macro definition

3. **Macro Name Table Counter (MNTC)**
   used to indicate the next available entry in the MNT

4. **Macro Definition Table (MDT)**
   used to store the body of macro definitions
   Every line of each macro definition, except the MACRO line, is stored in the MDT. The MEND is kept to indicate the end of definition; and the macro name line is retained to facilitate keyword argument replacement.

5. **Macro Definition Table Counter (MDTC)**
   used to indicate the next available entry in the MDT

6. **Argument List Array (ALA)**
   used to substitute index markers for dummy arguments before storing a macro definitions

7. **Assembly Language Program without macro definiton but with macro calls**
   All the assembly instructions excluding macro definitions will be stored in the output assembly program

Sample Example Showing Inputs and Outputs of Pass1 Assembler

Input Assembly Language Program with macro definitions and macro calls

```
        START
        MOVER    AREG, A
        MOVEM    BREG, B
        MACRO
        INCR     &ARG1, &ARG2
        ADD      AREG, &ARG1
        ADD      AREG, &ARG2
        MEND
        MACRO
        DECR     &ARG3, &ARG4
        SUB      BREG, &ARG3
        SUB      BREG, &ARG4
        MEND
        INCR     DATA1, DATA2
        DECR     DATA3, DATA4
DATA1   DC       5
DATA2   DC       10
DATA3   DC       15
DATA4   DC       20
        END
```

Output Assembly Language Program without macro definiton but with macro calls

```
        START
        MOVER    AREG, A
        MOVEM    BREG, B
        INCR     DATA1, DATA2
        DECR     DATA3, DATA4
DATA1   DC        5
DATA2   DC       10
DATA3   DC       15
DATA4   DC       20
        END
```

Macro Name Table (MNT)

MNT Index	Macro Name	MDT Index
1	INCR	1
2	DECR	5

Argument List Array (ALA)

Index	Argument Name
1	&ARG1
2	&ARG2
3	&ARG3
4	&ARG4

Macro Definition Table (MDT)

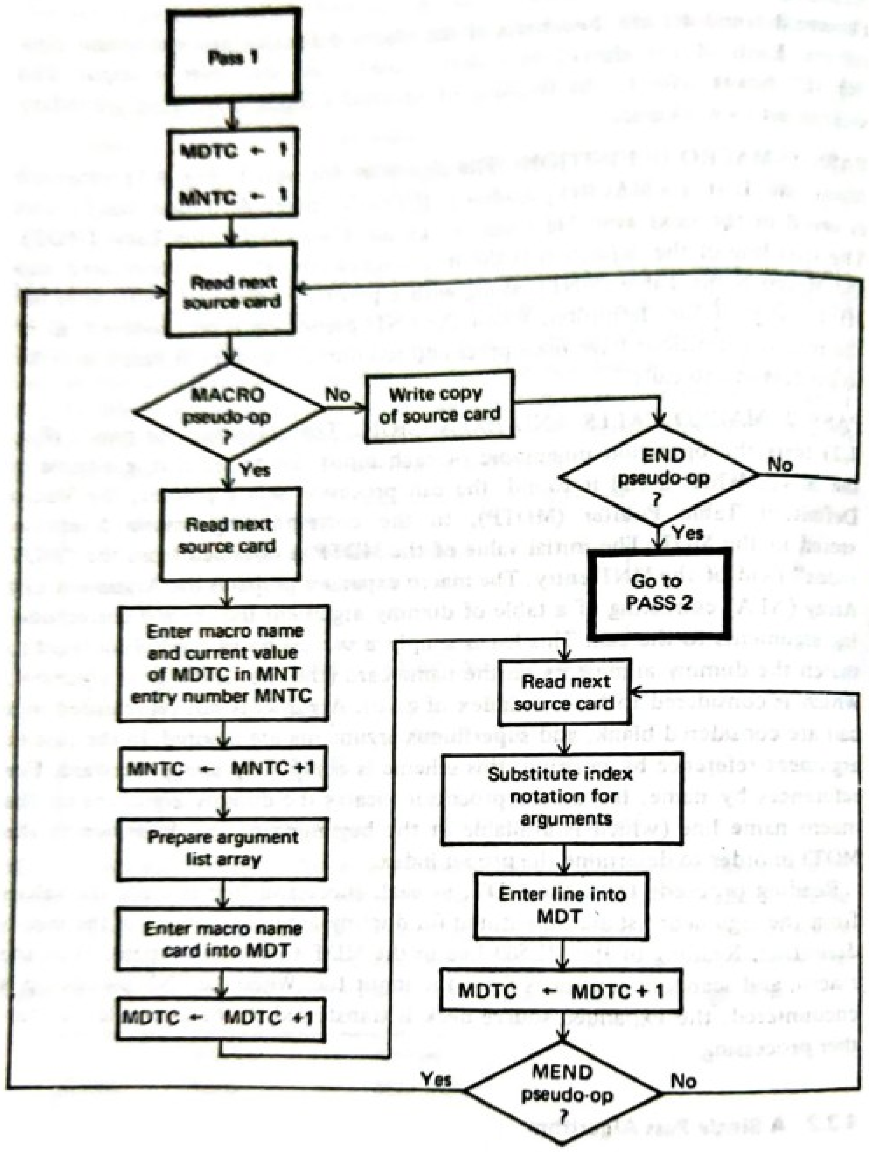
MDT Index	Macro Instructions
1	INCR &ARG1, &ARG2
2	ADD AREG, #1
3	ADD AREG, #2
4	MEND
5	DECR &ARG3, &ARG4
6	SUB BREG, #3
7	SUB BREG, #4
8	MEND

Algorithm for Pass 1 of a two-pass assembler

1. Test mnemonic opcode of each instruction from input assebly language program
2. If mnemonic opcode is a MACRO pseudo-op then enter the macro name and current value of MDT index into the MNT
3. Enter the dummy arguments into the ALA, with proper index is assigned to them
4. Enter the whole macro definition, from macro name instruction to MEND instruction into the MDT
5. Repeat steps 2 to 4 for all macro definitions found in the input program
6. All other instructions, excluding macro definitions should be entered into the output assemby language program file of pass1

7. When END pseudo -op is encountered, all of the macro definitions have been processed so control transfers too pass2

Flowchart of Pass1 of Macro Processor



Conclusion: In this assignment we have implemented pass 1 of Macroprocessor. Input assembly language program with macro definitions is processed to generate MDT, MNT, ALA and output file without macro definitions.

References: Systems Programming by John J. Donovan

Prepared by
Prof.N.G.Pardeshi
Subject Teacher

Approved by
Dr. D.B.Kshirsagar
HOD