Roll No 17:

3rd Assignment

Nikita D Bhosale

GURUKUL | Page No. | Date / /

Aim:- To design Data Structure for macroProcessor

Problem stat :- Design suitable data structure and implement Pass I of a two Pass MacroProcessor using oop feature in Java

Theory :-

① **MacroProcessor :-**

- It is a Program that reads a file and scans them for certain keywords
- when a keyword is found it is replaced by some text
- The keyword / text combination is called MACRO

② Basic tasks Performed by MacroProcesses

- Recognize MACRO definition
- save the defination
- Recognize call
- expanded calls and substitute arguments.

③ MACRO defination Part :-

1) Macro Prototype Statement :- declares the name of macro and types of parameters

2) Model statement :- statement for which assembly language statement is generated during macro expansion.

3) preprocesses statement :- used to perform auxillary function during macro expansion.

④ MACRO CALL and expansion :-

- appearance of a macro name in the memory field loads to macro call
- Macro call replaces such statement by sequence of stat^nt comprising the macro This known as Macro expansion.

⑤ Implementation Logic :-

1) Defination Processing :-

- Scan all macro defination and for each macro defination into MACRO name in the MACRO NAME TABLE
- Store entire defination in the MACRO defination table and add info into MNT

2) MACRO Expansion :-

- examine all statement in assembly source prog to detect the macro calls
- for each macro call locate the macro in MNT retrive MDTP establish the correspondance between formal and actual parameteres and expand macro

⑥ Data structure required for macro defination processing

① MacroNAme table (MNT)
② Parameter NAME Table (PNT)
③ Keyword paramet & default table position
④ Macro defination table

Algorithm :-

```
begin
    GETLINE
    PROCESSLINE
    end {while}
end {macro processor}
Procedure PROCESSLINE
  begin
    search NAHTAB for OPCODE
    if found them
      EXPAND
    else if OPCODE = "MACRO" then
      DEFINE
    else write source line to expanded file.
    end {PROCESSLINE}

Procedure EXPAND.
    begin
    EXPANDING := TRUE
    get first line of macro defination {PrototYpe
      from DEFTAB.
    set up arguments from macro innovative
    in ARGTAB
    Write macro innovation to expanded
    file as a comment
    while not end of macro definition doc.
      begin
        GETLINE
        PROCESSLINE
      end{while}
  EXPANDIND := FALSE
end {EXPAND}
```

GURUKUL | Page No. | Date / /

```
Procedure GETLINE
  begin
    IF EXPANDING then
    begin get next line of macro defination
        from DEFTAB
        substitute arg from ARGTAB.
        for posetional notation
    end {IF}
    else
        read next line from input file.
  end {GETLINE}
```

Conclusion:-

Thus Pass I of macroprocesses is implemented and MNT, MDT & ALA file is generated.