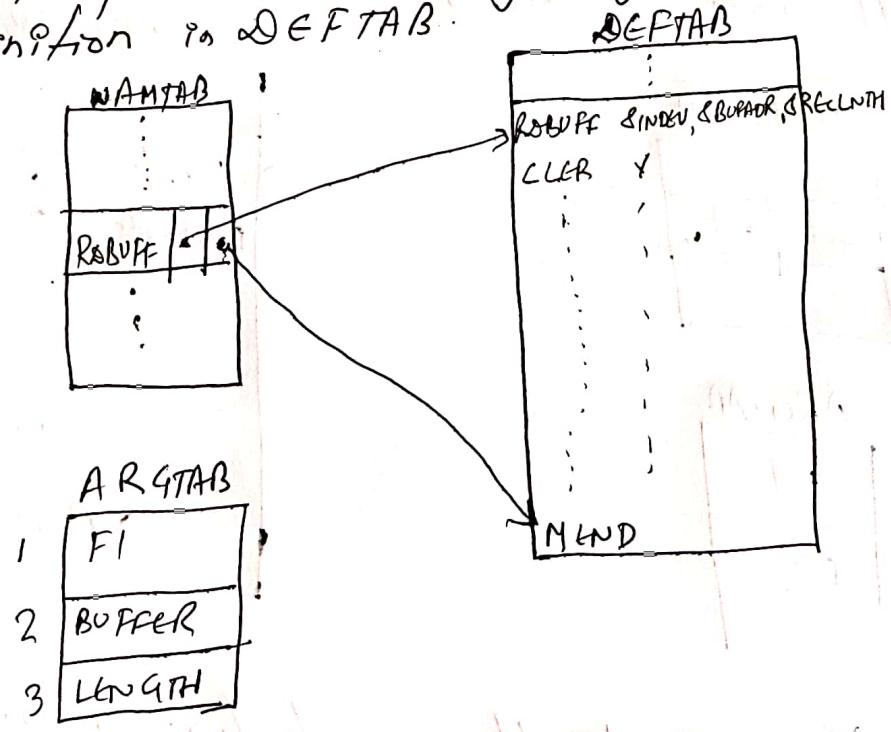
Try to make short definition of the following from the notes which I have sent as well as the scanned file which we have discussed during online classes.

① Macro
② Macro definition
③ Macro expansion.
* ④ Macro processor functions (~~FIRST~~ like macro definition, macro expansion, macro invocation)

* ⑤ Data Structures involved.
Mainly study about data structures involved in one pass macro processor.
① DEFTAB (Definition table)
② NAMTAB (Name table)
③ ARGTAB (Argument)

Note: The macro names are entered into NAMTAB. NAMTAB contains two pointers to the beginning & the end of the definition in DEFTAB.



PU Question: 2018, 19, 20 ...

* Consider the macro definition given below & show macro expansion for the macro call statement "Print 54 F2". Show all data structures used by macro processor clearly.

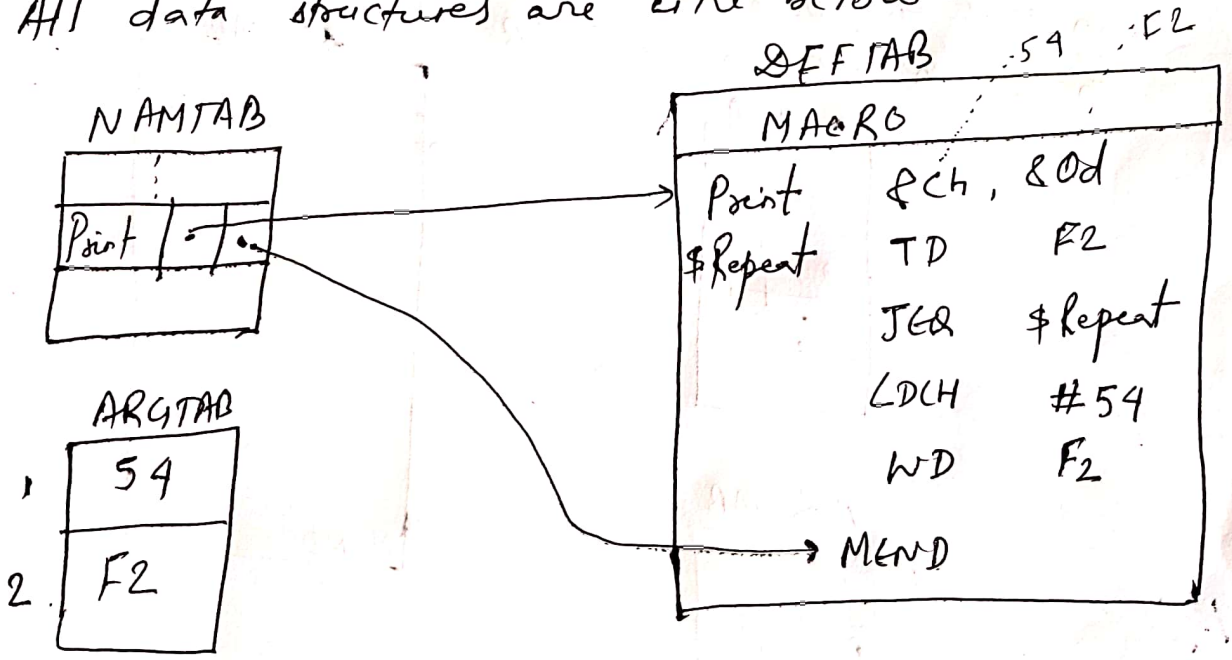
(5)

Print	MACRO	&ch, &od
$Repeat	TD	&od
	JEQ	$Repeat
	LDCH	#&ch
	WD	&od
	MEND	

→ · Macro expansion for the statement "Print 54 F2"

```
        Print    54, F2
$AARepeat TD     F2
        JEQ      $AARepeat
        LDCH     #54
        WD       F2
```

∴ All data structures are like below.



∴ You can do this much for 5 marks.
Two parameter &ch & &od are Two argument 54, F2 pass garko cha.

(2)

Differentiate Two Pass Vs One pass Macro processor.

Two pass Macro Processor	One pass Macro Processor.
• Passes: Pass1: Recognise macro definitions. Pass2: Recognize macro calls.	• Every macro must be defined before it is called.
• Nested macro definitions are not allowed.	• One-pass processor can alternate between macro definition & macro expansion.
	• Nested macro definitions are allowed but nested calls are not.

Imp. Machine Independent Features.

★ ① Concatenation of Macro Parameters.

- No more lengthy explanations are required. Just see the notes and scanned file which I have sent. Try to make & write short note during exam. That is more than enough.

② Generation of Unique Levels.

It allows the creation of special types of labels within macro instructions.

Duplicate labels problem हटाउन लाई use हुन्छ. भनेर Example पनि दिएका छौं। Refer two notes which I have sent.

③★ Conditional Macro expansion.

This is frequently asked topic.

You need not show whole programming example during exam. Only basic concept is required.

How to write then ??

③

Conditional macro expansion:

→ Most macro processors can modify the sequence of statements generated for a macro expansion, depending on the arguments supplied in the macro invocation.

→ This capability adds greatly to power and flexibility of a macro language.

```
MACRO    &COND
.....
IF (&COND NE '')
   part 1
ELSE
   part 2
ENDIF
ENDM.
```

part 1 is expanded if condition part is true, or part 2 is expanded.

. Compare operator: NE, EQ, LE, GT.

→ We allow conditional assembly so that it can generate code that is suitable for a particular application.

→ There are some macro-time control structures introduced.

- IF-ELSE-ENDIF
- WHILE-ENDW

→ Macro-time variables (also called a set symbol) can be used to store values that are used by these macro-time control structures.

→ Macro-time conditional statements.

. Macro-processor directives

① IF-ELSE-ENDIF

② SET.

→ Macro processor function.

. %NITEMS: is a macro processor function that returns as its value the number of members in the argument list.

Eg: &EOR = (00, 03, 04)

⇒ %NITEMS(&EOR) is 3

. %NITEMS() occurs at macro expansion time.

→ The testing of Boolean expression in IF statements occurs at the time macros are expanded.

→ COMPR instruction test data values during program execution.

→ When an IF statement is encountered during expansion of a macro,

- If TRUE, the macro processor continues to process until next ELSE or ENDIF.
  - If ELSE encountered, then skips to ENDIF.
- Otherwise, the assembler skips to ELSE & continues to process until it reaches ENDIF.

→ When WHILE statement is encountered,

- If TRUE,
  - → the macro processor continues to process line from DEFTAB until it encounters the next ENDW.
  - → When ENDW is encountered, the macro processor returns to the preceding WHILE & re evaluates.
- Otherwise
  - The macro processor skips ahead in DEFTAB until it find the next ENDW statement & then resumes normal macro expansion.

While studying conditional macro conditional expansion, we came to the term macro time variable. What is it??

Question: What is macro time variable? How macro processor manages value of macro time variable?

Solution: Macro time variable, also called as set symbol, is used to store values are used by these macro time control structures:

- IF, ELSE - ENDIF
- WHILE - ENDW.

- It begins with "&" but is not a macro instruction parameter → any symbol that begins with the character & and is not a macro parameter.
- Used to store working values during the macro expansion.
  - store the evaluation result of Boolean expression.
  - Control the macro-time conditional structures.
- Be ~~initiated~~ initialized to 0
- Be changed with their values using SET directives
  - &EQORCK SET 1.

Example:

```
RDBUFF   MACRO   &INDEV,&BUFADR,&RECLTH,&EOR,
                 &MAXLETH
         IF      (&EOR NE '')
&EQORCK  SET     1
         ENDC
         CLEAR   X
         CLEAR   A
          .
          .
          .
         MEND
```

Macro-time variable

→ Here, the symbol &EQORCK is macro time variable. It is initialized to value 0. If there is argument corresponding to &EOR (i.e, if &EOR is not null), the variable &EQORCK is set to 1, otherwise to 0.

→ Use of macro time variable makes it clear that the same logical condition is involved in both IF statements.

→ The macro processor must maintain a symbol table that contains the value of all macro-time variables used. Entries in this table are made or modified when SET statements are processed. The table is used to look up the current value of a macro-time variables whenever it is required. //

④ Keyword Macro Parameters.
Look the notes & scanned copy discussed.
[Might be asked in short note].

Concept: हामीलाई positional parameter थाहा भयो / जसमा सबै parameter को लागि Argument को position पर्छ].

MACRO &INDEV, &RECLENTH, &BUFADR, &EOR
↓ ↓ ↓ ↓
F2, 20, BUFFER, 1.

[4 ota parameter, 4 ota argument.]
Keyword parameter ले चाहिं धेरै सबै position मा राख्न / पनि Corresponding values assign गर्न मद्दत गर्छ।
[parameter = argument]
Example. MACRO &INDEV = F2, &EOR = 1, &RECLENTH ...
. Argument लाई order मा राख्नु पनि पर्दैन।

Look notes !!!

MACRO PROCESSOR DESIGN OPTIONS:

① Recursive macro expansion.
[Short note को लागि महत्वपूर्ण Topic हो].
. If we want to allow a macro to be invoked in a macro definition, the already presented macro processor implementation cannot be used.
. EXPAND routine recursively called हुन्छ र variable used by it (eg. EXPANDING) is not saved across these calls.
. Invokation argument is ARGTAB will be overwritten.
. Different ARGTAB आउन गर्छ।
. Solution के त??
① Write macro processor in programming language that allows recursive calls. Thus local variables will be retained.
② Use stack to care of pushing & popping local variables and return address.

Identify the addressing mode & target address of the instruction is 012030

PC = 2000
B = 5030
X = 3000.

[PU Question]

Solution ??

Here Give instruction = 012030

PC = 2000
B = 5030
X = 3000

opcode	n	i	x	b	p	e	disp
000000	0	1	0	0	1	0	0000 0011 0000
0	1			2			3 0

Addressing mode = Immediate Addressing mode.

Target address = PC + disp
= 2000 + 030
= 2030 //

I hope you have gone through the videos of program linking and relocating loader. Make short descriptive note by yourself. [Refer to notes & videos].

- In relocating loader the concept of relocation bit, and modification bit should be clearly explained.

Solve Object Generation from Past Question Sets.