Q1. what is macro call and macro definition

Ans. → macro definition

- A macro is a group of commonly used statements in the source programming language.
- Macro definition consists of:

1. Macro prototype
2. One or more model
3. Macro preprocessor

- macro defination is in between the macro header statement and a macro ~~x~~ end statement.

→ Macro call

- A macro is called by writing the macro name with actual parameters in an assembly program
- The macro call has the following syntax:
  <macro name> [<list of parameters>]
- For eg:
  INCR X
  will call the macro INCR with the actual parameter X.
- A macro call leads to macro expansion.

Q2. Explain macro definition table and macro name table table

Ans. → Macro Definition Table:-

- It stores the macro definition including macro prototype and macro body.
- Comment lines are omitted.
- References to the macro instruction parameters are converted to a positional notation for efficiency in substituting arguments.
- The Fields used in MDT include:-

1. label
2. opcode
3. Operands

→ Macro Name Table :-

- It is used to store macro names
- It serves as an index to Macro Definition Table (MDT).
- Pointers to the beginning and the end of the macro Definition Table (MDT).
- Fields in MNT include:

1. macro name
2. Number of positional parameter
3. Number of keyword parameter
4. Number of expansion time variables
5. MDT pointer (MDTP)
6. KPDTAB pointer (KPDTP)
7. SSTAB pointer (SSTP)

Q3. Explain Advance macro Facilities in detail.

Ans. Advanced macro facilities permit conditional reordering of the sequence of macro expansion.

- It allows conditional ~~reordering~~ selection of the machine instructions that appear in expansion of macro call
- Flow of control during macro expansion can be altered using:

1. Conditional branch Pseudo-opcode AIF
2. Unconditional branch Pseudo-opcode AGO.

→ AIF:

- AIF is similar to IF statement, the label used for branching is known as sequencing symbol.

- A sequencing symbol has the syntax!
  <ordinary string>

→ AGO:
- AGO is similar to GO TO statement
- An AIF statement has the syntax
  AIF (<expression>) <sequencing symbol>
- An AGO statement has the syntax
  AGO <sequencing symbol>

→ ANOP:
- The instruction ANOP does no operation.
- A sequencing symbol can be specified in the name field of an ANOP instruction.

Q4. Explain handling of nested macro call

Ans 1. Several levels of expansions:
- The macro call
  COMPUTE1 X, Y, Z
  can be expanded (level 1) using the algorithm of macro expansion.

$$\text{COMPUTE1 X, Y, Z} \Rightarrow \left\{ \begin{array}{l} \text{COMPUTE X} \\ \text{COMPUTE Y} \\ \text{COMPUTE Z} \end{array} \right\}$$

- The expanded code itself contains macro calls:
- The macro expansion algorithm can be applied to the first level expanded code to expand these macro calls and so on, until we obtain a code form which does not contain any macro calls.

2. Recursive expansion!
- During recursion, while processing one macro the processing of inner macro can begin and after the expansion of inner macro finishes, the processing of

outer macro may continue

3 Use of stack during expansion:

- Nested macro calls can be handled with the help of explicit stack.
- macro calls are handled in LIFO manner.
- stack can be used to accomodate the expansion time data structure.
- Every call to a macro involves pushing an activation record onto the stack.
- At the end of the macro expansion, an activation record is removed from the stack.

Q5. Explain formal, actual, positional and keyword parameter.

Ans. 1. ~~En~~ Formal parameters:

- These are variables that are defined in the macro description and are used to define how the macro should process the data.
- They are always variables and require a data type.

2. Actual parameters:

- These are the values that are referenced in the parameter index of a subprogram call.
- They can be variables, constants, expressions, or even function calls, and they don't need to be variables.

3. Positional parameters:

- A positional parameter is written as &parameter_name. For eg, in the statement

→ INCR &VARIABLE, &INCR_BY, &USE_REG

~~Eg~~ VARIABLE, INCR_BY, USE_REG are positional parameters.

4. Keyword parameters;
- Keyword parameters are used for following purposes:

1) Default value can be assigned to the parameter
2) During a call to macro, a keyword parameter is specified by its name. It takes the following form:

   <parameter name> = <parameter value>

Q6. Explain in detail what kind of Data structure use for your pass II code.

Ans. Pass II uses the following data structures:

1. • Input source program for pass-2
   • It is produced by pass-1.
2. macro definition table (MDT) produced by pass-I.
3. macro name table (MNT) produced by pass-I.
4. MNTP (macro name table pointer) gives number of entries in MNT.
5. Argument list array (ALA) gives association between integer indices and actual parameters.
6. • Source program with macro-calls expanded.
   • This is output of pass-2.
7. MDTP (macro defination table pointer)
   • It gives the address of macro definition in macro defination table.