R (2)	C (4)	V (4)	Total (10)	Dated Sign

Assignment No. Group A(Macro)

Problem Definition:

Write a Java program for pass-II of a two-pass macro-processor. The output of assignment-3 (MNT, MDT and file without any macro definitions) should be input for this assignment.

1.1 Prerequisite:
Basic concepts of Macro processor

1.2 Learning Objectives:
- To Identify and create the data structures required in the design of macro processor.
- To Learn parameter processing in macro

1.3. Problem Statement:
Write a program to create pass-II Macro-processor

1.4. Outcomes:
After completion of this assignment students will be able to:
- Understand the Programming language of Java

- Understand the concept of Pass-II Macro-processor

1.5. Software Requirements:

Linux OS, JDK1.7

1.6. Hardware Requirement:
- 4GB RAM , 500GB HDD

1.7. Theory Concepts:

Pass II:
Replace every occurrence of macro call with macro definition. (Expanded Code)
There are four basic tasks that any macro instruction process must perform:

1.7.1. Recognize macro definition:
A macro instruction processor must recognize macro definition identified by the MACRO and MEND pseudo-ops. These tasks can be complicated when macro definition appears within macros. When MACROs and MENDs are nested, as the macro processor must recognize the nesting and correctly match the last or or outer MEND with first MACRO. All intervening text, including nested MACROs and MENDs defines a single macro instruction.

1.7.2 Implementation of a 2 pass algorithm

1. We assume that our macro processor is functionally independent of the assembler and that the output text from the macro processor will be fed into the assembler.
2. The macro processor will make two independent scans or passes over the input text , searching first for macro definitions and then for macro calls

3. The macro processor cannot expand a macro call before having found and saved the corresponding macro definitions.
4. Thus we need two passes over the input text , one to handle macro definitions and other to handle macro calls.
5. The first pass examines every operation code, will save all macro definitions in a macro Definition Table and save a copy of the input text, minus macro definitions on the secondary storage.

6. The first pass also prepares a Macro Name Table along with Macro Definition Table As seen in the previous assignment that successfully implemented pass – I of macro pre-processor. The second pass will now examine every operation mnemonic and replace each macro name with the appropriate text from the macro definitions.

1.7.3 Pass 2 databases:

1. The copy of the input source deck obtained from Pass- I
2. The output expanded source deck to be used as input to the assembler
3. The Macro Definition Table (MDT), created by pass 1 thus we have successfully implemented pass-II of a two-pass macro-processor
4. The Macro Name Table (MNT), created by pass 1
5. The Macro Definition Table Counter (MNTC), used to indicate the next line of text to be used during macro expansion
6. The Argument List Array (ALA), used to substitute macro call arguments for the index markers in stored macro definition

1.8. Conclusion:

Thus we have successfully implemented pass-II of a two-pass macro-processor

Assignment Questions: [Write short answer]

1. Give the data structures of macro processor.
2. Explain macro expansion.
3. Explain nested Macro calls