Amity University, Mumbai
Amity School of Engineering and Technology
Department of Computer Science and Engineering

Lab Assignment

Concept: Macro **Submission Date: 03/04/2020**

1. Write a program to design a macro to perform add operation.
   Example:

```
MACRO
INCR &MEM_VAL, &INCR_VAL, &REG
MOVER &REG, &MEM_VAL
ADD &REG, &INCR_VAL
MOVEM &REG, &MEM_VAL
MEND
```

2. Write a program to create output TEXT file of macro definition from the given input assembly source code.
3. Write a program to implement nested macro definition.
   Example:

```
        TEST            START           2000h
        MACROS          MACRO
        CELTOFER MACRO          &CEL    &FER
                        LDA             &CEL
                        MULT            NINE
                        DIV             FIVE
                        ADD             THIRTYTWO
                        STA             &FER
                        MEND
                        MEND

MACROF                  MACRO
CELTOFER MACRO                  &CEL    &FER
                        LDAF            &CEL
                        MULTF           NINE
                        DIV             FFIVE
                        ADD             FTHIRTYTWO
                        STAF            &FER
                        MEND
                        MEND
```

4. Write a program to implement One Pass Macro.

   A one-pass macro processor that alternate between macro definition and macro expansion in a recursive way is able to handle recursive macro definition. Because of the one-pass structure, the definition of a macro must appear in the source program before any statements that invoke that macro.

Algorithm:

```
    begin {macro processor}
        EXPANDING : = FALSE
            while OPCODE ≠ 'END' do
              begin
                 GETLINE
                 PROCESSLINE
                 end  {while}
            end {macro processor}
        procedure PROCESSLINE
            begin
              search NAMTAB for OPCODE
              if found then
                 EXPAND
             else if OPCODE = 'MACRO' then
                 DEFINE
             else write source line to expanded file
             end {PROCESSLINE}
```

Algorithm:

```
procedure EXPAND
    begin
      EXPANDING : = TRUE
      get first line of macro definition {prototype} from DEFTAB
      set up arguments from macro invocation in ARGTAB
      write macro invocation to expanded file as a comment
      while not end of macro definition do
         begin
           GETLINE
           PROCESSLINE
           end {while}
         EXPANDING : = FALSE
    end  {EXPAND}

procedure GETLINE
    begin
      if EXPANDING then
         begin                get next line of macro definition from DEFTAB
           substitute arguments from ARGTAB for positional notation
         end {if}
    else
```

```
          read next line from input file
end {GETLINE}
```