**Practical - 20**

**Aim:** Write an assembly code to compare two strings. Print appropriate message “same” or “not same” on output string.

**Description of instructions used:**

**MSG:** Here MSG stands for message here in place of MSG we will also use STRING.

**DB** **(DEFINE BYTE):** The **DB** directive is used to declare a byte type variable, or a set aside one or more storage locations of type byte in memory.

**INT:** INT is an assembly language instruction for x86 processors that generates a software interrupt. It takes the interrupt number formatted as a byte value.

**CMP:** The CMP instruction compares two operands. It is generally used in conditional execution. This instruction basically subtracts one operand from the other for comparing whether the operands are equal or not. It does not disturb the destination or source operands.

**JNE:** The JNE (or JNZ) instruction is a conditional jump that follows a test. It jumps to the specified location if the Zero Flag (ZF) is cleared (0). JNZ is commonly used to explicitly test for something not being equal to zero whereas JNE is commonly found after a CMP instruction.

**JMP:** the JMP instruction performs an unconditional jump. Such an instruction transfers the flow of execution by changing the instruction pointer register.

**LEA (Load Effective Address):** LEA and MOV both are same but in that there are quite difference between both of them.

* LEA means Load Effective Address
* MOV means Load Value

In short, LEA loads a pointer to the item you're addressing whereas MOV loads the actual value at that address. The purpose of LEA is to allow one to perform a non-trivial address calculation and store the result.

**LEA AX, [BP+SI+5]; Compute address of value**

**MOV AX, [BP+SI+5]; Load value at that address**

**ASSUME:** The ASSUME directive tells the assembler to assume, that a certain register contains the base of some structure (in your case: segments). In your case, CS and DS point to the code segment and the data segmentrespectively, both the one and only of their respective kind.

**DISPLAY:** For display SRTING or in here MSG.

**Code:**

DATA SEGMENT

**STR1** DB 'ENTER FIRST STRING HERE ->$'

**STR2** DB 'ENTER SECOND STRING HERE ->$'

NEWLINE DB 10,13,'$'

**MSG1** DB 'BOTH STRING ARE SAME$'

**MSG2** DB 'BOTH STRING ARE DIFFERENT$'

DATA ENDS

CODE SEGMENT

ASSUME DS:DATA,CS:CODE

START:

**MOV** AX,DATA

**MOV** DS,AX

**MOV** ES,AX

**LEA** SI,STR1

**LEA** DI,STR2

**MOV** AH,09H

**LEA** DX,STR1

**INT** 21H

**MOV** AH,0AH

**MOV** DX,SI

**INT** 21H

**MOV** AH,09H

**LEA** DX,NEWLINE

**INT** 21H

**MOV** AH,09H

**LEA** DX,STR2

**INT** 21H

**MOV** AH,0AH

**MOV** DX,DI

**INT** 21H

**MOV** AH,09H

**LEA** DX,NEWLINE

**INT** 21H

**MOV** CL,STR1+1

**MOV** BL,STR2+1

**CMP** cl,bl

**JNE** L1

**REP** CMPSB

**JNE** L1

**LEA** dx,MSG1

**JMP** L2

L1:**LEA** DX,MSG2

L2:**MOV** AH,09H

**INT** 21H

**MOV** AH,4CH

**INT** 21H

CODE ENDS

END START

**Output:**

 