EE309 Assignment 2, Sheel Shah, 19D070052.

Q1

lea ax, table[si]: the address obtained by addition of offset to the constant value of AX is placed.

mov ax, table[si]: the value present at the address obtained by addition of offset to the constant value of AX is placed.

Eg. Consider effective address 100 which has value 10H. Hence lea ax, [100] will store 100H in AX, but mov ax, [100] will store 10H in AX.

Q2

• JL/JNGE:

Jumps when SF != OF. Used to jump when first operand is strictly less than the second one.

Case 1 (operand 1 < operand 2):

If difference overflows -128, OF = 1, SF = 0. Therefore jump takes place.

If difference doesn't overflow, OF = 0, SF = 1. Therefore jump takes place.

Case 2 (operand 1 >= operand 2):

The difference is positive. Therefore OF = SF = 0 if difference <= 127 and otherwise OF = SF = 1. Hence no jump.

• JLE/JNG:

Jumps when (SF != OF) or (ZF = 1). Used to jump when first operand is less equal the second one.

The strictly less condition has already been discussed above. Hence, if strictly lesser, then first term causes jump. If not strictly lesser, but less equal (i.e. equal) the difference is 0, implying ZF = 1. Hence second condition causes jump when both operands are equal. If first operand is strictly greater, then the difference is positive and not zero. Now even if it overflows 127, SF = OF. Both zero when no overflow, and both one otherwise. Hence, no jump will take place in this case.

• JNL/JGE:

Jumps when SF = OF. Used to jump when first operand is greater equal than the second one.

Case 1 (operand 1 < operand 2):

If difference overflows -128, OF = 1, SF = 0. Therefore no jump takes place.

If difference doesn't overflow, OF = 0, SF = 1. Therefore no jump takes place.

Case 2 (operand 1 >= operand 2):

The difference is positive. Therefore OF = SF = 0 if difference <= 127 and otherwise OF = SF = 1. Hence jump takes place.

• JNLE/JG:

Jumps when (SF = OF) and (ZF = 0). Used to jump when first operand is strictly greater than the second one.

The greater equal condition has already been discussed above. We want to exclude the equal case which is done by ZF = 0 check. Since both conditions should be satisfied we want (greater equal) and (not equal) which is as good as strictly greater.

• JS:

Jumps when SF=1. Used to jump when an operation yields negative result. A negative result sets SF, hence causing the jump.

• JO:

Jumps when OF=1. Used to jump when an operation yields overflown result. An overflown result (<= -128 or >127) sets OF, hence causing the jump.

Q3.

The last two digits of my roll no. are 5 and 2. Look at the co-submitted code for the exact solution.

Results:

Subtraction by 79: 7, 3 in successive bytes

Multiplication of 7 and 3: 2, 1 in successive bytes

Division by 7: 3, 0 overwrite the 2, 1

Flags:

CF = 0, ZF = 0, SF = 0, OF = 0, PF = 0, AF = 0

Approach:

SUB for subtraction with DAS for decimal adjust with sign. The digits were separated by apt ANDing/ORing

MUL for multiplication with AAM for decimal adjust

Plain DIV is used

Q4

```
MOV [52H], AH ;save AH at 52H

MOV AH, 00 ;clr AH

MOV SI, AX ;move AX to SI

CALL TABLE[SI] ;call function
```