HW#3

Q1

1. MOV AL, I+1
   valid.
2. MOV [SI], I
   invalid, there is ambiguity whether to move the immediate value I to a byte or to a word.
3. MOV AX, [BL]
   Invalid, BL cannot be used as a base register
4. MOV AX, J+2
   invalid, operand J+2 is of type byte while AX is of type word.
5. MOV BX, 2*J
   invalid, J is a variable. Such operations are allowed only on constants
6. MOV BL, K+6
   invalid, K+6 = 256 which cannot fit in a byte.
7. MOV L, I
   valid.
8. MOV DS, I
   invalid, an immediate value cannot be moved directly to a segment register
9. SUB AX, DS
   invalid, a segment register cannot be used in subtraction.
10. ADD AX, J+2[BX]
    invalid, the source operand is of byte type while the destination operand is of word type.
11. MOV [SI], word PTR [DI-1]
    invalid, both operands cannot be memory operands.
12. INC [DI+1]
    invalid, there is ambiguity whether to increment a byte or a word.

13. MOV DS, ES
invalid, a segment register cannot be moved directly to another segment register

14. INC L+1
valid

15. DEC Byte PTR [SI+DI]
invalid, both index registers cannot be used simultaneously.

16. ADD Byte PTR [BX], 2*I+1
valid.

17. SUB AH, [BX-SI-2]
invalid, in the based-indexed addressing mode, operation between registers cannot be subtraction.

18. IMUL K
invalid, the operand of IMUL cannot be constant.

19. SUB CX, [AX]
invalid, AX is not allowed to be used for register indirect addressing.

20. ADC CX, [BP]2[SI]
invalid, in based-indexed addressing mode, the immediate value has to be specified first in this format -

Q2 1. ADC BX, CX
BX = 7FED + F1A4 + 1 = 7192
OF=0, SF=0, ZF=0, AF=1, PF=0, CF=1

2. INC Byte PTR [DI-16]

[0010] ← [0010] + 1 = FF + 1 = 00

OF=0, SF=0, ZF=1, AF=1, PF=1, CF=1 (unchanged)

*Note that 16 here is in decimal.

3. SBB BL, AL

BL = ED - 14 - 1 = D8

OF=0, SF=1, ZF=0, AF=0, PF=1, CF=0

4. SUB AL, 2+[SI]

AL = 14 - [0012] = 14 - BC = 58

OF=0, SF=0, ZF=0, AF=1, PF=0, CF=1

5. DEC Byte PTR 4[SI]

[0014] ← [0014] - 1 = FE - 1 = FD

OF=0, SF=1, ZF=0, AF=0, PF=0, CF=1 (unchanged)

6. NEG word PTR [BX-7FDCh]

[0012:0011] = 0 - BC1A = 43E6

OF=0, SF=0, ZF=0, AF=1, PF=0, CF=1

7. MUL DL

AX ← AL * DL = 14h * FFh = 13EC

OF=CF=1, other flags are undefined

8. IMUL DL

AX ← AL * DL (signed) = 14h * FFh = FFEC

OF=CF=0, other flags are undefined

9. DIV Byte PTR [DI-13]

AX ← AX / [0013] = FE14 / 6 = $(65044)_{10}$ / 6

This division results in a quotient = $(10840)_{10}$ and a remainder of 4. However, since the quotient is very large and cannot fit in AL, the CPU will generate an interrupt. All the flags are undefined.

10. IDIV CH

AX ← AX / CH = FE14 / F1 = (-492) / (-15)

AH = F4 , AL = 20

The quotient is $(32)_{10}$ and the remainder is $(-12)_{10}$. All the flags are undefined.

Q3

1. SUB Word PTR [EFA1], 0FFh
2. ADD BX, [BX]
3. SBB AH, CL
4. NEG Byte PTR [BX]
5. MUL BX or IMUL BX
6. MOV BX, [DI+BX-7]

Q4

```
.Model Small
.stack 100
.Data
   i  db ?
   j  db ?
   K  dw ?
   L  db ?
.CODE
.STARTUP
   Mov i, -4
   Mov j, 30
   Mov AL, 4
   IMUL i          ;  AX = 4*i
   IMUL j          ;  AX = 4*i*j
   Mov K, AX       ;  K = 4*i*j
   Mov AL, 5
   MUL j           ;  AX = 5*j

   Mov BX, AX      ;  BX = 5*j
   Mov AL, i       ;  AL = i
   CBW             ;  AX = i
   Add BX, AX      ;  BX = 5*j+i
   Add BX, 1       ;  BX = 5*j+i+1
   Add K, BX       ;  K = (4*i*j)+(5*j+i)+1
   Mov AX, K
   IDIV i          ;  AX = K/i
   Mov L, AL
   INC i
   Mov AL, j
   SUB AL, i
   DEC AL
   Mov j, AL
.EXIT
END
```