**Addition of 2 16 bit numbers**

MOV SI,5000 //starting address of stored data

MOV DI,6000 //destination address

Mov AX,[SI]

INC SI

INC SI

MOV BX,[SI]

ADD AX,BX

MOV [DI],AX

INT 03 //Display using Interrupt

**Subtraction of 2 32 bit numbers**

MOV SI,5000

MOV DI,6000

MOV AX,[SI]

INC SI

INC SI

MOV BX,[SI]

SUB AX,BX

MOV [DI],AX

INT 03

**MULTIPLICATION OF 16 BIT NUMBERS**

MOV SI,5000

MOV DI,6000

MOV AX,[SI]

INC SI

INC SI

MOV BX,[SI]

MUL BX

MOV [DI],AX //LOWER 16 BIT

INC DI

INC DI

MOV [DI],DX // UPPER 16 BIT

INT 03

**DIVISION OF 2 16 BIT NUMBERS**

MOV SI,5000

MOV DI,6000

MOV AX,[SI]

INC SI

INC SI

MOV BX,[SI]

XOR DX,DX //CLEARING DX

DIV BX

MOV [DI],AX

INC DI

INC DI

MOV [DI],DX

INT 03

**Array sorting of 16 bit numbers**

STORE THE NUMBER OF NUMBEERS INTO 5000

MOV SI,5000

MOV CL,[SI]

DEC CL

Outer loop🡪 MOV SI,5000

MOV CH,[SI]

DEC CH

INC SI

Inner loop-🡪 MOV AX,[SI]

INC SI

INC SI

CMP AX,[SI]

JC loop SKIPPING

XCHG AX,[SI]

DEC SI

DEC SI

XCHG AX,[SI]

INC SI

INC SI

LOOP SKIPPING🡪 DEC CH

JNZ //address of inner loop

DEC CL

JNZ //address of outer loop

INT 03

**SEARCHING OF 16 BIT NUMBERS IN AN ARRAY**

MOV SI,5000 //LENGTH STORED AT THIS ADDRESS

MOV CL,[SI]

MOV DI,6000 //KEY TO BE SEARCHED

MOV BX,[DI]

INC DI

INC DI

INC SI

INC SI

LOOP🡪 MOV AX,[SI]

CMP AX,BX

JE FOUND

INC SI

INC SI

DEC CL

JNZ LOOP

JMP NOT FOUND

FOUND: MOV DX,0FFFF

JMP STORE RESULT

NOT FOUND : MOV DX,0000

STORE RESULT: MOV [DI],DX

INT 03