

ACTIVITY 1: Give the value of the zero flag, the carry flag, the sign flag, and the overflow flag after each of the following instructions:

	ZF	CF	SF	OF
mov ax, 0x1254	-	-	-	-
mov bx, 0x0FFF	-	-	-	-
add ax, 0xEDAB	0	0	1	0
add ax, bx	0	1	0	0
add bx, 0xF001	1	1	0	0

ACTIVITY 2: Write a program which calculates the square of a number in memory variable. Display the result in accumulator (AX).

DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD

AX 0064 SI 0000 CS 19F5 IP 0117 Stack +0 0000 Flags 7244
 BX 000A DI 0000 DS 19F5
 CX 0000 BP 0000 ES 19F5 HS 19F5
 DX 0000 SP FFFE SS 19F5 FS 19F5

OF DF IF SF ZF AF PF CF
 0 0 1 0 1 0 1 0

S or SI or SYM

CMD >S

	0	1	2	3	4	5	6	7
0115 75F6 JNZ 010D	DS:0000	CD	20	FF	9F	00	EA	F0
0117 B8004C MOV AX,4C00	DS:0008	AD	DE	1B	05	C5	06	00
011A CD21 INT 21	DS:0010	18	01	10	01	18	01	92
011C 46 INC SI	DS:0018	01	01	01	00	02	FF	FF
011D E6C7 OUT [C7],AL	DS:0020	FF	FF	FF	FF	FF	FF	FF
011F 46 INC SI	DS:0028	FF	FF	FF	FF	EB	19	C0
0120 F60000 TEST [BX+SI],00	DS:0030	A2	01	14	00	18	00	F5
0123 8B46F6 MOV AX,[BP-0A]	DS:0038	FF	FF	FF	FF	00	00	00
0126 D1E0 SHL AX,1	DS:0040	05	00	00	00	00	00	00
	DS:0048	00	00	00	00	00	00	00

2

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
DS:0000	CD	20	FF	9F	00	EA	F0	FE	AD	DE	1B	05	C5	06	00	00
DS:0010	18	01	10	01	18	01	92	01	01	01	00	02	FF	FF	FF	FF
DS:0020	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	FF	EB	19	C0	11
DS:0030	A2	01	14	00	18	00	F5	19	FF	FF	FF	FF	00	00	00	00
DS:0040	05	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00

1 Step 2ProcStep 3Retrieve 4Help ON 5BRK Menu 6 7 up 8 dn 9 le 10 ri

[org 0x0100]

jmp start

num dw 10

start:

mov bx, [num]

mov cx, [num]

l1:

add ax, [num]

sub cx, 1

jnz l1

mov ax, 0x4c00

int 0x21

ACTIVITY 3: Write a program which finds the frequency of a specific number from the given array. array: dw 1, 9, 9,9, 8, 8,8, 8, 8,8, 1, 1, 9, 9, 8, 8, 8, 8, 1, 9, 8, 8

The screenshot shows the DOSBox 0.74 interface. At the top, it displays 'DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD'. Below this, the register state is shown: AX 0004, SI 002C, CS 19F5, IP 0156, Stack +0 0000, Flags 7200; BX 0008, DI 0000, DS 19F5; CX 002C, BP 0000, ES 19F5, HS 19F5; DX 0000, SP FFFE, SS 19F5, FS 19F5. The status bar shows 'S or SI or SYM'. The command prompt shows 'CMD >S'. The assembly code is displayed in a table with columns for address, instruction, and comment. The memory dump shows the contents of the DS segment starting at address 0000.

Address	Instruction	Comment
0152	81C60200	ADD SI,0002
0156	EBEC	JMP 0144
0158	B8004C	MOV AX,4C00
015B	CD21	INT 21
015D	8946E2	MOV [BP-1E],AX
0160	8B46F6	MOV AX,[BP-0A]
0163	D1E0	SHL AX,1
0165	D1E0	SHL AX,1
0167	C5ED8	LDS BX,[BP-28]

Address	0	1	2	3	4	5	6	7
DS:0000	CD	20	FF	9F	00	EA	F0	FE
DS:0008	AD	DE	1B	05	C5	06	00	00
DS:0010	18	01	10	01	18	01	92	01
DS:0018	01	01	01	00	FF	00	01	FF
DS:0020	FF	FF	FF	FF	FF	FF	FF	FF
DS:0028	FF	FF	FF	FF	EB	19	C0	11
DS:0030	A2	01	14	00	18	00	F5	19
DS:0038	FF	FF	FF	FF	00	00	00	00
DS:0040	05	00	00	00	00	00	00	00
DS:0048	00	00	00	00	00	00	00	00

[org 0x0100]

jmp start

array: dw 1, 9, 9,9, 8, 8,8, 8, 8,8, 1, 1, 9, 9, 8, 8, 8, 8, 1, 9, 8, 8
find dw 1

incr:

```
add ax, 1
add si, 2
jmp l1
```

start:

```
mov cx, 44
mov si, 0
mov ax, 0
```

l1:

```
cmp cx, si
je term
mov bx, [array+si]
```

```
    cmp bx, [find]
    je incr
    add si, 2
    jmp l1
```

term:

```
mov ax, 0x4c00
int 0x21
```

```

DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD
AX 0078 SI 0000 CS 19F5 IP 0127 Stack +0 0000 Flags 7244
BX 0078 DI 0000 DS 19F5          +2 20CD
CX 0000 BP 0000 ES 19F5 HS 19F5   +4 9FFF OF DF IF SF ZF AF PF CF
DX 0078 SP FFFE SS 19F5 FS 19F5   +6 EA00  0 0 1 0 1 0 1 0

S or SI or SYM
CMD >S

0123 81E90100 SUB CX,0001
0127 EBE3 JMP 010C
0129 B8004C MOV AX,4C00
012C CD21 INT 21
012E C3 RET
012F 8B07 MOV AX,[BX]
0131 8B5702 MOV DX,[BX+02]
0134 85D2 TEST DX,DX
0136 7504 JNZ 013C

2 0 1 2 3 4 5 6 7 8 9 A B C D E F
DS:0000 CD 20 FF 9F 00 EA FF FF AD DE 1B 05 C5 06 00 00 = f.n i |.+. ...
DS:0010 18 01 10 01 18 01 92 01 01 01 01 00 FF 00 01 FF .....f. ....
DS:0020 FF FF FF FF FF FF FF FF FF FF FF FF EB 19 E6 11 .....δ.μ.
DS:0030 A2 01 14 00 18 00 F5 19 FF FF FF FF 00 00 00 00 6.....J. ....
DS:0040 05 00 00 00 00 00 00 00 00 00 00 00 00 00 00 .....
1 Step 2ProcStep 3Retrieve 4Help ON 5BRK Menu 6 7 up 8 dn 9 le 10 ri

```

```
jmp start
num dw 5 ; Factorial should be 120 (0078h)
```

```
mov ax, 1
mov cx, [num]
```

```
cmp cx, 0
je term
mov bx, ax
mov dx, 0
mov si, cx
```

```
add dx, bx
sub si, 1
jnz Adding
```

```
    mov ax, dx
    sub cx, 1
    jmp Mult
```

term:

```
mov ax, 0x4c00
int 0x21
```

ACTIVITY 5: Write a program which determines smallest number from the given array. array:
dw 111, 999, 888, 888, 11, 99, 88, 88, 1, 9, 8, 8

DOSBox 0.74, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD

AX 0008	SI 0016	CS 19F5	IP 014D	Stack +0 0000	Flags 7244
BX 0008	DI 0000	DS 19F5		+2 20CD	
CX 0052	BP 0000	ES 19F5	HS 19F5	+4 9FFF	OF DF IF SF ZF AF PF CF
DX 0001	SP FFFE	SS 19F5	FS 19F5	+6 EA00	0 0 1 0 1 0 1 0

CMD >S

0141 740A	JZ	014D
014D B8004C	MOV	AX, 4C00
0150 CD21	INT	21
0152 08B80100	OR	[0001+BX+SI], BH
0156 EB05	JMP	015D
0158 E94201	JMP	029D
015B 31C0	XOR	AX, AX
015D 8946E2	MOV	[BP-1E], AX
0160 8B46F6	MOV	AX, [BP-0A]

DS:0000	CD 20 FF 9F 00 EA F0 FE	AD DE 1B 05 C5 06 00 00
DS:0010	18 01 10 01 18 01 92 01	01 01 01 00 FF 00 01 FF
DS:0020	FF FF FF FF FF FF FF FF	FF FF FF FF EB 19 C0 11
DS:0030	A2 01 14 00 18 00 F5 19	FF FF FF FF 00 00 00 00
DS:0040	05 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00

1 Step 2ProcStep 3Retrieve 4Help ON 5BRK Menu 6 7 up 8 dn 9 le 10 ri

[org 0x0100]

jmp start

array: dw 111, 999, 888, 888, 11, 99, 88, 88, 1, 9, 8, 8

start:

```
mov ax, [array]
mov dx, ax
mov bx, [array+2]
mov si, 0
```

I11:

```
mov dx, ax
jmp I2
```

I12:

```
mov dx, bx
jmp I2
```

comp:

```
cmp ax, dx
jl I11
cmp bx, dx
```

```
        jl l12
l2:      add si, 2
        cmp si, 22
        je term
        mov ax, [array+si]
        mov bx, [array+si+2]
        jmp comp
term:
```

```
mov ax, 0x4c00
int 0x21
```