

Question No. 1

➤ Code Screen Shot

```

1  [org 0x0100]
2
3  mov ax , 0
4  mov cx , 20
5
6  loop1: add ax , 20
7         dec cx
8
9         jnz loop1
10
11 mov ax , 0x4c00
12 int 0x21
13

```

➤ Final Answer In Debugger

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD

AX 0190 SI 0000 CS 19F5 IP 010A Stack +0 0000 Flags 7244
 BX 0000 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 0000 SP FFFE SS 19F5 FS 19F5 +6 EA00 0 0 1 0 1 0 1 0

CMD >

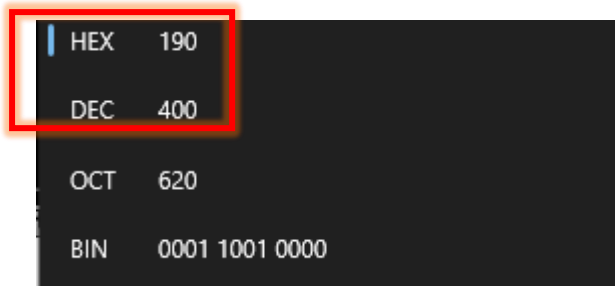
Address	Disassembly	Comment
0109 49	DEC CX	
010A 75FA	JNZ 0106	
010C B8004C	MOV AX, 4C00	
010F CD21	INT 21	
0111 DAEB	ESC 15, BL	
0113 0431	ADD AL, 31	
0115 D2	DB D2	
0116 31C0	XOR AX, AX	
0118 8956E4	MOV [BP-1C], DX	

Memory Dump (DS:0000):

Address	Hex	Hex	Hex	Hex	Hex	Hex	Hex
DS:0000	CD	20	FF	9F	00	EA	F0
DS:0008	AD	DE	1B	05	C5	06	00
DS:0010	18	01	10	01	18	01	92
DS:0018	01	01	01	00	02	FF	FF
DS:0020	FF	FF	FF	FF	FF	FF	FF
DS:0028	FF	FF	FF	FF	EB	19	C0
DS:0030	A2	01	14	00	18	00	F5
DS:0038	FF	FF	FF	FF	00	00	00
DS:0040	05	00	00	00	00	00	00
DS:0048	00	00	00	00	00	00	00

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

➤ **Checking The Answer with Calculator**



HEX	190
DEC	400
OCT	620
BIN	0001 1001 0000

Which Is the Square Of 20 And Got It by Adding 20 20 times in AX

Question No. 2

➤ Code Screen Shot

```
1  [org 0x0100]
2
3  jmp start
4
5  array : dw 7 , 14 , 21 , 28 , 35 , 42 , 49 , 6 , 13 , 20
6  even1 : dw 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0
7  odd1  : dw 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0 , 0
8
9  start:
10     mov ax , 1
11     mov si , 0
12     mov bx , 0
13     mov bp , 0
14
15     loop1: mov ax , [array + si]
16             shr ax , 1
17
18             jnc evenInstructions
19
20             mov cx , [array + si]
21             mov [odd1 + bp] , cx
22             add bp , 2
23             jmp endOfLoop
24
25     evenInstructions:
26
27             mov cx , [array + si]
28             mov [even1 + bx] , cx
29             add bx , 2
30
31             jmp endOfLoop
32
33     endOfLoop
34         add si , 2
35         cmp si , 20
36     jnz loop1
37
38     mov ax , 0x4c00
39     int 0x21
40
```

➤ Final Answers in Debugger

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD

AX 000A SI 0014 CS 19F5 IP 017F Stack +0 0000 Flags 7244
 BX 000A DI 0000 DS 19F5 +2 20CD
 CX 0014 BP 000A ES 19F5 HS 19F5 +4 9FFF OF DF IF SF ZF AF PF CF
 DX 0000 SP FFFE SS 19F5 FS 19F5 +6 EA00 0 0 1 0 1 0 1 0

CMD >

Address	Disassembly	Comment
017D	75D0	JNZ 014F
017F	B8004C	MOV AX,4C00
0182	CD21	INT 21
0184	8B04	MOV AX,[SI]
0186	8B5C02	MOV BX,[SI+02]
0189	89C6	MOV SI,AX
018B	895EFE	MOV [BP-02],BX
018E	837EFE00	CMP [BP-02],0000
0192	7504	JNZ 0198

DS:0117 0E 00 1C 00 2A 00 06 00
 DS:011F 14 00 00 00 00 00 00 00
 DS:0127 00 00 00 00 00 00 00 07 00
 DS:012F 15 00 23 00 31 00 0D 00
 DS:0137 00 00 00 00 00 00 00 00
 DS:013F 00 00 00 00 00 00 00 00
 DS:0147 00 00 00 00 00 00 00 00
 DS:014F 8B 84 03 01 D1 E8 73 0F
 DS:0157 8B 8C 03 01 89 8E 2D 01
 DS:015F 81 C5 02 00 E9 0F 8B

DS:012D 07 00 15 00 23 00 31 00
 DS:013D 00 00 00 00 00 00 00 00
 DS:014D 00 00 00 00 00 00 00 00
 DS:015D 2D 01 81 C5 02 00 E9 0F
 DS:016D 01 81 C3 02 00 E9 00 81

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

➤ Explanation

I've done Question 2 in a manner that I'm accessing each number in the array and then performing right shift that causes the most right bit of that number to fall into the carry. And as we know the right most bit decides whether the number is odd or even. If the carry due to the right shift is 1 then it means that the number is odd or if the carry is 0 then that means that it is even and then I'm finally storing them into their respective arrays as named in the code. This process happens until all the numbers in the given array are checked and distributed into the even and odd arrays.