FAST-NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES PESHAWAR CAMPUS

NAME: Junaid saeed

ROLL NO: 22P-9200

SECTION: BS(CS)-3D

COURSE: COAL

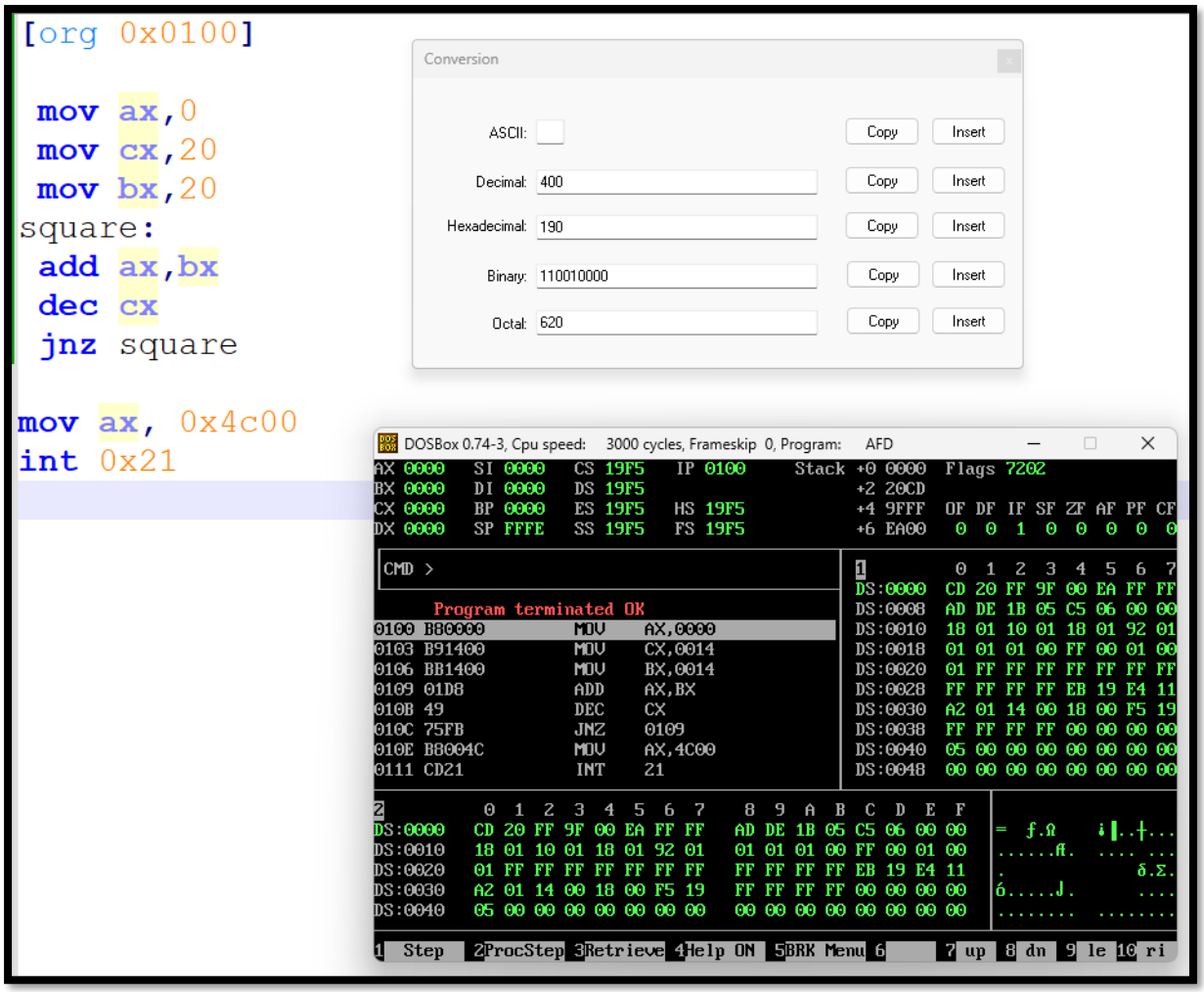
ASSIGNMENT 2

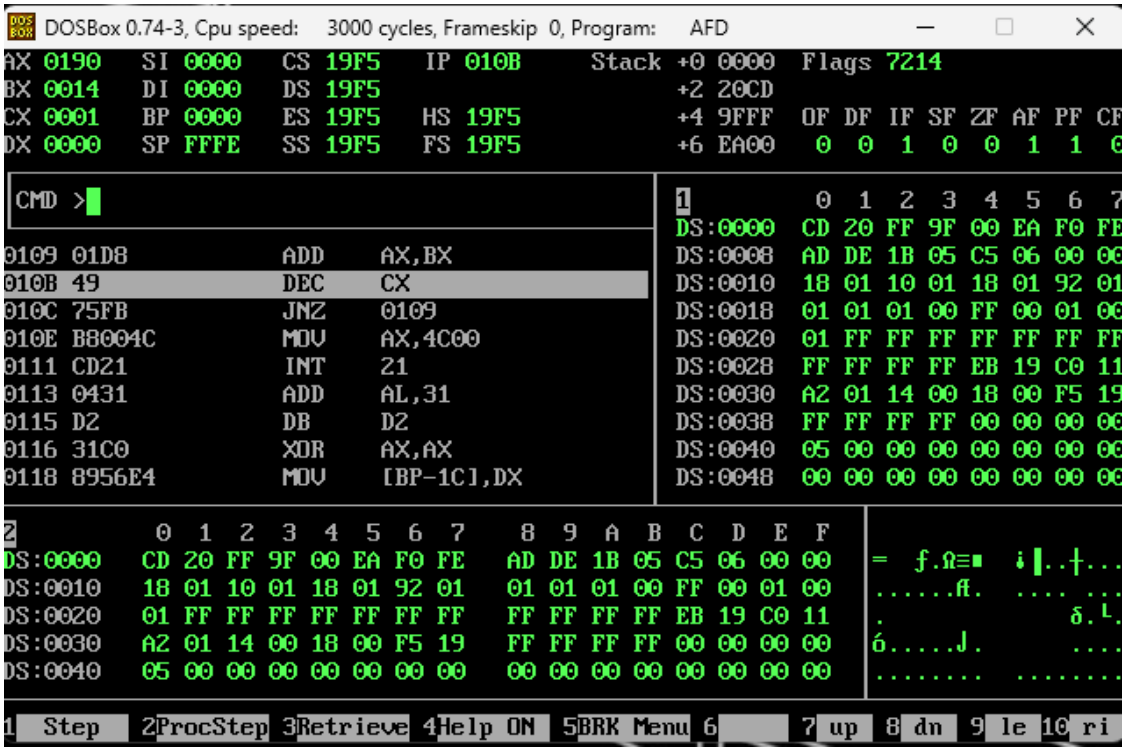
SUBMITTED TO: Sir Usman Abassi

QUESTION # 1

```
[org 0x0100]

 mov ax,0
 mov cx,20
 mov bx,20
square:
 add ax,bx
 dec cx
 jnz square

mov ax, 0x4c00
int 0x21
```

```
[org 0x0100]

 mov ax,0
 mov cx,20
 mov bx,20
square:
 add ax,bx
 dec cx
 jnz square

mov ax, 0x4c00
int 0x21
```





Question#2

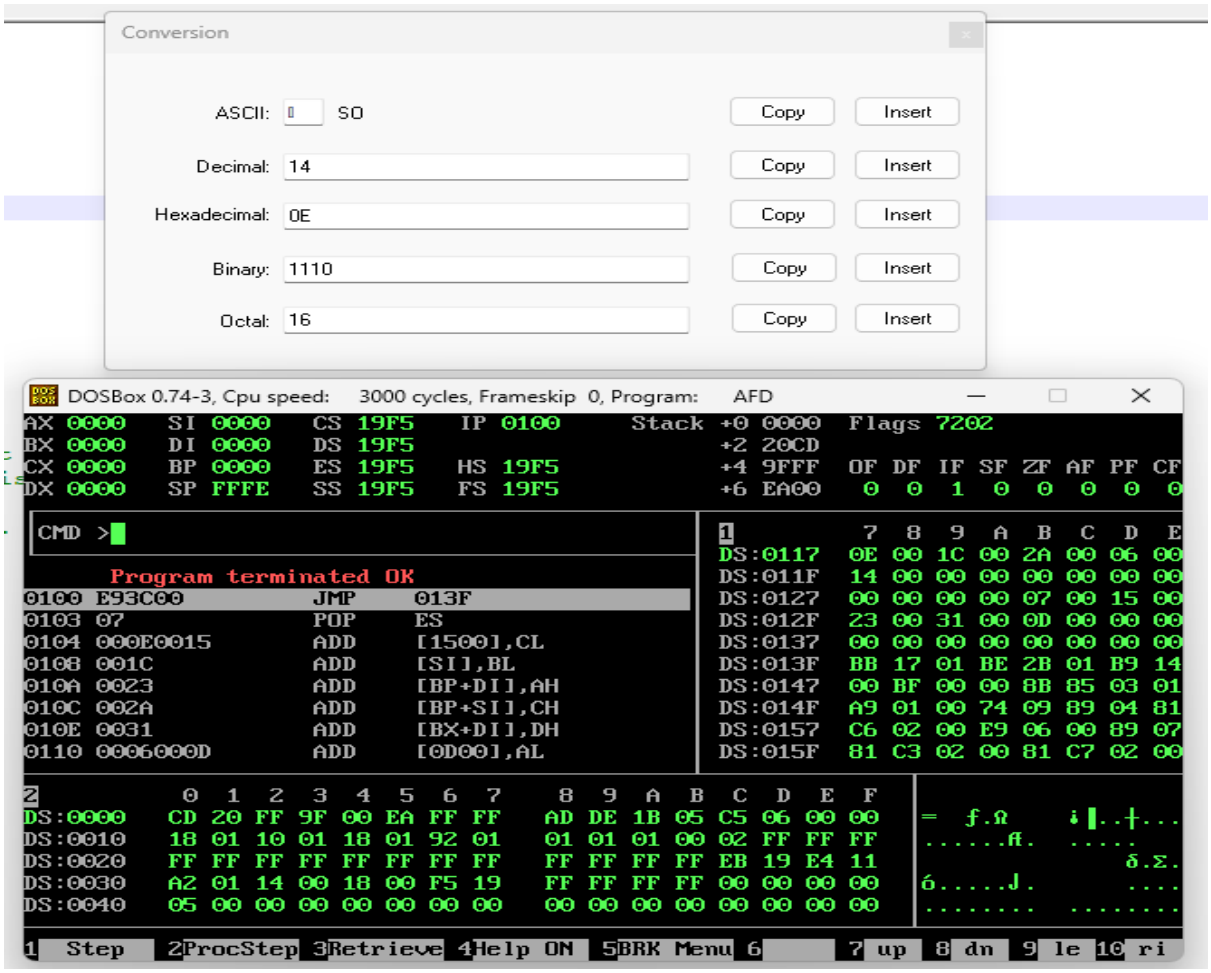
```
[org 0x0100]                                                    ; Explanation in the code
jmp start

array:dw 7, 14, 21, 28, 35, 42, 49, 6, 13, 20
evenn:dw 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
odd  :dw 0, 0, 0, 0, 0, 0, 0, 0, 0, 0

start:

 mov bx, evenn
 mov si, odd
 mov cx, 20
 mov di, 0

check:

 mov ax, [array + di]
 test ax, 1                  ; checking the least significant bit .
 jz even1                ; Jump to even label if the number is even

 mov [si], ax              ; Storing odd number in odd array.
 add si, 2
 jmp next

even1:

 mov [bx], ax       ;storing even number in even array
 add bx, 2

next:

  add di, 2
  cmp di,cx
  jnz check

    mov ax, 0x4c00
    int 0x21
```

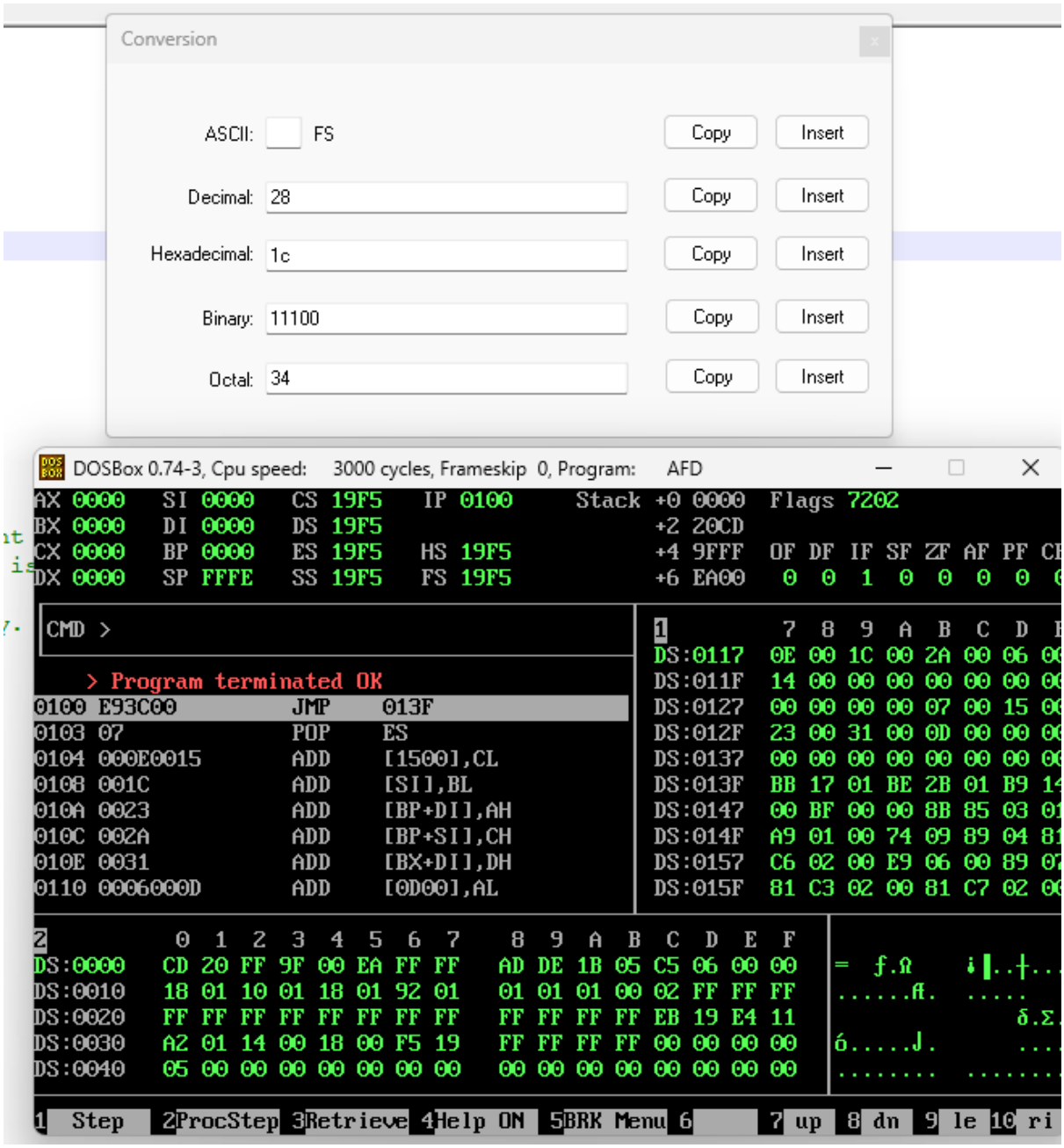
Explanation :

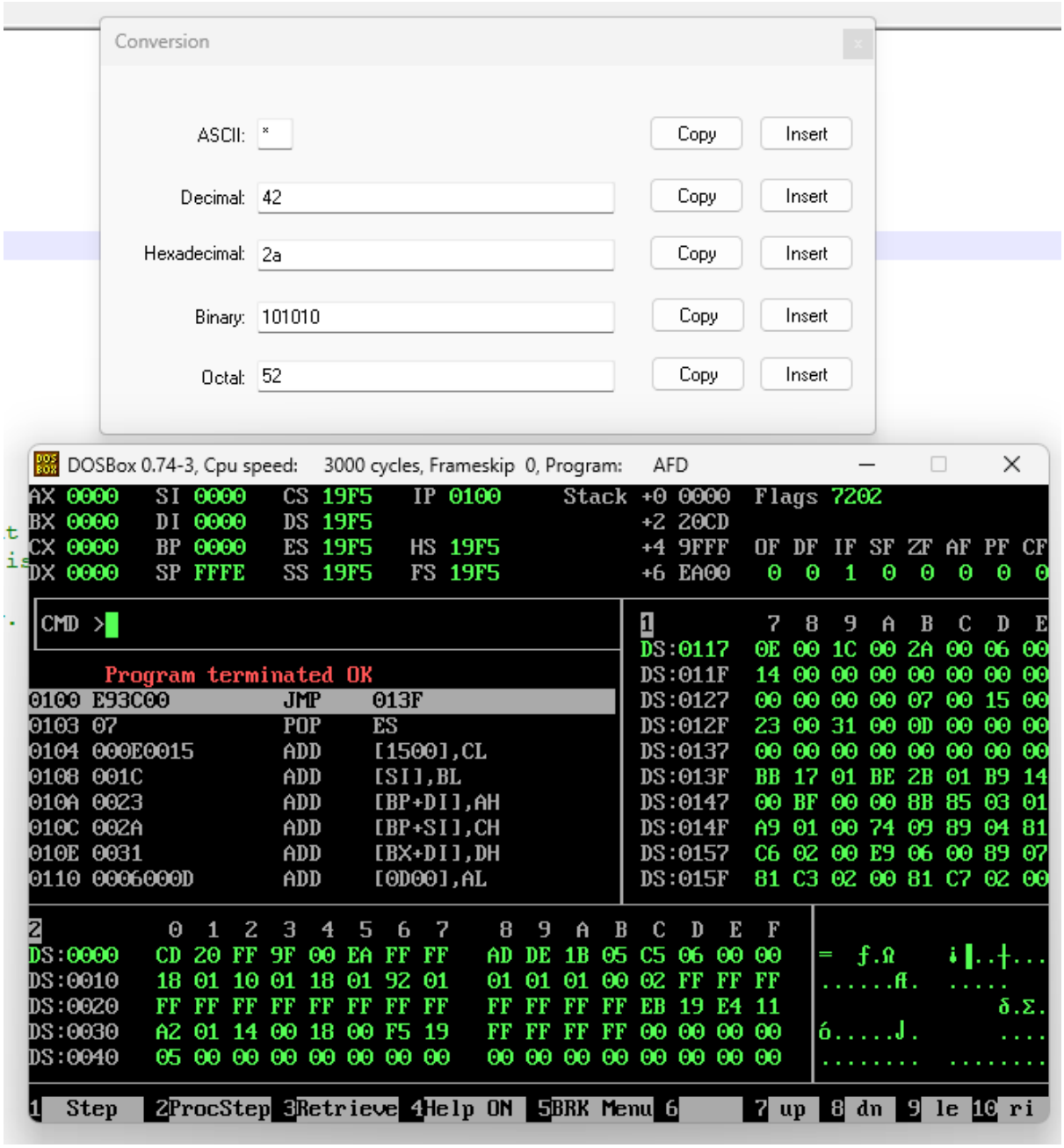
In this code to find out the even and odd from the array of numbers .first of all I am storing 0's in evenn and odd array that will be later updated by the even and odd numbers . so to find this in used test wich will check the least significant bit and it will jump accordingly to the even and odd ones.

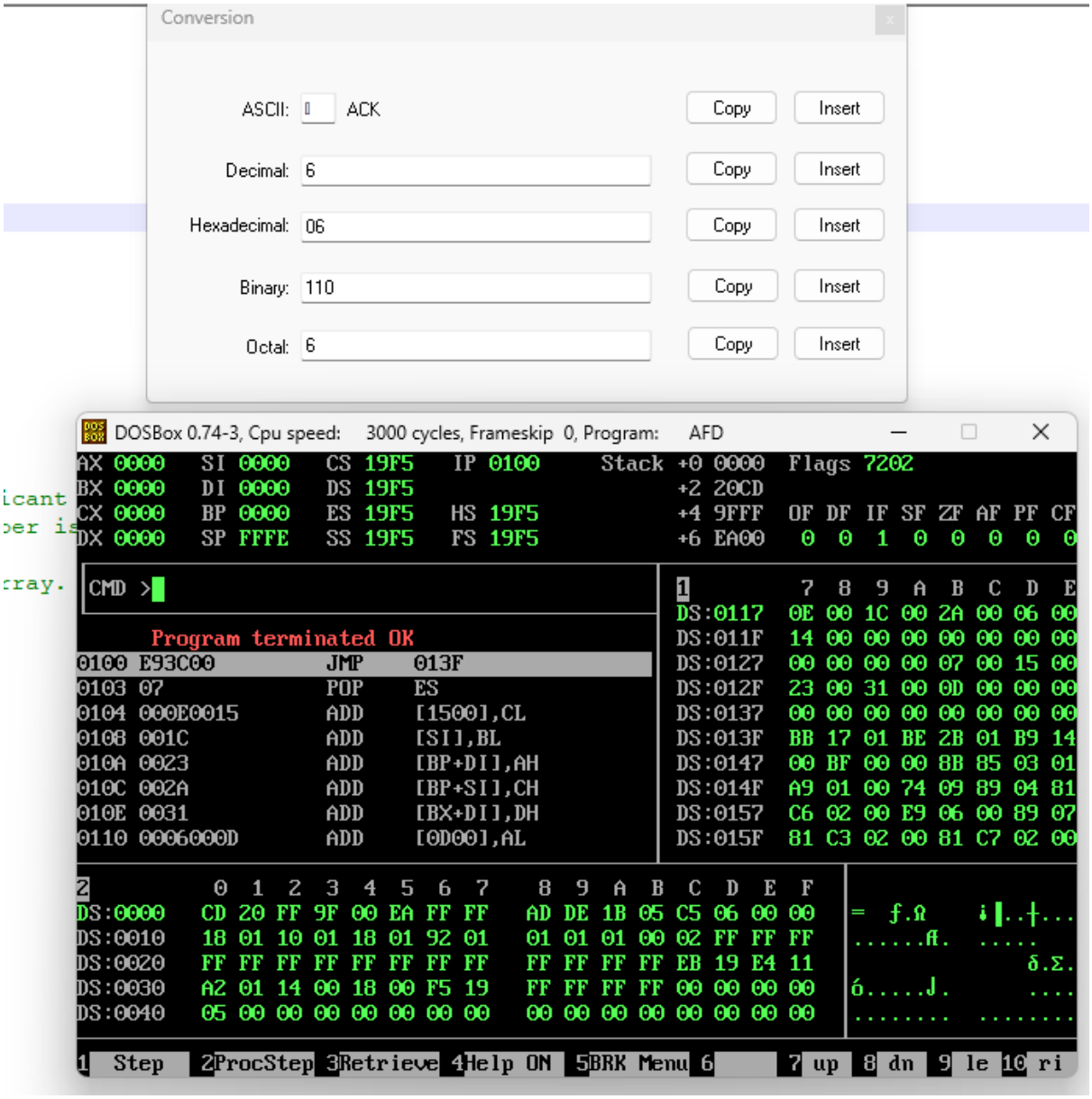
```
[org 0x0100]
jmp start

array:dw 8, 15, 22, 29, 36, 43, 50, 7, 14, 21
evenn:dw 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
odd  :dw 0, 0, 0, 0, 0, 0, 0, 0, 0, 0

start:

 mov bx, evenn
 mov si, odd
 mov cx, 20
 mov di, 0

check:

 mov ax, [array + di]
 test ax, 1                ; checking the least significant bit .
 jz even1                ; Jump to even label if the number is even

 mov [si], ax              ; Storing odd number in odd array.
 add si, 2
 jmp next

even1:

 mov [bx], ax       ;storing even number in even array
 add bx, 2

next:

  add di, 2
  cmp di,cx
  jnz check

    mov ax, 0x4c00
    int 0x21
```

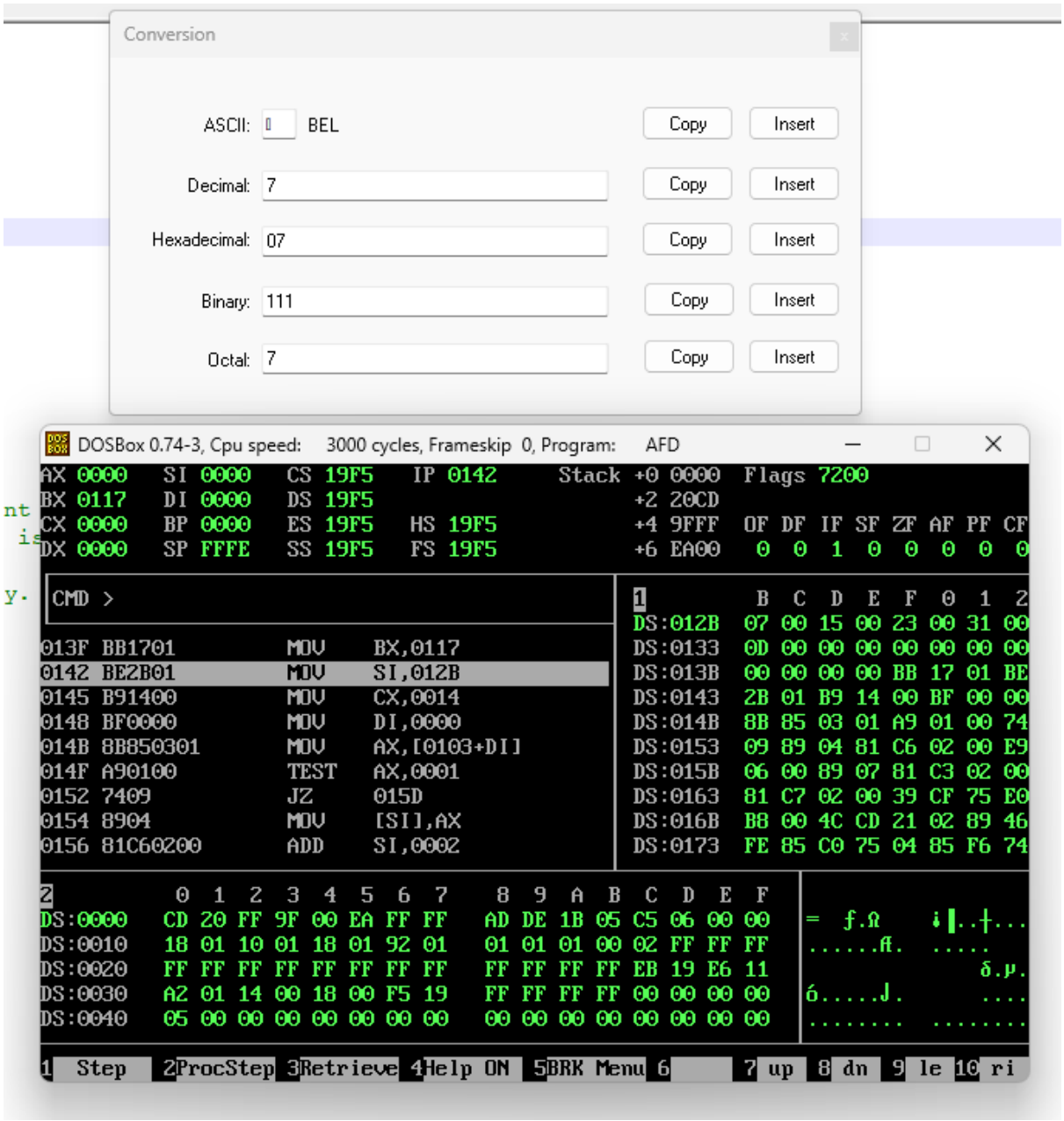
```
DOSBox 0.74-3, Cpu speed:   3000 cycles, Frameskip  0, Program:   AFD
AX 0000   SI 0000   CS 19F5   IP 0100     Stack +0 0000  Flags 7202
BX 0000   DI 0000   DS 19F5               +2 20CD
CX 0000   BP 0000   ES 19F5   HS 19F5      +4 9FFF  OF DF IF SF ZF AF PF CF
DX 0000   SP FFFE   SS 19F5   FS 19F5      +6 EA00   0  0  1  0  0  0  0  0

CMD >

   > Program terminated OK
0100 E93C00          JMP     013F
0103 0800            OR      [BX+SI],AL
0105 0F0016001D      LLDT    W/[1D00]
010A 0024            ADD     [SI],AH
010C 002B            ADD     [BP+DI],CH
010E 0032            ADD     [BP+SI],DH
0110 0007            ADD     [BX],AL
0112 000E0015        ADD     [1500],CL

1          0  1  2  3  4  5  6  7
DS:0000   CD 20 FF 9F 00 EA FF FF
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 02 FF FF FF
DS:0020   FF FF FF FF FF FF FF FF
DS:0028   FF FF FF FF EB 19 E4 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

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
DS:0010   18 01 10 01 18 01 92 01   01 01 01 00 02 FF FF FF
DS:0020   FF FF FF FF FF FF FF FF   FF FF FF FF EB 19 E4 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
```


Conversion
ASCII: SO
Copy
Insert
Decimal: 14
Copy
Insert
Hexadecimal: 0E
Copy
Insert
Binary: 1110
Copy
Insert
Octal: 16
Copy
Insert
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD
AX 0000 SI 0000 CS 19F5 IP 0100 Stack +0 0000 Flags 7202
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 0 0 0 0
CMD >
Program terminated OK
0100 E93C00 JMP 013F
0103 07 POP ES
0104 000E0015 ADD [1500],CL
0108 001C ADD [SI],BL
010A 0023 ADD [BP+DI],AH
010C 002A ADD [BP+SI],CH
010E 0031 ADD [BX+DI],DH
0110 0006000D ADD [0D00],AL
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 07 00 15 00
DS:012F 23 00 31 00 0D 00 00 00
DS:0137 00 00 00 00 00 00 00 00
DS:013F BB 17 01 BE 2B 01 B9 14
DS:0147 00 BF 00 00 8B 85 03 01
DS:014F A9 01 00 74 09 89 04 81
DS:0157 C6 02 00 E9 06 00 89 07
DS:015F 81 C3 02 00 81 C7 02 00
DS:0000 CD 20 FF 9F 00 EA FF FF AD DE 1B 05 C5 06 00 00
DS:0010 18 01 10 01 18 01 92 01 01 01 01 00 02 FF FF FF
DS:0020 FF FF FF FF FF FF FF FF FF FF FF FF EB 19 E4 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


Conversion
ASCII: FS
Copy
Insert
Decimal: 28
Copy
Insert
Hexadecimal: 1c
Copy
Insert
Binary: 11100
Copy
Insert
Octal: 34
Copy
Insert
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD
AX 0000 SI 0000 CS 19F5 IP 0100 Stack +0 0000 Flags 7202
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 0 0 0 0
CMD >
> Program terminated OK
0100 E93C00 JMP 013F
0103 07 POP ES
0104 000E0015 ADD [1500],CL
0108 001C ADD [SI],BL
010A 0023 ADD [BP+DI],AH
010C 002A ADD [BP+SI],CH
010E 0031 ADD [BX+DI],DH
0110 0006000D ADD [0D00],AL
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 07 00 15 00
DS:012F 23 00 31 00 0D 00 00 00
DS:0137 00 00 00 00 00 00 00 00
DS:013F BB 17 01 BE 2B 01 B9 14
DS:0147 00 BF 00 00 8B 85 03 01
DS:014F A9 01 00 74 09 89 04 81
DS:0157 C6 02 00 E9 06 00 89 07
DS:015F 81 C3 02 00 81 C7 02 00
DS:0000 CD 20 FF 9F 00 EA FF FF AD DE 1B 05 C5 06 00 00
DS:0010 18 01 10 01 18 01 92 01 01 01 01 00 02 FF FF FF
DS:0020 FF FF FF FF FF FF FF FF FF FF FF FF EB 19 E4 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


Conversion
ASCII: *
Copy
Insert
Decimal: 42
Copy
Insert
Hexadecimal: 2a
Copy
Insert
Binary: 101010
Copy
Insert
Octal: 52
Copy
Insert
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD
AX 0000 SI 0000 CS 19F5 IP 0100 Stack +0 0000 Flags 7202
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 0 0 0 0
CMD >
Program terminated OK
0100 E93C00 JMP 013F
0103 07 POP ES
0104 000E0015 ADD [1500],CL
0108 001C ADD [SI],BL
010A 0023 ADD [BP+DI],AH
010C 002A ADD [BP+SI],CH
010E 0031 ADD [BX+DI],DH
0110 0006000D ADD [0D00],AL
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 07 00 15 00
DS:012F 23 00 31 00 0D 00 00 00
DS:0137 00 00 00 00 00 00 00 00
DS:013F BB 17 01 BE 2B 01 B9 14
DS:0147 00 BF 00 00 8B 85 03 01
DS:014F A9 01 00 74 09 89 04 81
DS:0157 C6 02 00 E9 06 00 89 07
DS:015F 81 C3 02 00 81 C7 02 00
DS:0000 CD 20 FF 9F 00 EA FF FF AD DE 1B 05 C5 06 00 00
DS:0010 18 01 10 01 18 01 92 01 01 01 01 00 02 FF FF FF
DS:0020 FF FF FF FF FF FF FF FF FF FF FF FF EB 19 E4 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


Conversion
ASCII: ACK
Copy
Insert
Decimal: 6
Copy
Insert
Hexadecimal: 06
Copy
Insert
Binary: 110
Copy
Insert
Octal: 6
Copy
Insert
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD
AX 0000 SI 0000 CS 19F5 IP 0100 Stack +0 0000 Flags 7202
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 0 0 0 0
CMD >
Program terminated OK
0100 E93C00 JMP 013F
0103 07 POP ES
0104 000E0015 ADD [1500],CL
0108 001C ADD [SI],BL
010A 0023 ADD [BP+DI],AH
010C 002A ADD [BP+SI],CH
010E 0031 ADD [BX+DI],DH
0110 0006000D ADD [0D00],AL
1 7 8 9 A B C D E
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 07 00 15 00
DS:012F 23 00 31 00 0D 00 00 00
DS:0137 00 00 00 00 00 00 00 00
DS:013F BB 17 01 BE 2B 01 B9 14
DS:0147 00 BF 00 00 8B 85 03 01
DS:014F A9 01 00 74 09 89 04 81
DS:0157 C6 02 00 E9 06 00 89 07
DS:015F 81 C3 02 00 81 C7 02 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 FF FF AD DE 1B 05 C5 06 00 00
DS:0010 18 01 10 01 18 01 92 01 01 01 01 00 02 FF FF FF
DS:0020 FF FF FF FF FF FF FF FF FF FF FF FF EB 19 E4 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


Conversion
ASCII: BEL
Copy
Insert
Decimal: 7
Copy
Insert
Hexadecimal: 07
Copy
Insert
Binary: 111
Copy
Insert
Octal: 7
Copy
Insert
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD
AX 0000 SI 0000 CS 19F5 IP 0142 Stack +0 0000 Flags 7200
BX 0117 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 0 0 0 0
CMD >
013F BB1701 MOV BX,0117
0142 BE2B01 MOV SI,012B
0145 B91400 MOV CX,0014
0148 BF0000 MOV DI,0000
014B 8B850301 MOV AX,[0103+DI]
014F A90100 TEST AX,0001
0152 7409 JZ 015D
0154 8904 MOV [SI],AX
0156 81C60200 ADD SI,0002
DS:012B 07 00 15 00 23 00 31 00
DS:0133 0D 00 00 00 00 00 00 00
DS:013B 00 00 00 00 BB 17 01 BE
DS:0143 2B 01 B9 14 00 BF 00 00
DS:014B 8B 85 03 01 A9 01 00 74
DS:0153 09 89 04 81 C6 02 00 E9
DS:015B 06 00 89 07 81 C3 02 00
DS:0163 81 C7 02 00 39 CF 75 E0
DS:016B B8 00 4C CD 21 02 89 46
DS:0173 FE 85 C0 75 04 85 F6 74
DS:0000 CD 20 FF 9F 00 EA FF FF AD DE 1B 05 C5 06 00 00
DS:0010 18 01 10 01 18 01 92 01 01 01 01 00 02 FF FF FF
DS:0020 FF FF FF FF FF FF FF FF FF FF FF FF EB 19 E6 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

Thanks