CS/EEE/INSTR F241
Lab 9 – Advanced BIOS Interrupts for Display

Anubhav Elhence

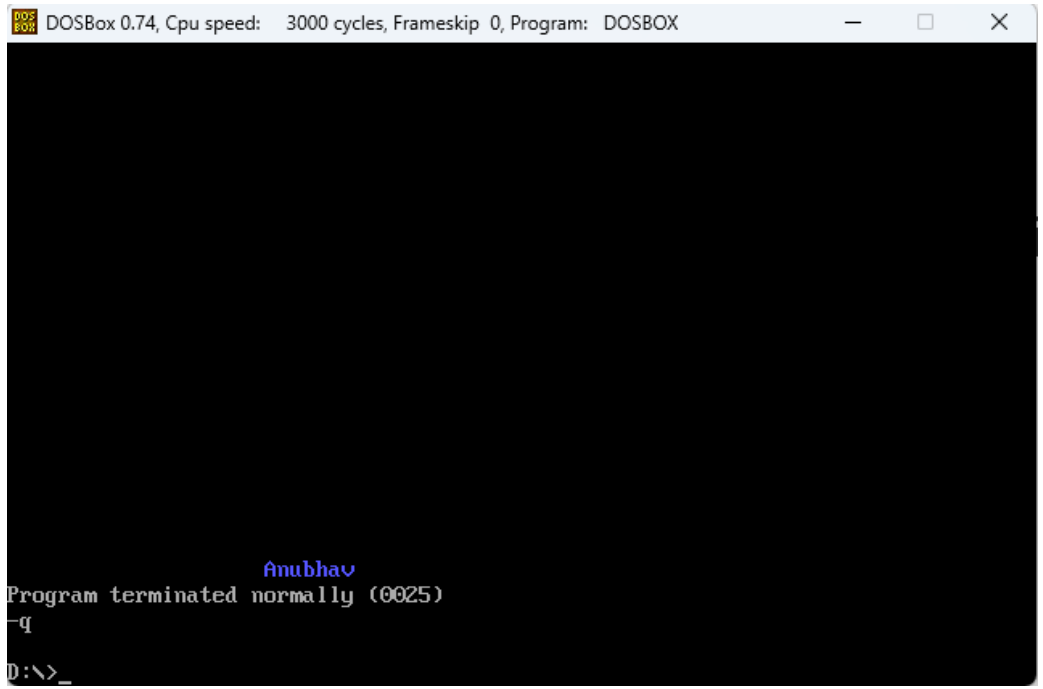
BIOS Interrupt 10h

- There are four main aspects to Video Display
  - Setting an appropriate video mode (or resolution, as you know it)
  - Reading/Setting the cursor position
  - Reading/Writing an ASCII character at a given cursor position
  - Working at the pixel level on the screen (for e.g., drawing a line, square on the screen

- We can choose what action to perform by identifying the interrupt option type, which is the value stored in register AH and providing whatever extra information that the specific option requires. We shall only discuss the important interrupt types in the next section. However, additional interrupts are provided at the end for your benefit.

Revision

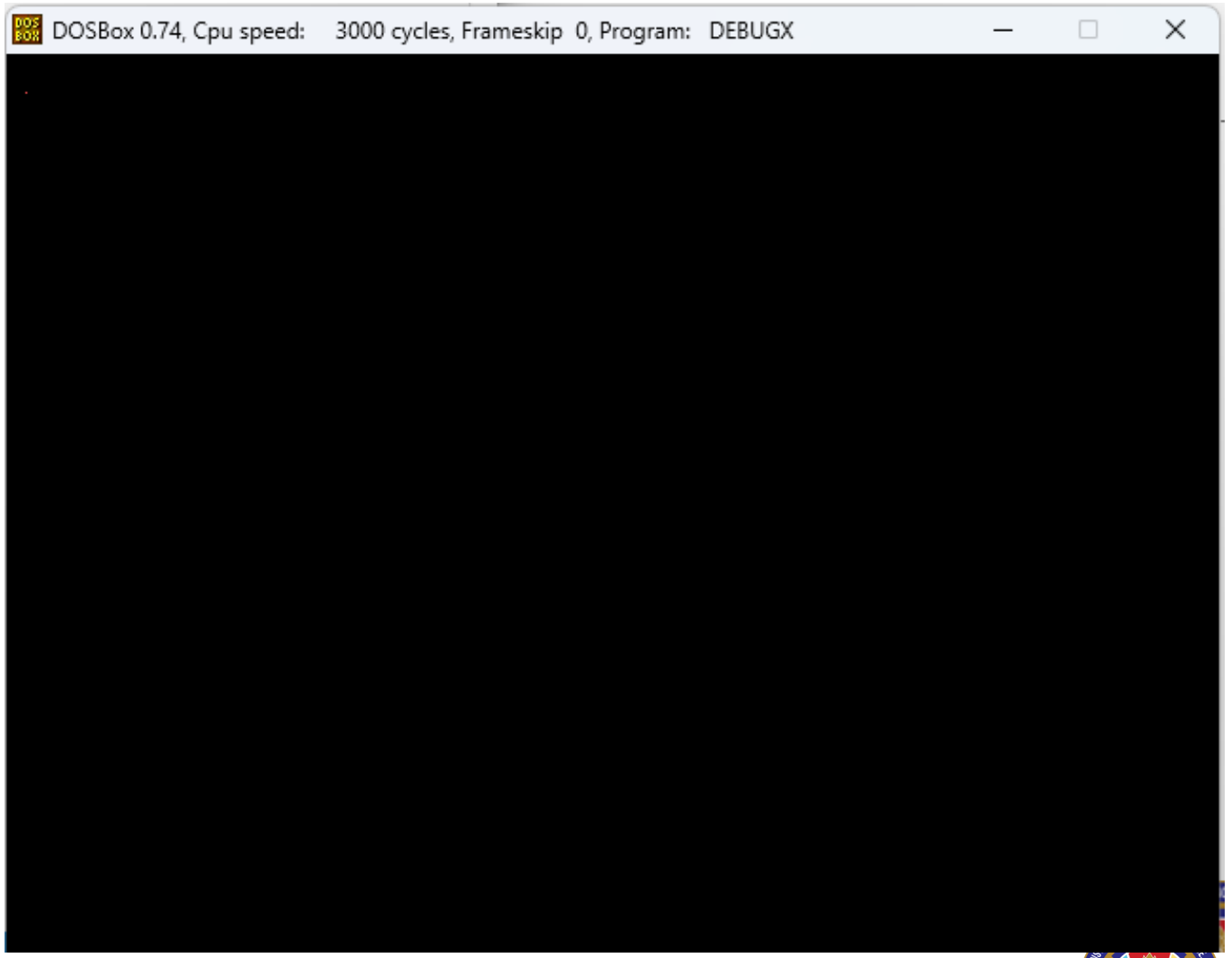
- Example 1: Write your name at cursor position (20, 20) in blue blinking text with a black background. Use display mode 03H or Text VGA mode.



Filling up a pixel

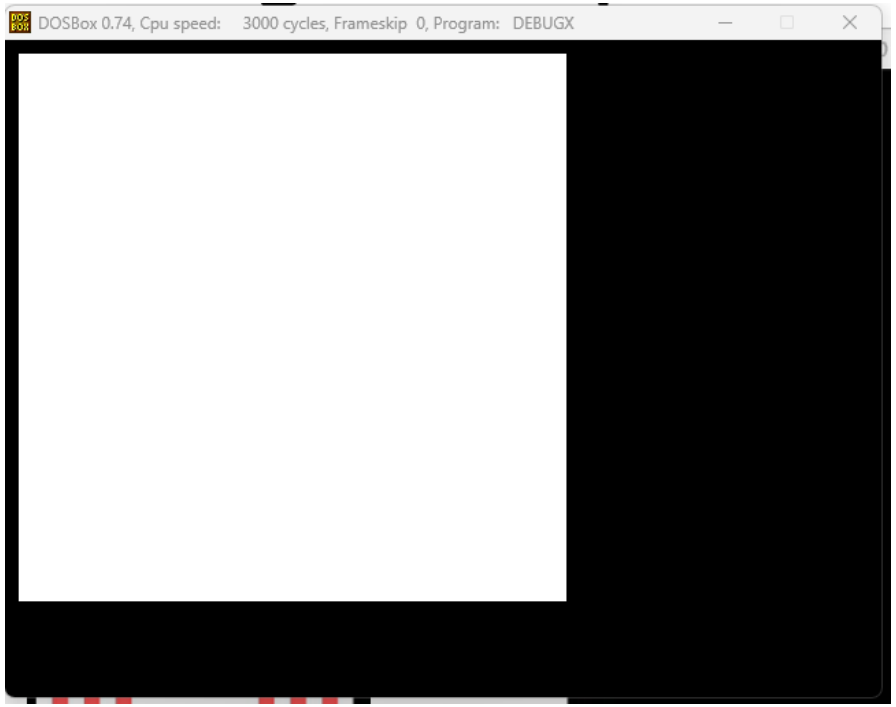
- Input:
  AH = 0Ch
  AL = pixel color //SAME ATTRIBUTE FORMAT AS PREVIOUS LAB
  CX = column.
  DX = row

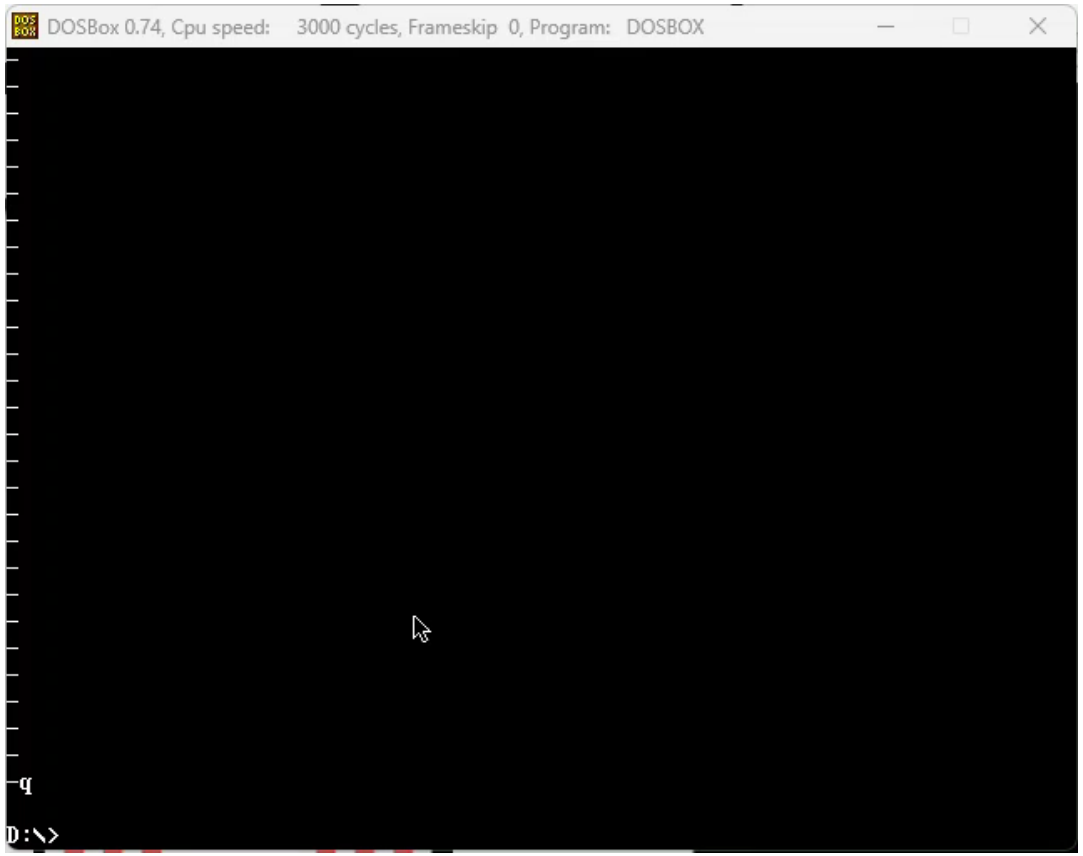
```
4   .code
5   .startup
6   
7       mov ah, 0
8       Mov al, 12h
9       int 10h ; set graphics video mode.
10  
11      mov ah, 0ch
12      mov al, 00001100b
13      mov cx, 10
14      mov dx, 20
15      int 10h ; set pixel.
16  
17      mov ah,07h
18      x1: int 21h
19      cmp al,'%'
20      jnz x1
21  
```



Follow along example

- Write an ALP to print a white square of pixel 400x 400 and starting from pixel (10,10)





- Set display mode to 640x480 with 16 colors:

```
14      ; Set display mode to 640x480 16 colors
15      MOV AH, 00H
16      MOV AL, 12H
17      INT 10H
```

- Set cursor position to (25,25) for graceful exit

```
19      ; Set cursor position to (20, 20)
20      MOV AH, 02H
21      MOV DH, 20
22      MOV DL, 20
23      MOV BH, 00
24      INT 10H
```

- Initialize Parameters for box drawing

```
26      ; Initialize parameters for box drawing
27      MOV rowstr, 10
28      MOV rowend, 410
29      MOV colmstr, 10
30      MOV colmend, 210
31      MOV cnt, 00
```

Paint the white Box code

```
; Paint the first box
PAINT1:
MOV SI, rowstr ; Row start
COLM1:
MOV CX, colmend ; Column end
ROW1:
DEC CX
MOV DI, CX
PUSH CX
MOV AH, 0Ch
MOV AL, 1111b ; Color for first box
MOV CX, DI
MOV DX, SI
INT 10h
POP CX
CMP CX, colmstr ; Column start
JNZ ROW1
INC SI
MOV AX, rowend ; Row end
CMP SI, AX
JNZ COLM1
```

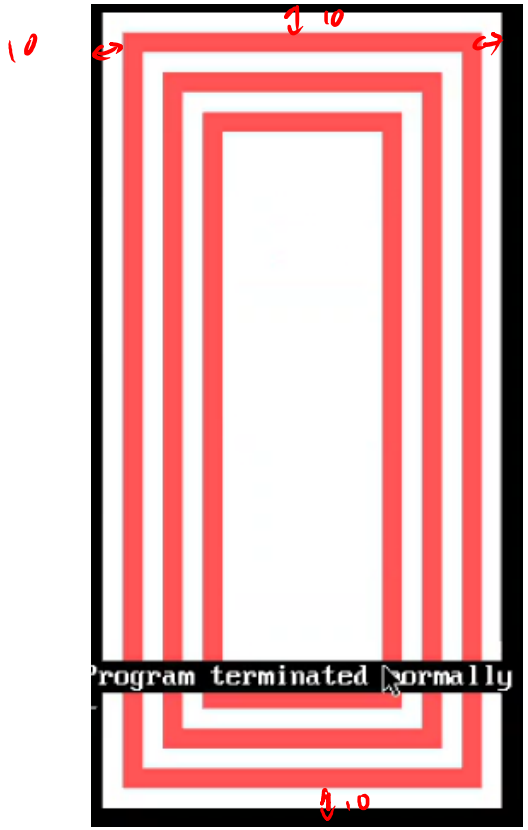
Blocking Function

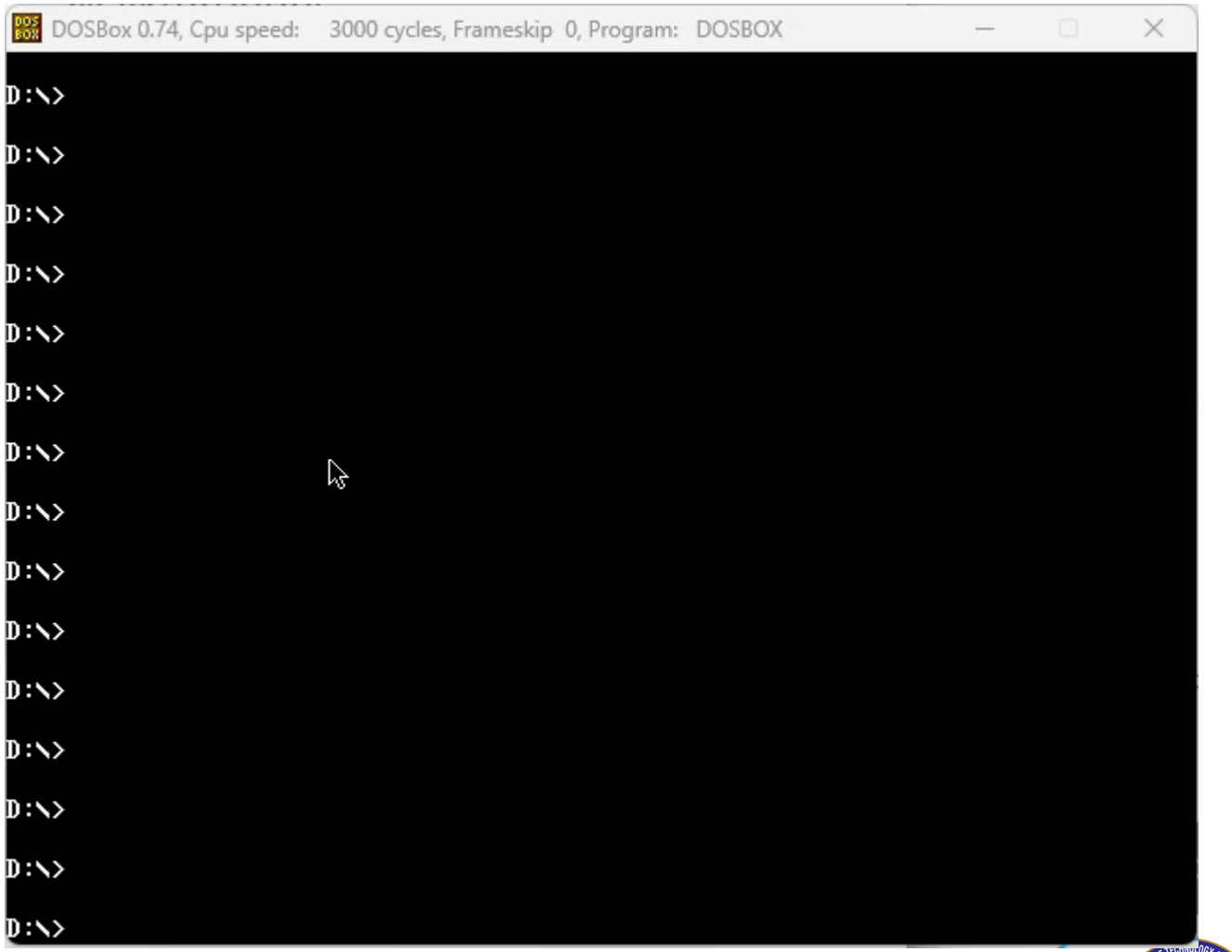
```
END1:
MOV AH, 07H
INT 21h
CMP AL, "%"
JNZ END1 ; Loop until '%' is received

TERM:
MOV AH, 4CH ; Terminate program
INT 21H
```

Follow along example

- Write an ALP to print the following pattern in 12h mode graphic mode





Anubhav Elhence and Dr. Vinay Chamola

- Set display mode to 640x480 with 16 colors:

```
14      ; Set display mode to 640x480 16 colors
15      MOV AH, 00H
16      MOV AL, 12H
17      INT 10H
```

- Set cursor position to (20,20)

```
19      ; Set cursor position to (20, 20)
20      MOV AH, 02H
21      MOV DH, 20
22      MOV DL, 20
23      MOV BH, 00
24      INT 10H
```

- Initialize Parameters for box drawing

```
26      ; Initialize parameters for box drawing
27      MOV rowstr, 10
28      MOV rowend, 410
29      MOV colmstr, 10
30      MOV colmend, 210
31      MOV cnt, 00
```

- Painting the first box: The code uses a nested loop structure to draw the box by iterating through rows and columns, setting the color, and calling the BIOS interrupt 10h to paint the pixel.

```
33      ; Paint the first box
34      PAINT1:
35      MOV SI, rowstr ; Row start
36      COLM1:
37      MOV CX, colmend ; Column end
38      ROW1:
39      DEC CX
40      MOV DI, CX
41      PUSH CX
42      MOV AH, 0Ch
43      MOV AL, 1111b ; Color for first box
44      MOV CX, DI
45      MOV DX, SI
46      INT 10h
47      POP CX
48      CMP CX, colmstr ; Column start
49      JNZ ROW1
50      INC SI
51      MOV AX, rowend ; Row end
52      CMP SI, AX
53      JNZ COLM1
```

- Change vertices for the next box:
  After drawing the first box, the code adjusts the row and column start and end points, as well as increments the counter variable to keep track of how many boxes have been drawn.

```
; Change vertices for the next box
LEA SI, rowstr
ADD WORD PTR[SI], 10
LEA SI, rowend
SUB WORD PTR[SI], 10
LEA SI, colmstr
ADD WORD PTR[SI], 10
LEA SI, colmend
SUB WORD PTR[SI], 10
LEA SI, cnt
INC BYTE PTR[SI]
```

- Check if the counter has reached 7: If the counter reaches 7 or more, the code jumps to the termination process.

```
MOV AL, 07h
MOV BL, cnt
CMP BL, AL
JGE TERM ; Terminate if cnt >= 7
```

- Paint the second box:
  Similar to the first box, this part paints the second box with a different color using the same nested loop structure

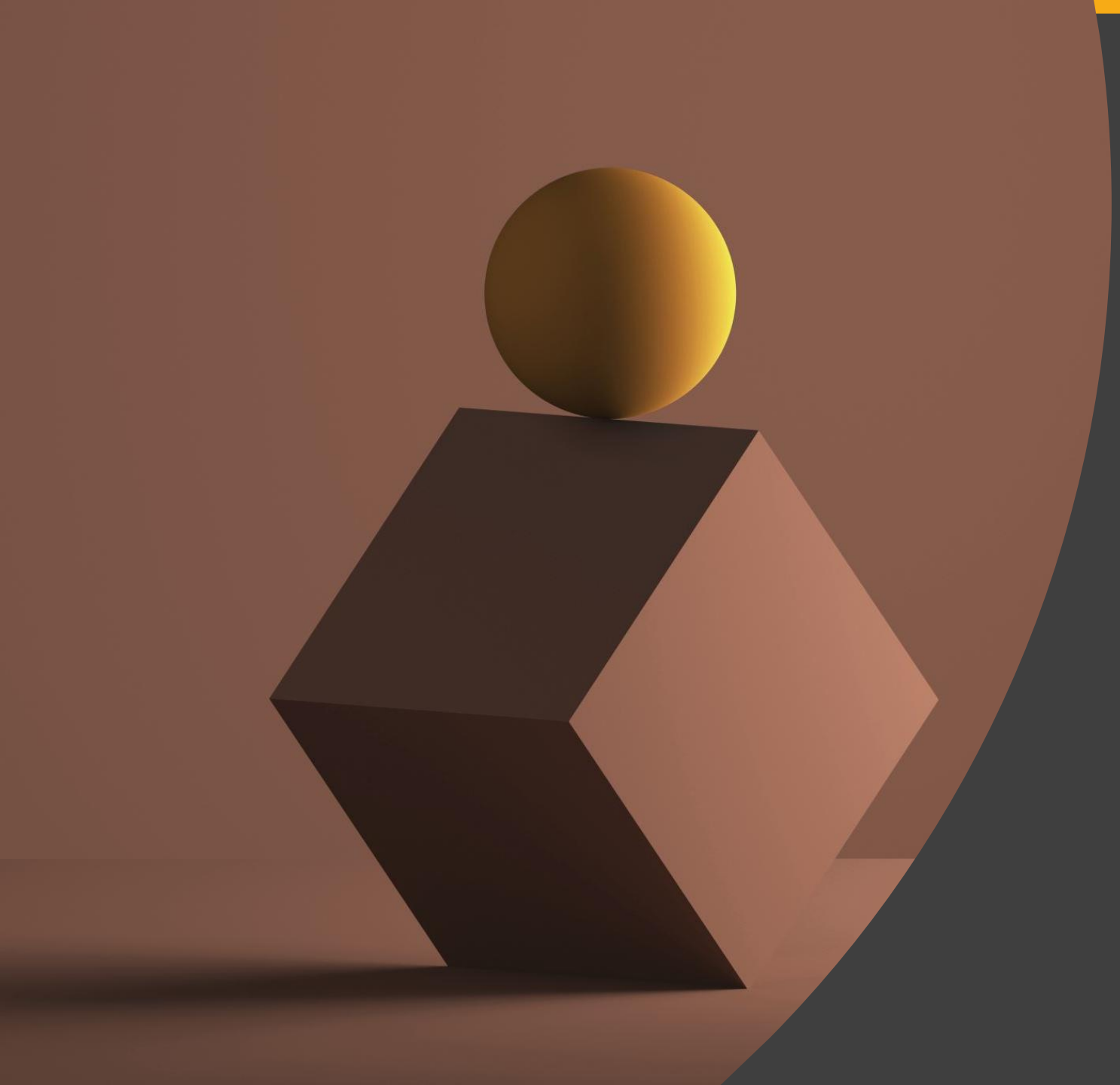
```
; Paint the second box
MOV SI, rowstr ; Row start
COLM2:
MOV CX, colmend ; Column end
ROW2:
DEC CX
MOV DI, CX
PUSH CX
MOV AH, 0Ch
MOV AL, 1100b ; Color for second box
MOV CX, DI
MOV DX, SI
INT 10h
POP CX
CMP CX, colmstr ; Column start
JNZ ROW2
INC SI
MOV AX, rowend ; Row end
CMP SI, AX
JNZ COLM2
```

- Change vertices for the next box: The code again adjusts the row and column start and end points and increments the counter variable.

```
; Change vertices for the next box
LEA SI, rowstr
ADD WORD PTR[SI], 10
LEA SI, rowend
SUB WORD PTR[SI], 10
LEA SI, colmstr
ADD WORD PTR[SI], 10
LEA SI, colmend
SUB WORD PTR[SI], 10
LEA SI, cnt
INC BYTE PTR[SI]
```

Time for Lab Tasks:

Please check the description of this video.



Thankyou

Anubhav Elhence and Dr. Vinay Chamola