Name:	Ketan Mahadik
Roll No:	24
Class/Sem:	SE/IV
Experiment No.:	3
Title:	Program for drawing square using Assembly Language.
Date of Performance:	05/02/24
Date of Submission:	12/02/24
Marks:	
Sign of Faculty:	

Aim: Program for drawing square using Assembly Language.

Theory: INT 10h is a video service bios interrupt. It includes services like setting the video mode, character and string output and reading and writing pixels in graphics mode. To use the BIOS interrupt load ah with the desired sub-function. Load other required parameters in other registers and make a call to INT 10h.

INT 10h/AH = 0ch -Write graphics pixel.

Input:
AL = pixel colour
CX = column
DX = row

Algorithm:
1. Start

2. Initialize ax to 0013h for graphics mode.

3. Set the Counter bx to 60 h.

4. Initialize the co-ordinates cx and dx to 60h.

5. Set the Color.

6. Set Display Mode function by making ah = 0ch.

7. Increment cx and Decrement bx.

8. Repeat step 7 until bx = 0.

9. Initialize the counter by making bx = 60h.

10. Set the color.

11. Set Display Mode function by making ah = 0ch.

12. Increment dx & Decrement bx.

13. Repeat step 12 until bx = 0.

14. Initialize the counter by making bx = 60h.

15. Set the Color.

16. Set Display Mode function by making ah = 0ch.

17. Decrement cx and Decrement bx.

18. Repeat step 17 until bx = 0.

19. Initialize the counter by making bx = 60h.

20. Set the color.

21. Set Display Mede function by making ah = 0ch.

22. Decrement dx & Decrement bx.

23. Repeat step 22 until bx = 0.

24. To end the program use DOS interrupt:

    1) Load ah = 4ch.

    2) Call int 21h.

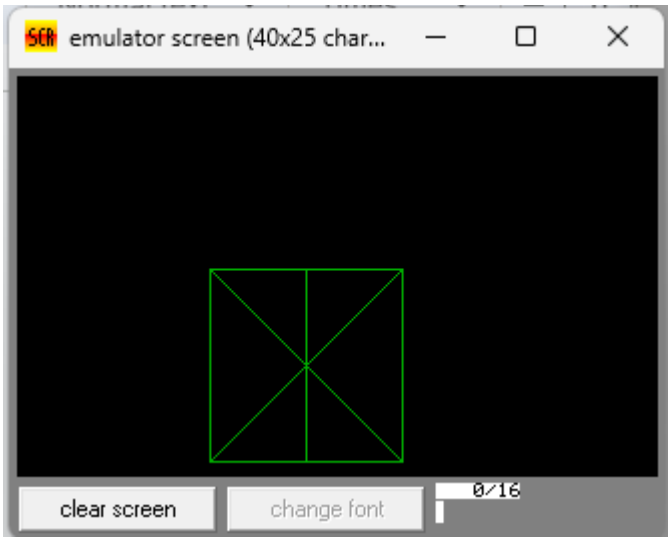
25. Stop.

Code:

```
MOV AX,0013H
INT 10H

MOV BX,60H
MOV CX,60H
MOV DX,60H

MOV AL,02H

L1:MOV AH,0CH

INC CX
DEC BX
INT 10H

JNZ L1
MOV BX,60H

L2:MOV AH,0CH

INC DX
DEC BX
INT 10H

JNZ L2

MOV BX,60H
```

```
L3:MOV AH,0CH

DEC CX
DEC BX
INT 10H

JNZ L3
MOV BX,60H

L4:MOV AH,0CH

DEC DX
DEC BX
INT 10H

JNZ L4
MOV BX,60H

L5:MOV AH,0CH

INC CX
INC DX
DEC BX
INT 10H

JNZ L5
MOV BX,60H
MOV CX,60H

L6:MOV AH,0CH

INC CX
DEC DX
DEC BX
INT 10H

JNZ L6
MOV BX,60H
MOV CX,90H

L7:MOV AH,0CH

INC DX
DEC BX
INT 10H

JNZ L7
```

Output:



Conclusion:

1. Explain the use of int 10.

   Ans: The "int 10h" instruction is a software interrupt used in the x86 architecture, including the Intel 8086 microprocessor. It is primarily used for BIOS video services, allowing programmers to interact with the computer's display hardware through the BIOS (Basic Input/Output System) layer.

   When the "int 10h" instruction is executed, control is transferred to a predefined interrupt handler within the BIOS, which then provides various video-related services based on the value passed in the AH register. Common uses of "int 10h" include:

   1. Display Output: It allows programs to display characters, strings, and graphical elements on the screen

   2. Cursor Control: Programs can manipulate the position and appearance of the cursor on the screen.

   3. Screen Clearing: It provides functions to clear all or part of the screen.

   4. Screen Mode Setting: Programs can change the video mode, altering the resolution and color depth of the display.

   5. Scrolling: It supports scrolling of text and graphics on the screen.

   6. Palette Management: It allows manipulation of the color palette for graphics modes.

   Overall, "int 10h" provides a standardized interface for accessing video-related functions, making it easier for programmers to develop applications that interact with the display hardware across different computer systems.

2. Explain hardware interrupts.

   Ans: Hardware interrupts are signals sent by external devices or internal components to the CPU, indicating that they require attention. Upon receiving an interrupt signal, the CPU suspends its current task, saves its state, and jumps to the corresponding interrupt service routine (ISR), which is a specific piece of code designed to handle the interrupt's task. ISRs address various events, such as keyboard input, disk I/O requests, or timer updates. After completing the ISR, the CPU resumes the interrupted task. Interrupt controllers prioritize interrupts based on their importance, ensuring critical events are handled promptly. Overall, hardware interrupts facilitate efficient communication between the CPU and peripherals, enabling multitasking and real-time processing in computer systems.