```
thestack segment stack
    db 256 dup(?)
thestack ends

thedata segment 'data'
    row db ?
    clm db ?
    rrow db ?
    rclm db ?
    temp db ?
    res dw ?
    row1 db ?
    row2 db ?
    row3 db ?
    freq dw 0
    p1 db "Enter the first 3*3 matrix:",0dh,0ah,"$"
    p2 db "Enter the second 3*3   matrix:",0dh,0ah,"$"
    p3 db "The product of the above matrices is:",0dh,0ah,"$"
    p4 db "Enter 1 to continue,any key to exit: $"
thedata ends

thecode segment 'code'
    assume cs:thecode,ds:thedata,ss:thestack
main proc

    mov ax,seg row
    mov ds,ax

    ; Macro for making the page blank and setting cursor.
    blank macro
        mov ax,0500
        int 10h
        mov ah,0h
        mov al,0h
        int 10h
        mov ax,0600h
        mov cx,0
        mov dx,1827h
        mov bh,07h
        int 10h
        mov ah,2
        mov bh,0
        mov dx,0
        int 10h
    endm

    ; Macro for positioning of cursor.
    setcursor macro
        mov bh,0
        mov ah,02
        int 10h
    endm

    ; Macro for getting the position.
```

```
getcursor macro
    mov ah,03
    mov bh,0
    int 10h
endm

; Macro for displaying the number.
dispno macro
    mov ah,0eh
    mov bx,0007h
    int 10h
endm

; Macro for displaying of prompts.
display macro A
    mov ah,09h
    mov dx,offset A
    int 21h
endm

; Macro for getting binary equivalent of number in AL.
getnumber macro
    mov ah,8
    mov bh,0
    int 10h
    xor ah,ah
    sub al,30h
endm


; MAIN PROCEDURE FOLLOWS.

; This block prompts for inputs and uses  cursor procedure .
; It displays the output using  result procedure  repeatedly.
; It passes the row number and column to this  procedure.

ask:
    blank
    display p1
    call cursor
    mov dx,0400h
    setcursor
    display p2
    call cursor

    mov dx,0800h
    setcursor
    display p3
    mov row,1

pos11:
    mov clm,0
    cmp row,4
    je pos14
```

```
pos12:
    cmp clm,6
    je pos13
    call result
    add clm,2
    jmp pos12

pos13:
    inc row
    jmp pos11

; This block is for users choice to continue or exit.

pos14:
    mov dx,0c00h
    setcursor
    display p4
    mov ah,1
    int 21h
    cmp al,31h
    je ask
    mov ah,4ch
    int 21h
    main endp

; This procedures is called to input the elements.
cursor proc
    getcursor
    mov row1,dh
    inc dh
    mov row2,dh
    inc dh
    mov row3,dh

    pos1:
        getcursor
        mov ah,0
        int 16h
        cmp ah,48h
        jne pos2
        dec dh
        setcursor
        jmp pos1

    pos2:
        cmp ah,50h
        jne pos3
        inc dh
        setcursor
        jmp pos1

    pos3:
        cmp ah,4dh
```

```
        jne pos4
        inc dl
        setcursor
        jmp pos1

    pos4:
        cmp ah,4bh
        jne pos5
        dec dl
        setcursor
        jmp pos1

    pos5:
       cmp al,0dh
       je return

    pos6:
        cmp dl,0
        je pos7
        cmp dl,2
        je pos7
        cmp dl,4
        je pos7
        call beep
        jmp pos1

    pos7:
        cmp dh,row1
        jne pos8
        dispno
        jmp pos1

    pos8:
        cmp dh,row2
        jne pos9
        dispno
        jmp pos1

    pos9:
        cmp dh,row3
        jne pos10
        dispno
        jmp pos1

    pos10:
        call beep
        jmp pos1

    return:
        ret
cursor endp

; This procedure produces the beep if attempt is made to enter
; an element at wrong position.
```

```
beep proc
    push ax
    push bx
    push dx
    mov dx,0
    in al,61h
    and al,0fch

    noise:
        mov freq,7000
        inc dx
        cmp dx,8
        je quit

    more:
        xor al,02h
        mov cx,freq
        cmp cx,8000
        je noise
        inc freq
        out 61h,al

    here:
        loop here
        jmp more

    quit:
        pop dx
        pop bx
        pop ax
        ret
beep endp

; The result procedure
result proc

    mov dh,row
    mov dl,0
    setcursor
    getnumber
    mov temp,al
    mov dh,5
    mov dl,clm
    setcursor
    getnumber
    mul temp
    mov res,ax

    mov dh,row
    mov dl,2
    setcursor
    getnumber
    mov temp,al
```

```
mov dh,6
mov dl,clm
setcursor
getnumber
mul temp
add res,ax

mov dh,row
mov dl,4
setcursor
getnumber
mov temp,al
mov dh,7
mov dl,clm
setcursor
getnumber
mul temp
add res,ax

mov bl,row
add bl,8
mov rrow,bl
cmp clm,0
jne pos20
mov rclm,0
jmp pos22

pos20:
    cmp clm,2
    jne pos21
    mov rclm,4
    jmp pos22

pos21:
    mov rclm,8

pos22:
    mov dh,rrow
    mov dl,rclm
    setcursor
    mov ax,res
    mov bx,10
    xor cx,cx

pos23:
    xor dx,dx
    div bx
    push dx
    inc cx
    or ax,ax
    jnz pos23

pos24:
    pop dx
```

```
            mov ax,dx
            add al,30h
            dispno
            loop pos24
            ret
    result endp

thecode ends
end main
```