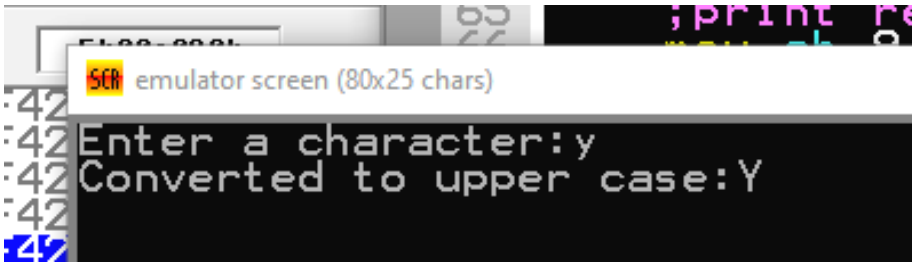
Experiment 1: Write an assembly language program to convert a lowercase letter to an uppercase letter

```
.model small
.stack 100h
.data
prompt db 'Enter a character:$'
msg_upper db 'Converted to upper case:$'
msg_lower db 'Converted to lower case:$'
msg_not_letter db 'Input is not a letter.$'
input_char db ?,'$' ;? means not to initialize memory
.code
main proc
    ;print prompt
    mov ax,@data
    mov ds,ax
    mov ah,9
    lea dx,prompt
    int 21h
    ;read input character
    mov ah,1
    int 21h
    mov bl,al ;save input character
    MOV dl, 10
    MOV ah, 02h
    INT 21h
    MOV dl, 13
    MOV ah, 02h
    INT 21h
    ;check if input is a letter
    cmp bl,'A'
    jb not_letter
    cmp bl,'Z'
    jbe upper_case
    cmp bl,'a'
    jb not_letter
    cmp bl,'z'
    ja not_letter

    lower_case:
```

```
    ;convert to upper case
    and bl,11011111b ;returns 1
    ;print result
    mov ah,9
    lea dx, msg_upper
    int 21h
    mov ah,2
    mov dl,bl
    int 21h
    jmp exit_program
    upper_case:
    ;convert to lower case
    or bl,00100000b  ;return 0
    ;print result
    mov ah,9
    lea dx,msg_lower
    int 21h
    mov ah,2
    mov dl,bl
    int 21h
    jmp exit_program
    not_letter:
    ;print error message
    mov ah,9
    lea dx,msg_not_letter
    int 21h
    exit_program:
    ;exit program
    mov ah,4ch
    int 21h
    main endp
end main
```

Output

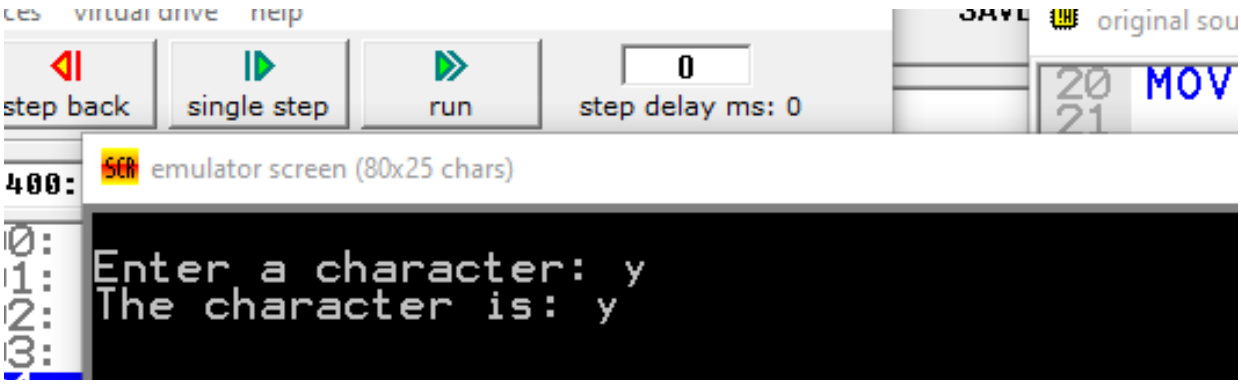


Experiment 2: Write an assembly language program to read a character. If it is "y" or "Y", display it: otherwise, terminate the program.

```
.MODEL SMALL
.STACK 100H

.DATA
    msg          DB 10, 13, 'Enter a character: $'
    msgDisplay   DB 10, 13, 'The character is: $'
    msgTerminated DB 10, 13, 'Program terminated.$'
    character    DB ?


.CODE
    MOV AX, @DATA
    MOV DS, AX

    MOV AH, 09H  ; Display prompt message
    LEA DX, msg
    INT 21H

    MOV AH, 01H  ; Read character from the user
    INT 21H
    MOV character, AL

    CMP AL, 'y'  ; Compare with lowercase 'y'
    JE  displayCharacter

    CMP AL, 'Y'  ; Compare with uppercase 'Y'
    JE  displayCharacter

    ; Terminate the program
    MOV AH, 4CH
    INT 21H


displayCharacter:
    MOV AH, 09H  ; Display the character
    LEA DX, msgDisplay
    INT 21H
```

```
    MOV DL, character
    MOV AH, 02H
    INT 21H

    ; Terminate the program
    MOV AH, 4CH
    INT 21H

END
```

Output:

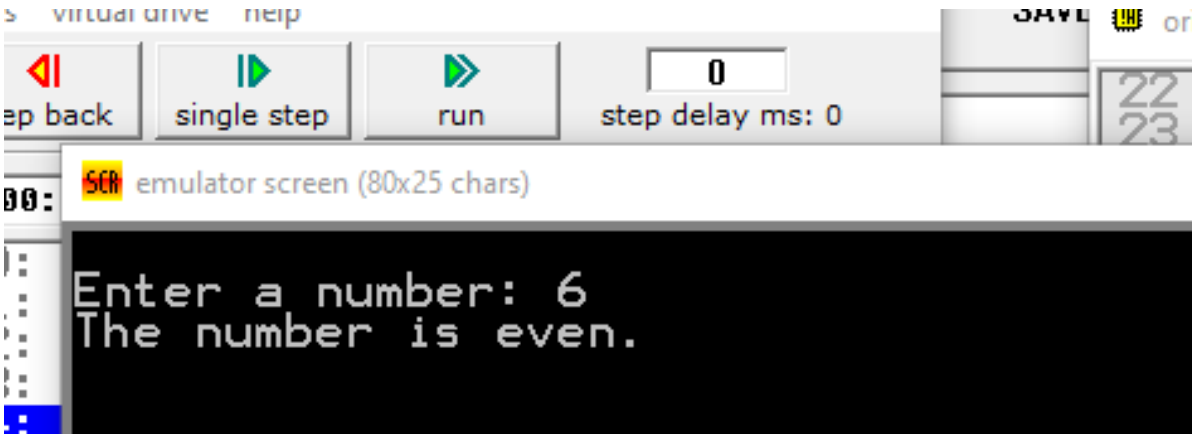


Experiment 3: Write an assembly language program to determine whether a number is odd or even.

```
.MODEL SMALL
.STACK 100H

.DATA
    msgPrompt    DB 10, 13, 'Enter a number: $'
    msgEven      DB 10, 13, 'The number is even.$'
    msgOdd       DB 10, 13, 'The number is odd.$'
    number       DW ?


.CODE
    MOV AX, @DATA
    MOV DS, AX

    MOV AH, 09H  ; Display prompt message
    LEA DX, msgPrompt
    INT 21H

    MOV AH, 01H  ; Read a character from the user
    INT 21H
    SUB AL, '0'  ; Convert ASCII digit to numerical value
    MOV number, AX

    MOV AX, number
    AND AX, 0001H  ; Perform bitwise AND with 0001b

    CMP AX, 0     ; Check if the result is zero
    JE  evenNumber

    ; Number is odd
    MOV AH, 09H  ; Display "The number is odd"
    LEA DX, msgOdd
    INT 21H

    JMP exitProgram


evenNumber:
```

```
    ; Number is even
    MOV AH, 09H  ; Display "The number is even"
    LEA DX, msgEven
    INT 21H

exitProgram:
    ; Terminate the program
    MOV AH, 4CH
    INT 21H

END
```

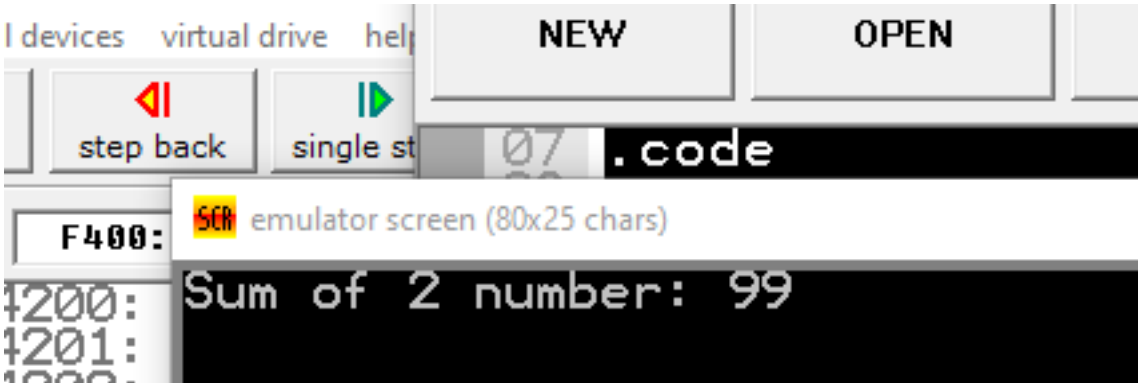
Output



Experiment 4: Write an assembly language program to add two decimal numbers.

```
.model small
.stack
.data
    val1 db 89
    val2 db 10
    msg db 'Sum of 2 number: $'
.code
main proc
    mov ax, @data
    mov ds,ax
    mov ax,0
    mov al,val1
    add al,val2
    aam
    add ax,3030h
    push ax
    lea dx,msg
    mov ah,09h
    int 21h
    pop ax
    mov dl,ah
    mov dh,al
    mov ah,02h
    int 21h
    mov dl,dh
    mov ah,02h
    int 21h
    mov ax,4c00h
    int 21h
main endp
end main
```

Output

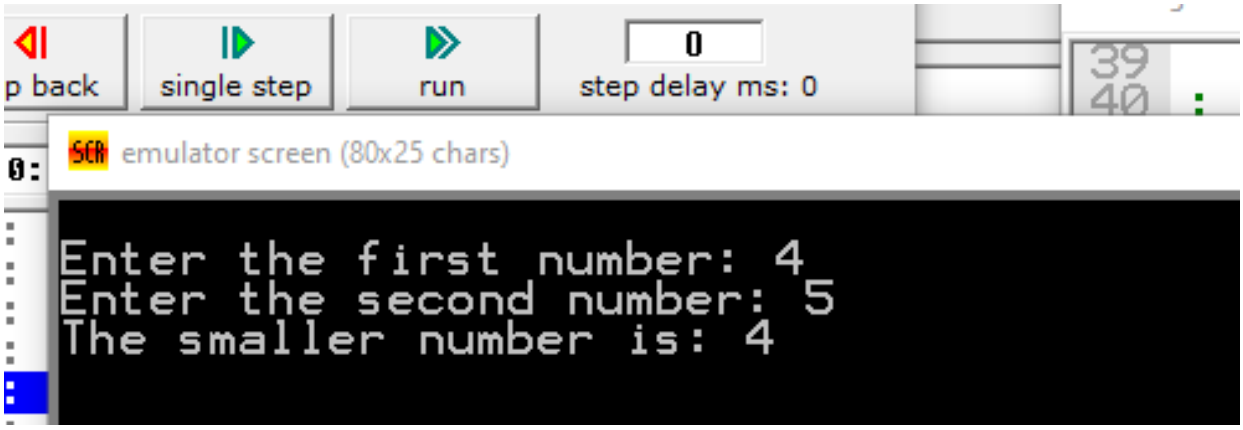


Experiment 5: Write an assembly language program to input two numbers, compare them, and display the smaller one.

```
.model small
.stack 100h

.data
    prompt1 db 10, 13, 'Enter the first number: $'
    prompt2 db 10, 13, 'Enter the second number: $'
    result db 10, 13, 'The smaller number is: $'

.code
    main proc
        mov ax, @data
        mov ds, ax

        ; Display prompt to enter the first number
        mov ah, 09h
        lea dx, prompt1
        int 21h

        ; Input the first number
        mov ah, 01h
        int 21h
        sub al, 30h
        mov bl, al

        ; Display prompt to enter the second number
        mov ah, 09h
        lea dx, prompt2
        int 21h

        ; Input the second number
        mov ah, 01h
        int 21h
        sub al, 30h
        mov cl, al

        ; Compare the numbers
        cmp bl, cl
        jge first_number_smaller
```

```
        ; Display the smaller number
        mov ah, 09h
        lea dx, result
        int 21h
        mov dl, bl
        add dl, 30h
        mov ah, 02h
        int 21h
        jmp exit_program

    first_number_smaller:
        ; Display the smaller number
        mov ah, 09h
        lea dx, result
        int 21h
        mov dl, cl
        add dl, 30h
        mov ah, 02h
        int 21h

    exit_program:
        mov ah, 4Ch
        int 21h
    main endp
end main
```

Output



Experiment 6: Write an assembly language program to find the largest element of an array.

```
; Program to find the largest element of an array

.model small
.stack 100h

.data
array db 11h, 2h, 7h, 5h, 9h, 1h, 4h ; Example array
array_size equ ($ - array) ; Calculate array size
largest_element db ? ; Variable to store the largest element

.code
main proc
    mov ax, @data
    mov ds, ax

    ; Initialize largest_element with the first element of the array
    mov al, array
    mov largest_element, al

    ; Loop through the array to find the largest element
    mov cx, array_size
    mov si, 1
    mov bl, al ; Store the current largest element in bl

loop_start:
    ; Compare current element with largest_element
    mov al, array[si]
    cmp al, bl
    jle next_element ; Jump if the current element is less than or equal to
the largest_element

    ; Update largest_element if the current element is larger
    mov largest_element, al
    mov bl, al

next_element:
    inc si
    loop loop_start
```

```
    ; Display the largest element
    mov ah, 02h ; Function to display a single character
    mov dl, largest_element ; Load the largest element into dl
    add dl, 30h ; Convert the number to its ASCII representation
    int 21h ; Display the character

    mov ah, 4Ch ; Terminate the program
    int 21h
main endp

end main
```

Output