CSCI 320-54 – Assignment 7: X86 Lab 1

Created by: Thomas Hoerger

Objectives

Analyze, edit, assemble, link and run the programs. Write a program to display a "?", read two decimal digits whose sum is less than 10, and display them and their sum on the next line, with an appropriate message.

Equipment Used

GUI Turbo Assembler 5.1

Procedure

To edit a program use notepad
Save with .asm extension.

To assemble the program

```
C: tasm pgm4_1; or masm pgm4_1
```

To link

```
C:tlink pgm4_1;
```

To run

```
Pgm4_1
```

1.Edit, assemble, link and run the above x86 programs. Make sure to fix the errors.

2.Write a program to (a) display a "?", (b) read two decimal digits whose sum is less than 10, (c) Display them and their sum on the next line, with an appropriate message.
Sample Execution:

```
?27
THE SUM OF 2 AND 7 IS 9
```

Submit your .docx report.

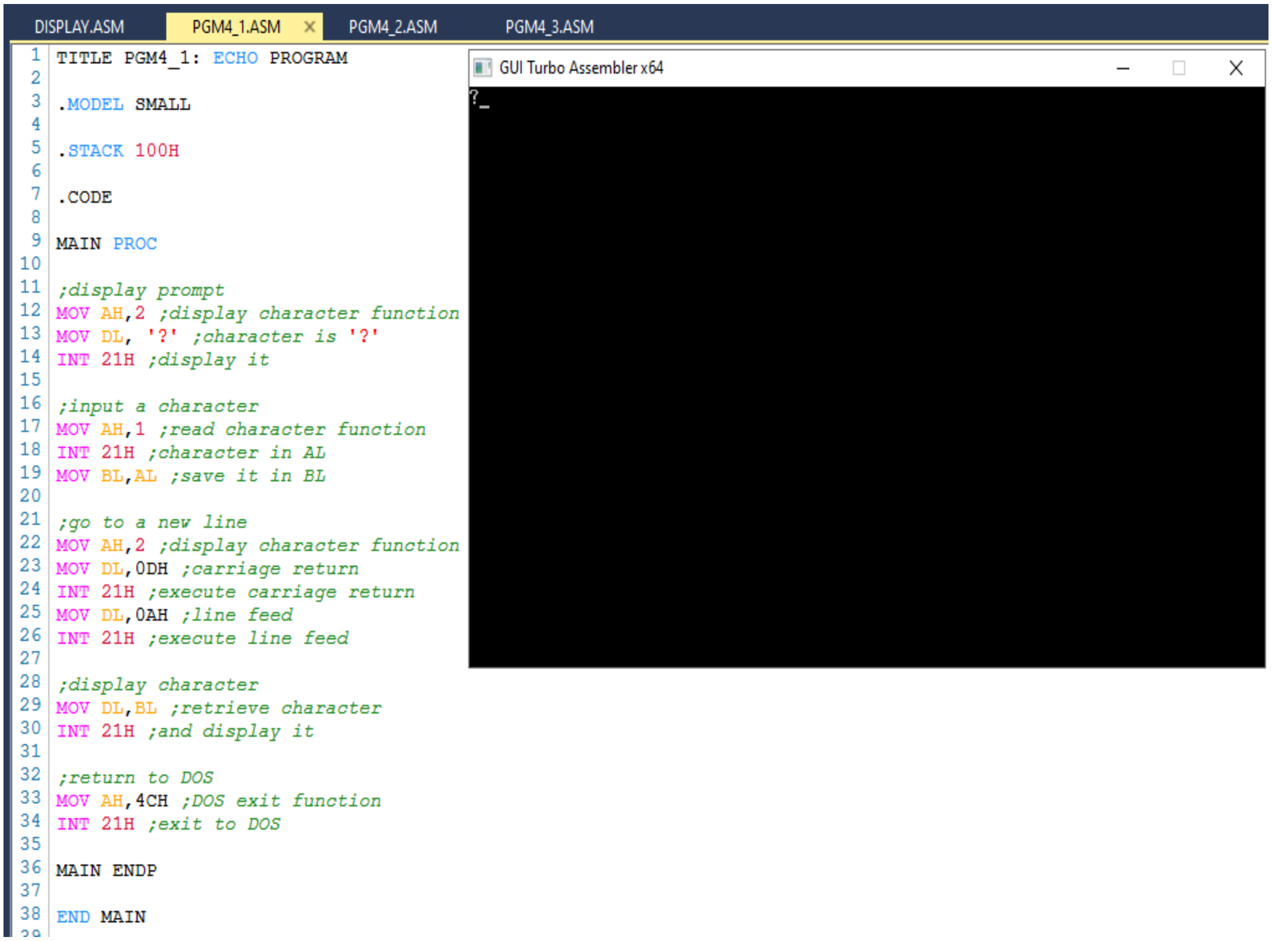
New Operations Learned

MOV AX: Moves into the AX register.
MOV DS, AX: Set the data segment register (DS) to point to the data segment.
MOV AH: Move the value into the AH register.
MOV DX, OFFSET msgPrompt: Move the offset address of msgPrompt into the DX register.
INT 21h: Call interrupt 21h used to display a string or read a character.
SUB AL, '0': Subtract the ASCII value of '0' from AL to convert the character to its corresponding numerical value.
ADD AL, num2: Add the value in num2 to AL.
CMP sum, 10: Compare the value of sum with 10.
JAE overflow: Jump to the overflow label if sum is greater than or equal to 10.
LEA DX,MSG2: get message MSG2.

Program Description

PGM4_1: This program displays a ?.
PGM4_2: This program displays “Hello!”.
PGM4_3: This program takes a lower-case letter in as an input then out puts it as a upper case letter.
DISPLAY.ASM: This program displays a ?, then reads two decimal digits whose sum is less than 10 and displays them and their sum on the next line.

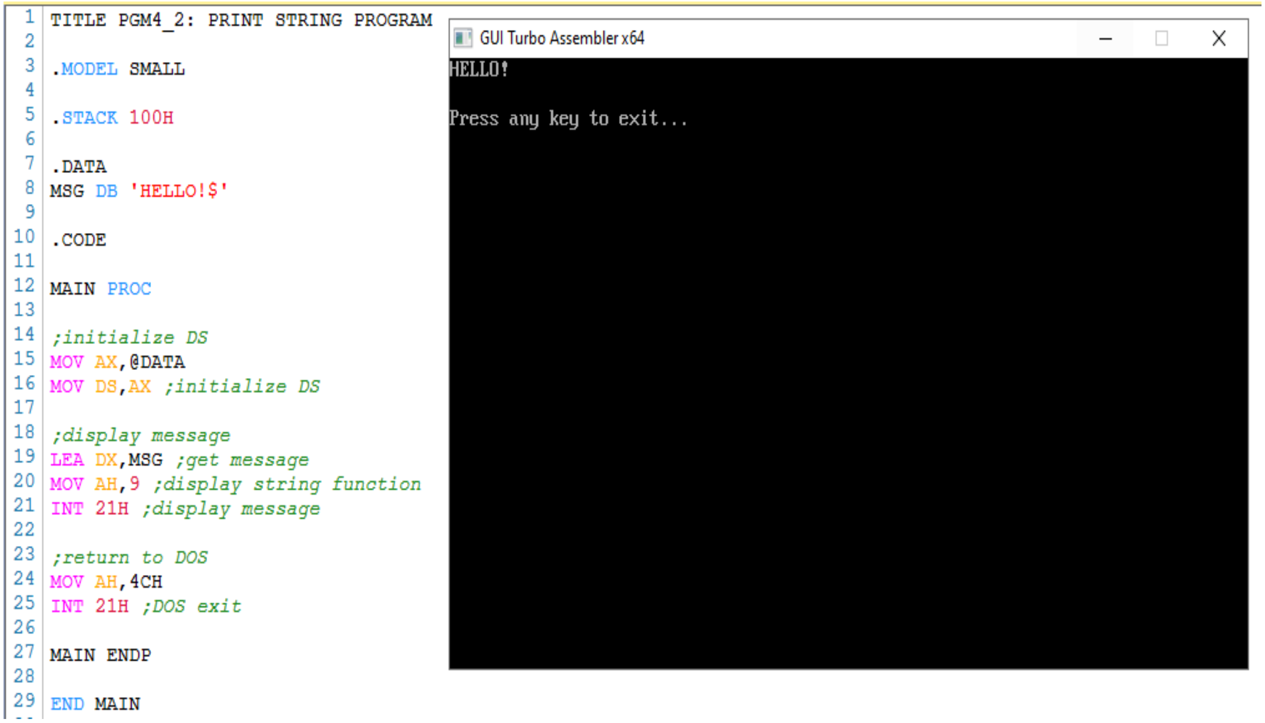
SOURCE CODE



```
TITLE PGM4_1: ECHO PROGRAM

.MODEL SMALL

.STACK 100H

.CODE

MAIN PROC

;display prompt
MOV AH,2 ;display character function
MOV DL, '?' ;character is '?'
INT 21H ;display it

;input a character
MOV AH,1 ;read character function
INT 21H ;character in AL
MOV BL,AL ;save it in BL

;go to a new line
MOV AH,2 ;display character function
MOV DL,0DH ;carriage return
INT 21H ;execute carriage return
MOV DL,0AH ;line feed
INT 21H ;execute line feed

;display character
MOV DL,BL ;retrieve character
INT 21H ;and display it

;return to DOS
MOV AH,4CH ;DOS exit function
INT 21H ;exit to DOS

MAIN ENDP

END MAIN
```

```
?_
```

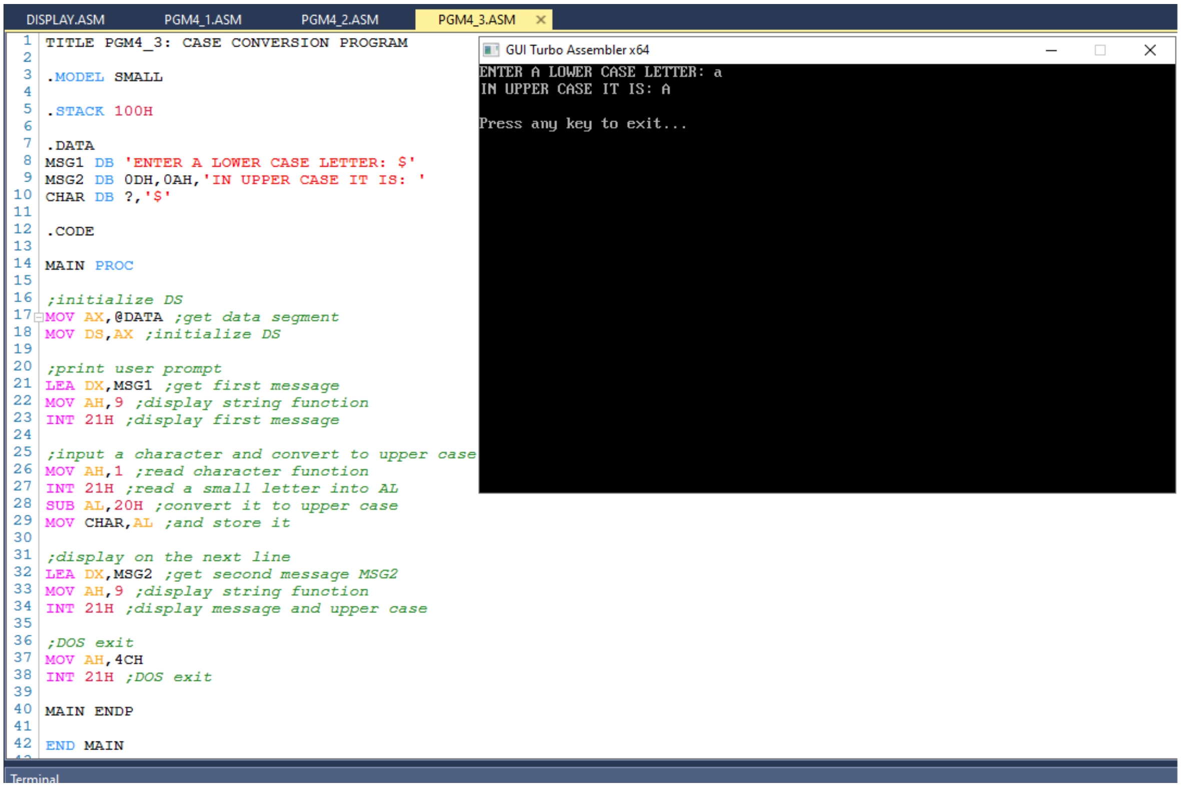
Figure 1 shows the source code for PGM4_1 and the terminal output after execution.



```
TITLE PGM4_2: PRINT STRING PROGRAM

.MODEL SMALL

.STACK 100H

.DATA
MSG DB 'HELLO!$'

.CODE

MAIN PROC

;initialize DS
MOV AX,@DATA
MOV DS,AX ;initialize DS

;display message
LEA DX,MSG ;get message
MOV AH,9 ;display string function
INT 21H ;display message

;return to DOS
MOV AH,4CH
INT 21H ;DOS exit

MAIN ENDP

END MAIN
```

```
HELLO!

Press any key to exit...
```

Figure 2 shows the source code for PGM4_2 and the terminal output after execution.

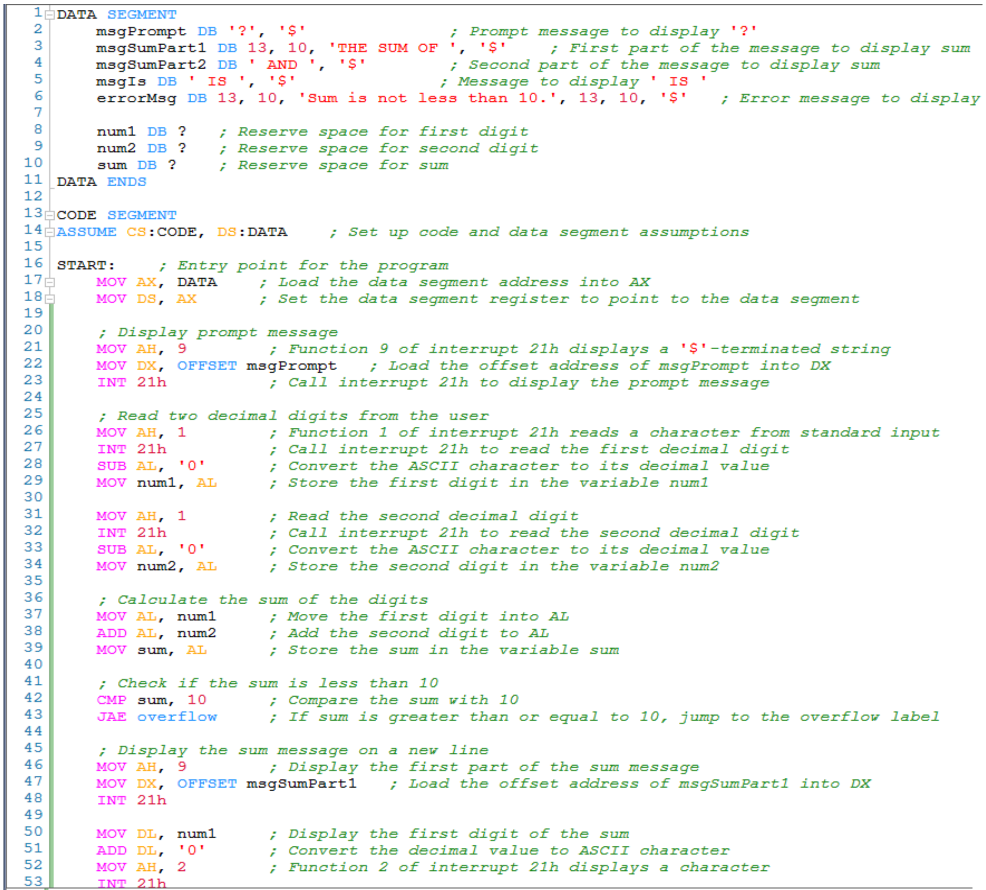


```
TITLE PGM4_3: CASE CONVERSION PROGRAM

.MODEL SMALL

.STACK 100H

.DATA
MSG1 DB 'ENTER A LOWER CASE LETTER: $'
MSG2 DB 0DH,0AH,'IN UPPER CASE IT IS: '
CHAR DB ?,'$'

.CODE

MAIN PROC

;initialize DS
MOV AX,@DATA ;get data segment
MOV DS,AX ;initialize DS

;print user prompt
LEA DX,MSG1 ;get first message
MOV AH,9 ;display string function
INT 21H ;display first message

;input a character and convert to upper case
MOV AH,1 ;read character function
INT 21H ;read a small letter into AL
SUB AL,20H ;convert it to upper case
MOV CHAR,AL ;and store it

;display on the next line
LEA DX,MSG2 ;get second message MSG2
MOV AH,9 ;display string function
INT 21H ;display message and upper case

;DOS exit
MOV AH,4CH
INT 21H ;DOS exit

MAIN ENDP

END MAIN
```

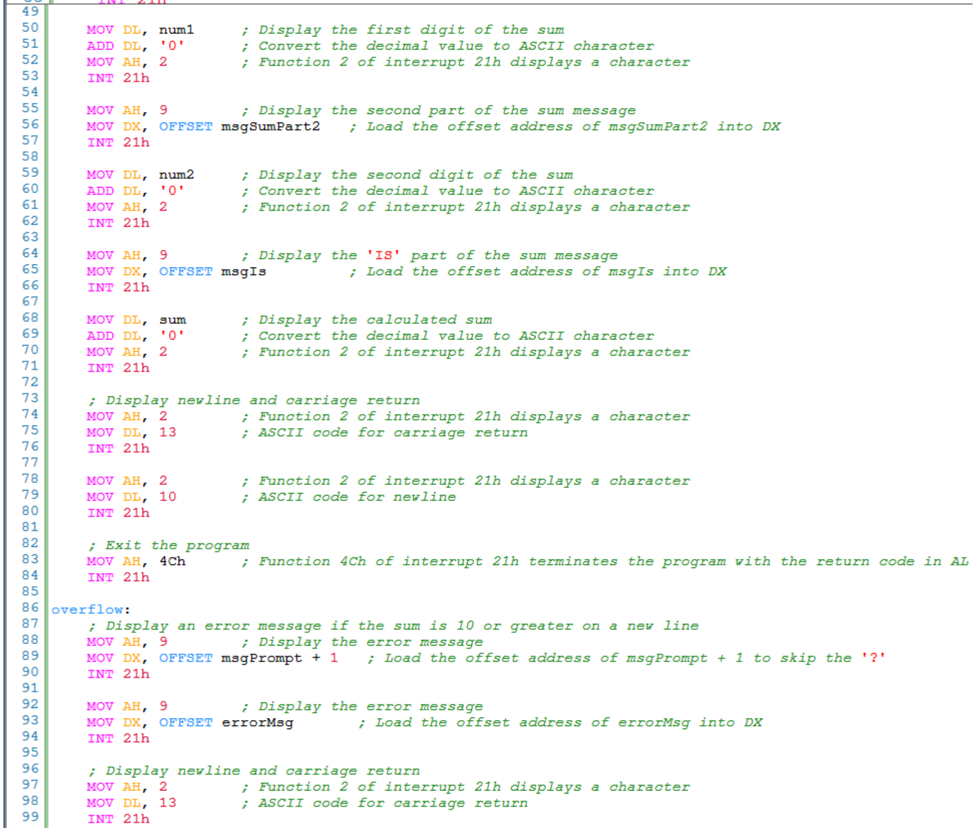
```
ENTER A LOWER CASE LETTER: a
IN UPPER CASE IT IS: A

Press any key to exit...
```

Figure 3 shows the source code for PGM4_3 and the terminal output after execution.

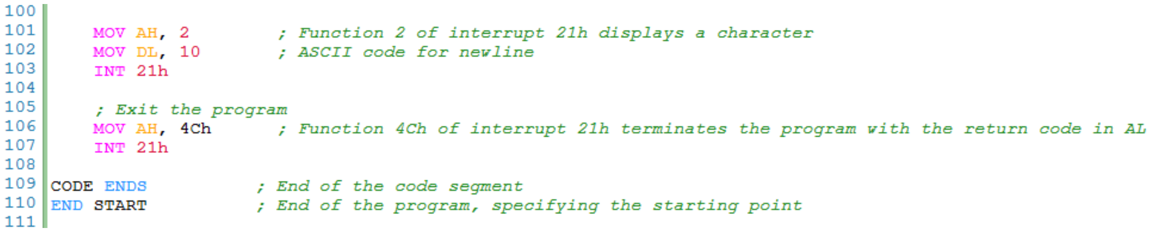
Before execution:



```
DATA SEGMENT
    msgPrompt DB '?', '$'               ; Prompt message to display '?'
    msgSumPart1 DB 13, 10, 'THE SUM OF ', '$'     ; First part of the message to display sum
    msgSumPart2 DB ' AND ', '$'         ; Second part of the message to display sum
    msgIs DB ' IS ', '$'               ; Message to display ' IS '
    errorMsg DB 13, 10, 'Sum is not less than 10.', 13, 10, '$'   ; Error message to display

    num1 DB ?    ; Reserve space for first digit
    num2 DB ?    ; Reserve space for second digit
    sum DB ?     ; Reserve space for sum
DATA ENDS

CODE SEGMENT
ASSUME CS:CODE, DS:DATA    ; Set up code and data segment assumptions

START:    ; Entry point for the program
    MOV AX, DATA    ; Load the data segment address into AX
    MOV DS, AX      ; Set the data segment register to point to the data segment

    ; Display prompt message
    MOV AH, 9        ; Function 9 of interrupt 21h displays a '$'-terminated string
    MOV DX, OFFSET msgPrompt   ; Load the offset address of msgPrompt into DX
    INT 21h          ; Call interrupt 21h to display the prompt message

    ; Read two decimal digits from the user
    MOV AH, 1        ; Function 1 of interrupt 21h reads a character from standard input
    INT 21h          ; Call interrupt 21h to read the first decimal digit
    SUB AL, '0'      ; Convert the ASCII character to its decimal value
    MOV num1, AL     ; Store the first digit in the variable num1

    MOV AH, 1        ; Read the second decimal digit
    INT 21h          ; Call interrupt 21h to read the second decimal digit
    SUB AL, '0'      ; Convert the ASCII character to its decimal value
    MOV num2, AL     ; Store the second digit in the variable num2

    ; Calculate the sum of the digits
    MOV AL, num1     ; Move the first digit into AL
    ADD AL, num2     ; Add the second digit to AL
    MOV sum, AL      ; Store the sum in the variable sum

    ; Check if the sum is less than 10
    CMP sum, 10      ; Compare the sum with 10
    JAE overflow     ; If sum is greater than or equal to 10, jump to the overflow label

    ; Display the sum message on a new line
    MOV AH, 9        ; Display the first part of the sum message
    MOV DX, OFFSET msgSumPart1   ; Load the offset address of msgSumPart1 into DX
    INT 21h

    MOV DL, num1     ; Display the first digit of the sum
    ADD DL, '0'      ; Convert the decimal value to ASCII character
    MOV AH, 2        ; Function 2 of interrupt 21h displays a character
    INT 21h
```



```
    MOV DL, num1     ; Display the first digit of the sum
    ADD DL, '0'      ; Convert the decimal value to ASCII character
    MOV AH, 2        ; Function 2 of interrupt 21h displays a character
    INT 21h

    MOV AH, 9        ; Display the second part of the sum message
    MOV DX, OFFSET msgSumPart2   ; Load the offset address of msgSumPart2 into DX
    INT 21h

    MOV DL, num2     ; Display the second digit of the sum
    ADD DL, '0'      ; Convert the decimal value to ASCII character
    MOV AH, 2        ; Function 2 of interrupt 21h displays a character
    INT 21h

    MOV AH, 9        ; Display the 'IS' part of the sum message
    MOV DX, OFFSET msgIs         ; Load the offset address of msgIs into DX
    INT 21h

    MOV DL, sum      ; Display the calculated sum
    ADD DL, '0'      ; Convert the decimal value to ASCII character
    MOV AH, 2        ; Function 2 of interrupt 21h displays a character
    INT 21h

    ; Display newline and carriage return
    MOV AH, 2        ; Function 2 of interrupt 21h displays a character
    MOV DL, 13       ; ASCII code for carriage return
    INT 21h

    MOV AH, 2        ; Function 2 of interrupt 21h displays a character
    MOV DL, 10       ; ASCII code for newline
    INT 21h

    ; Exit the program
    MOV AH, 4Ch      ; Function 4Ch of interrupt 21h terminates the program with the return code in AL
    INT 21h

overflow:
    ; Display an error message if the sum is 10 or greater on a new line
    MOV AH, 9        ; Display the error message
    MOV DX, OFFSET msgPrompt + 1   ; Load the offset address of msgPrompt + 1 to skip the '?'
    INT 21h

    MOV AH, 9        ; Display the error message
    MOV DX, OFFSET errorMsg       ; Load the offset address of errorMsg into DX
    INT 21h

    ; Display newline and carriage return
    MOV AH, 2        ; Function 2 of interrupt 21h displays a character
    MOV DL, 13       ; ASCII code for carriage return
    INT 21h
```

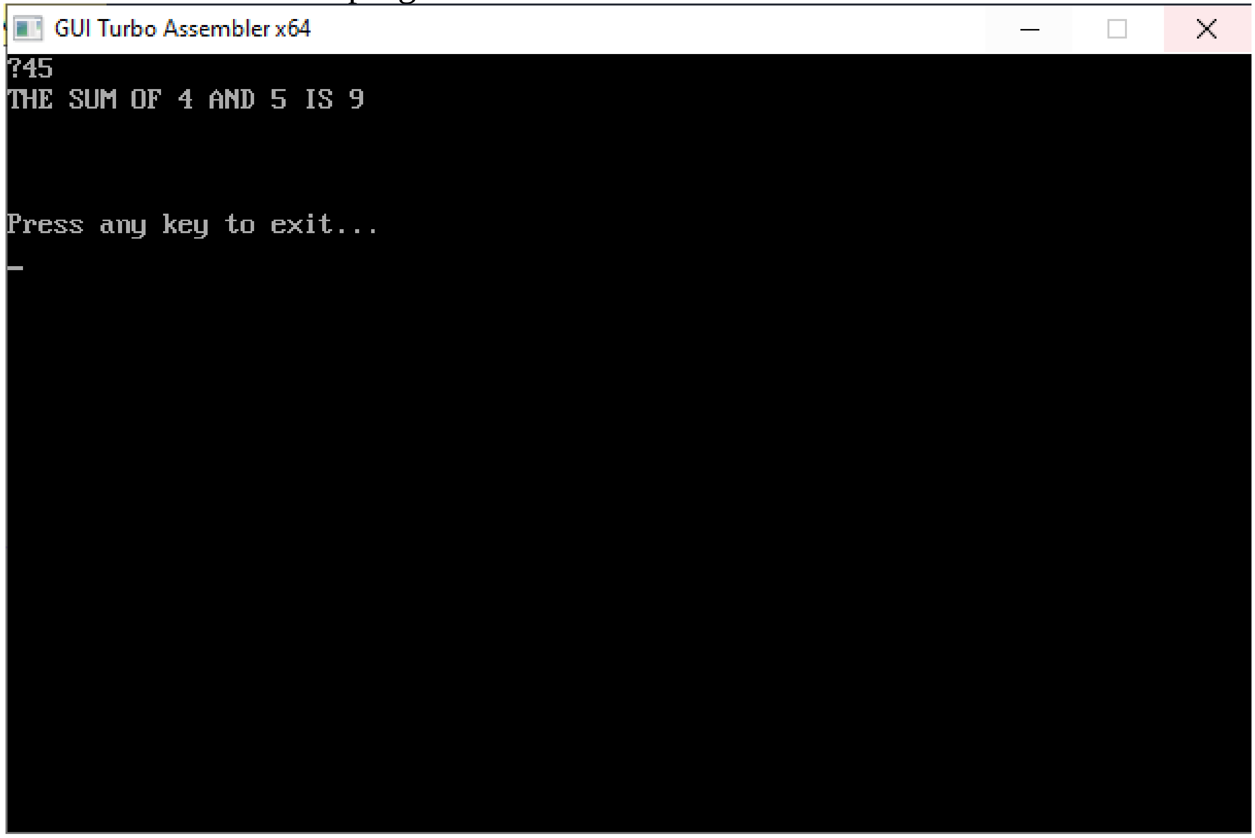


```
    MOV AH, 2           ; Function 2 of interrupt 21h displays a character
    MOV DL, 10          ; ASCII code for newline
    INT 21h

    ; Exit the program
    MOV AH, 4Ch         ; Function 4Ch of interrupt 21h terminates the program with the return code in AL
    INT 21h

CODE ENDS             ; End of the code segment
END START             ; End of the program, specifying the starting point
```

Figure 4, 5, 6 shows the source code for DISPLAY.ASM.

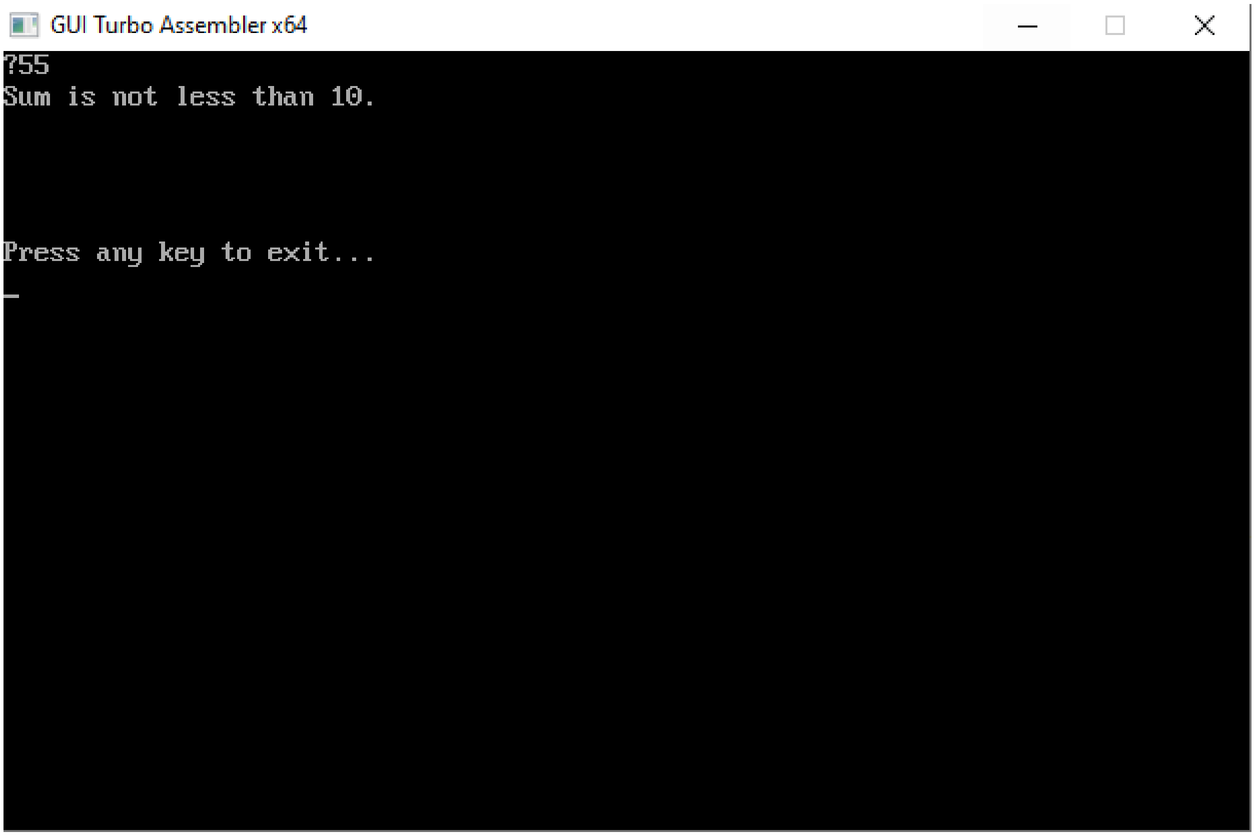
Results

After Execution of the program



```
?45
THE SUM OF 4 AND 5 IS 9



Press any key to exit...
_
```

Figure 7 shows the terminal output for DISPLAY.ASM if the two entered integer’s sum is less than 10.



```
?55
Sum is not less than 10.



Press any key to exit...
_
```

Figure 8 shows the terminal output for DISPLAY.ASM if the two entered integer’s sum is 10 or more.