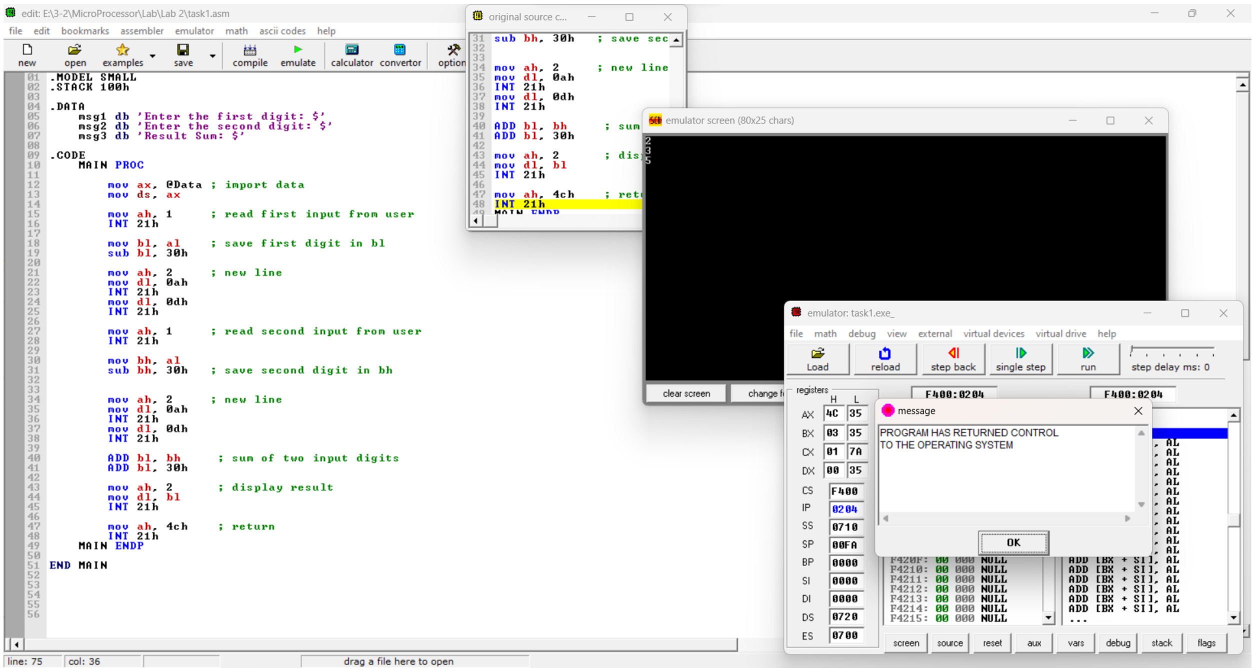
Task 1

Write an Assembly Language Program (ALP) to read two decimal digits both of which are less than 5 and display their summation in the next line.



edit: E:\3-2\MicroProcessor\Lab\Lab 2\task1.asm

```
.MODEL SMALL
.STACK 100h

.DATA
    msg1 db 'Enter the first digit: $'
    msg2 db 'Enter the second digit: $'
    msg3 db 'Result Sum: $'

.CODE
    MAIN PROC

        mov ax, @Data ; import data
        mov ds, ax

        mov ah, 1      ; read first input from user
        INT 21h

        mov bl, al     ; save first digit in bl
        sub bl, 30h

        mov ah, 2      ; new line
        mov dl, 0ah
        INT 21h
        mov dl, 0dh
        INT 21h

        mov ah, 1      ; read second input from user
        INT 21h

        mov bh, al
        sub bh, 30h    ; save second digit in bh


        mov ah, 2      ; new line
        mov dl, 0ah
        INT 21h
        mov dl, 0dh
        INT 21h

        ADD bl, bh      ; sum of two input digits
        ADD bl, 30h

        mov ah, 2       ; display result
        mov dl, bl
        INT 21h

        mov ah, 4ch     ; return
        INT 21h
    MAIN ENDP

END MAIN
```

line: 75 col: 36

emulator screen (80x25 chars)

```
2
3
5
```

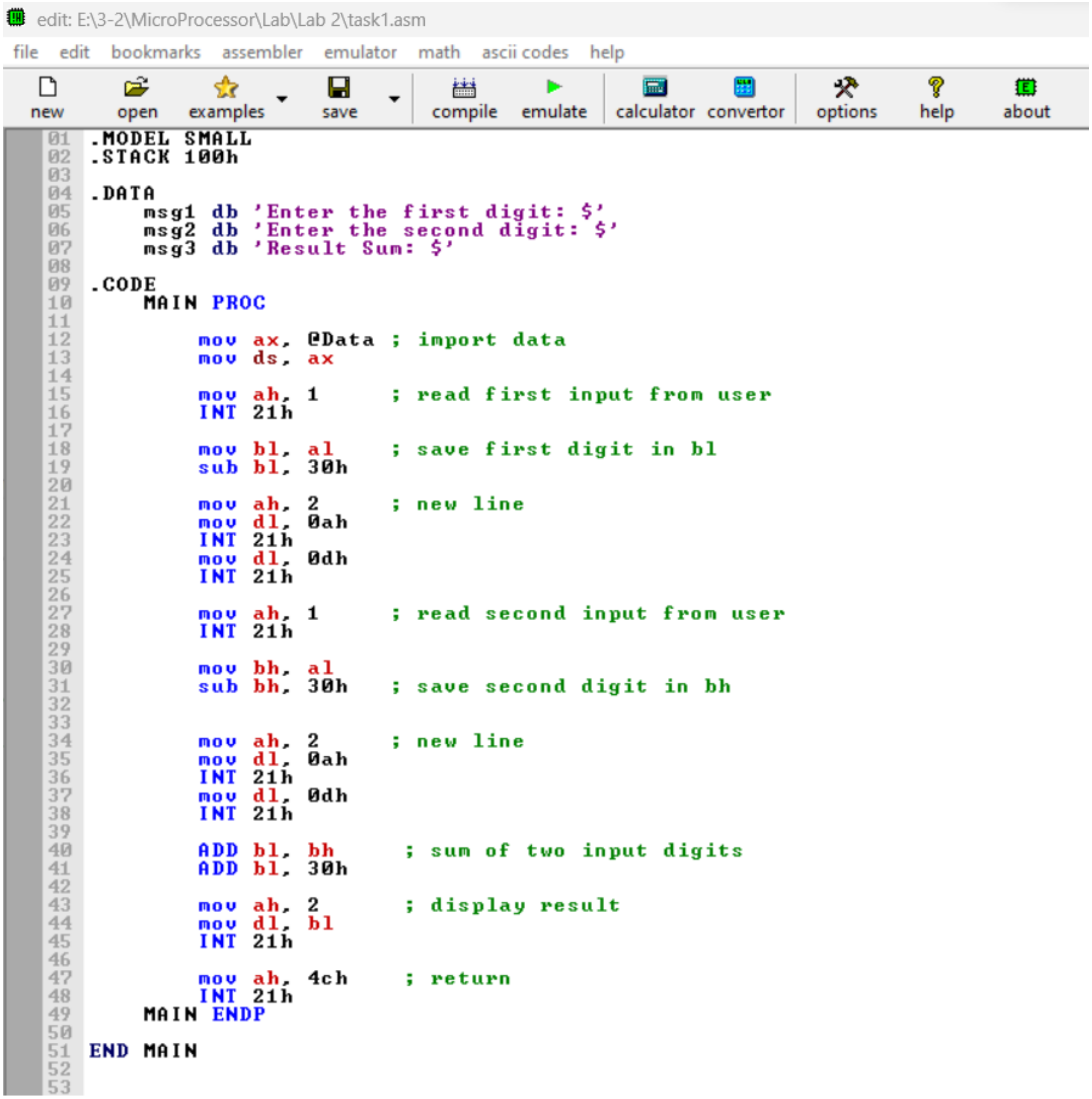
emulator: task1.exe_

message

PROGRAM HAS RETURNED CONTROL
TO THE OPERATING SYSTEM

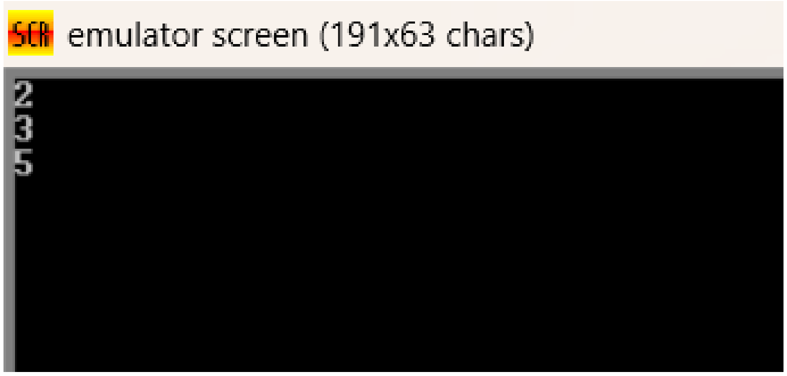
OK

file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor option
drag a file here to open
clear screen
Load reload step back single step run step delay ms: 0
registers
screen source reset aux vars debug stack flags


edit: E:\3-2\MicroProcessor\Lab\Lab 2\task1.asm
file edit bookmarks assembler emulator math ascii codes help
new
open
examples
save
compile
emulate
calculator
convertor
options
help
about

```
.MODEL SMALL
.STACK 100h

.DATA
    msg1 db 'Enter the first digit: $'
    msg2 db 'Enter the second digit: $'
    msg3 db 'Result Sum: $'

.CODE
    MAIN PROC

        mov ax, @Data ; import data
        mov ds, ax

        mov ah, 1     ; read first input from user
        INT 21h

        mov bl, al    ; save first digit in bl
        sub bl, 30h

        mov ah, 2     ; new line
        mov dl, 0ah
        INT 21h
        mov dl, 0dh
        INT 21h

        mov ah, 1     ; read second input from user
        INT 21h

        mov bh, al
        sub bh, 30h   ; save second digit in bh


        mov ah, 2     ; new line
        mov dl, 0ah
        INT 21h
        mov dl, 0dh
        INT 21h

        ADD bl, bh     ; sum of two input digits
        ADD bl, 30h

        mov ah, 2      ; display result
        mov dl, bl
        INT 21h

        mov ah, 4ch    ; return
        INT 21h
    MAIN ENDP

END MAIN
```

Taking first input from the user and saving it in the bl register.
Then a new line is given.
Taking second input from user and saving it in the bh register.
Then a new line is given.
Adding the two input digits and saving in bl.
Displaying the result in bl register.
Return and exit.

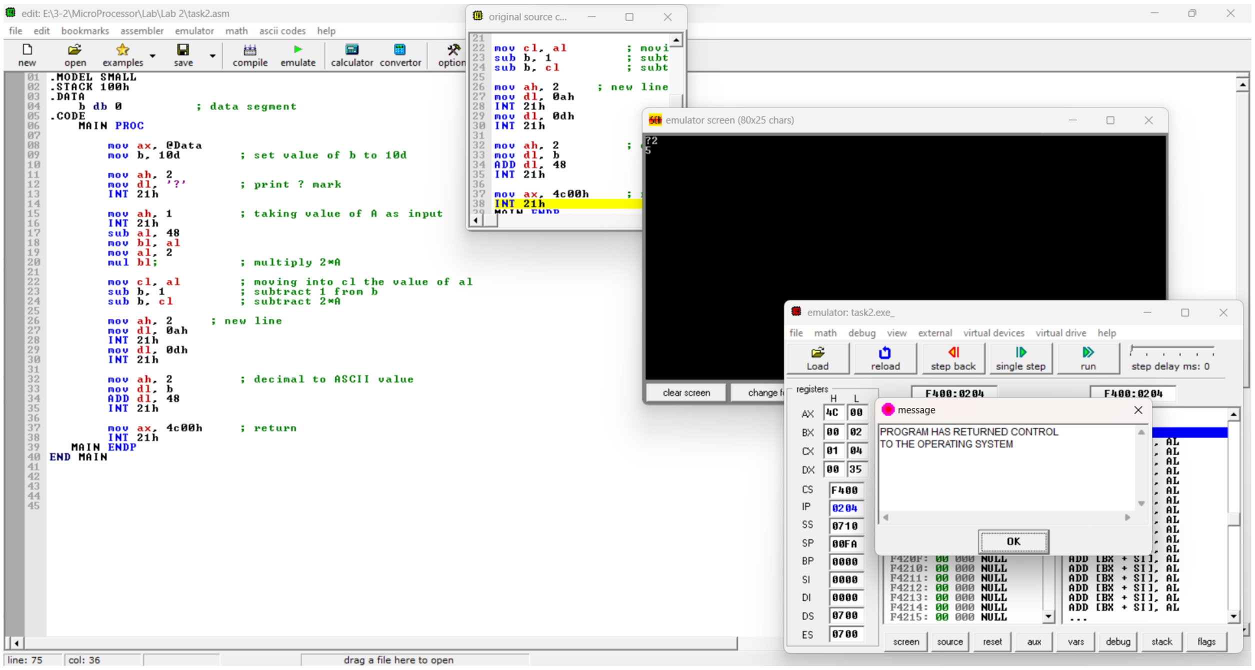


```
2
3
5
```

First input is 2
Second input is 3
Sum is 5.

Task 2

Write an Assembly Language Program (ALP) to translate the following high level language assignment statements into assembly language.
A= B - 2*A +1
Where the A, and B are byte variables. Your program does the following,
i. Keep the value of variable A as undefined and set the value of variable B to 10D
ii. Display ‘?’
iii. Read the value of variable A in new line
iv. Display the updated value of A= B - 2*A -1 in new line


edit: E:\3-2\MicroProcessor\Lab\Lab 2\task2.asm
file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor option
original source c...
emulator screen (80x25 chars)
clear screen change f
emulator: task2.exe_
file math debug view external virtual devices virtual drive help
Load reload step back single step run step delay ms: 0
registers
screen source reset aux vars debug stack flags
line: 75 col: 36 drag a file here to open

```
.MODEL SMALL
.STACK 100h
.DATA
    b db 0            ; data segment
.CODE
    MAIN PROC

        mov ax, @Data
        mov b, 10d          ; set value of b to 10d

        mov ah, 2
        mov dl, '?'         ; print ? mark
        INT 21h

        mov ah, 1           ; taking value of A as input
        INT 21h
        sub al, 48
        mov bl, al
        mov al, 2
        mul bl;             ; multiply 2*A

        mov cl, al          ; moving into cl the value of al
        sub b, 1            ; subtract 1 from b
        sub b, cl           ; subtract 2*A

        mov ah, 2     ; new line
        mov dl, 0ah
        INT 21h
        mov dl, 0dh
        INT 21h

        mov ah, 2           ; decimal to ASCII value
        mov dl, b
        ADD dl, 48
        INT 21h

        mov ax, 4c00h       ; return
        INT 21h
   MAIN ENDP
END MAIN
```

```
?2
5
```

```
     H  L
AX   4C 00
BX   00 02
CX   01 04
DX   00 35
CS   F400
IP   0204
SS   0710
SP   00FA
BP   0000
SI   0000
DI   0000
DS   0700
ES   0700

F400:0204          F400:0204

F420F: 00 000 NULL      ADD [BX + SI], AL
F4210: 00 000 NULL      ADD [BX + SI], AL
F4211: 00 000 NULL      ADD [BX + SI], AL
F4212: 00 000 NULL      ADD [BX + SI], AL
F4213: 00 000 NULL      ADD [BX + SI], AL
F4214: 00 000 NULL      ADD [BX + SI], AL
F4215: 00 000 NULL      ...
```

message

PROGRAM HAS RETURNED CONTROL
TO THE OPERATING SYSTEM

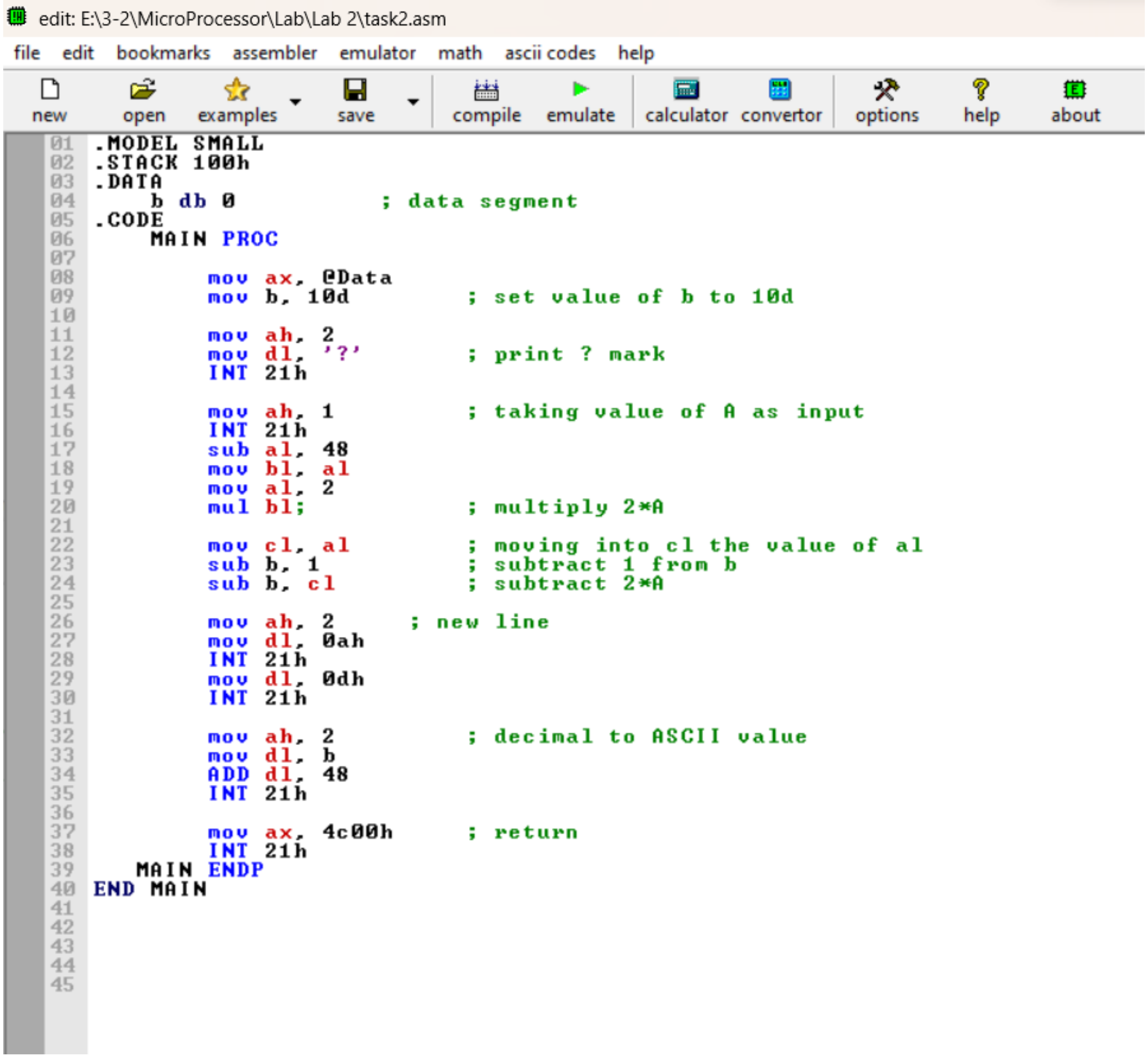
OK

Setting value of b to 10.

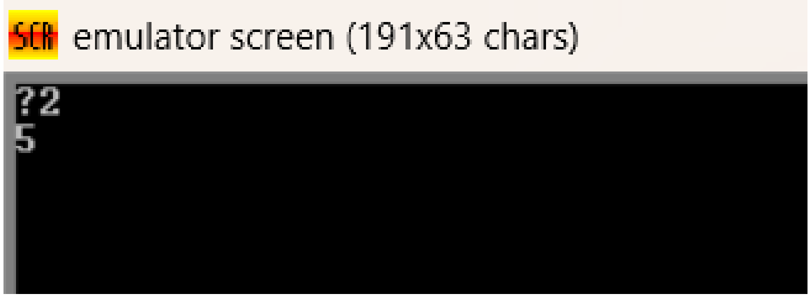
Display ? mark

Taking value of A as input

Multiplying 2 with A and performing the calculation according to the equation



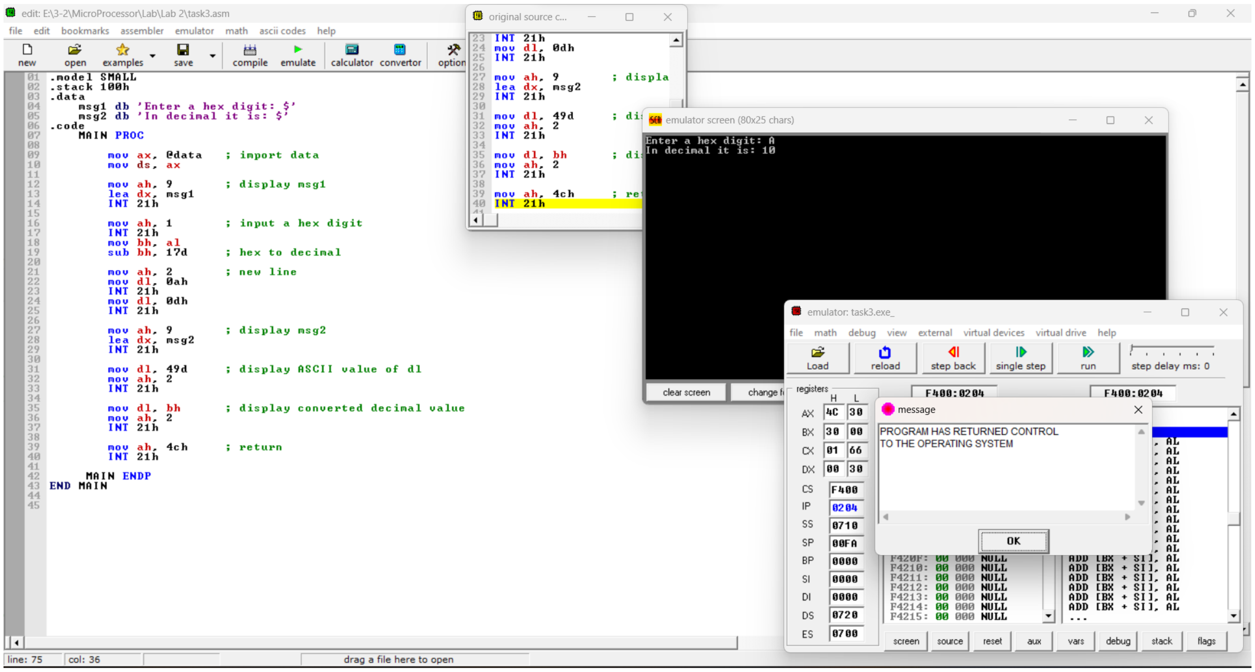
```
.MODEL SMALL
.STACK 100h
.DATA
    b db 0            ; data segment
.CODE
    MAIN PROC

        mov ax, @Data
        mov b, 10d          ; set value of b to 10d

        mov ah, 2
        mov dl, '?'         ; print ? mark
        INT 21h

        mov ah, 1           ; taking value of A as input
        INT 21h
        sub al, 48
        mov bl, al
        mov al, 2
        mul bl;             ; multiply 2*A

        mov cl, al          ; moving into cl the value of al
        sub b, 1            ; subtract 1 from b
        sub b, cl           ; subtract 2*A

        mov ah, 2     ; new line
        mov dl, 0ah
        INT 21h
        mov dl, 0dh
        INT 21h

        mov ah, 2           ; decimal to ASCII value
        mov dl, b
        ADD dl, 48
        INT 21h

        mov ax, 4c00h       ; return
        INT 21h
   MAIN ENDP
END MAIN
```



```
?2
5
```

Putting the value of A as 2.

Task 3


edit: E:\3-2\MicroProcessor\Lab\Lab 2\task3.asm

```
.model SMALL
.stack 100h
.data
    msg1 db 'Enter a hex digit: $'
    msg2 db 'In decimal it is: $'
.code
    MAIN PROC

        mov ax, @data    ; import data
        mov ds, ax

        mov ah, 9        ; display msg1
        lea dx, msg1
        INT 21h

        mov ah, 1        ; input a hex digit
        INT 21h
        mov bh, al
        sub bh, 17d      ; hex to decimal

        mov ah, 2        ; new line
        mov dl, 0ah
        INT 21h
        mov dl, 0dh
        INT 21h

        mov ah, 9        ; display msg2
        lea dx, msg2
        INT 21h

        mov dl, 49d      ; display ASCII value of dl
        mov ah, 2
        INT 21h

        mov dl, bh       ; display converted decimal value
        mov ah, 2
        INT 21h

        mov ah, 4ch      ; return
        INT 21h

     MAIN ENDP
END MAIN
```

emulator screen (80x25 chars)

```
Enter a hex digit: A
In decimal it is: 10
```

emulator: task3.exe_

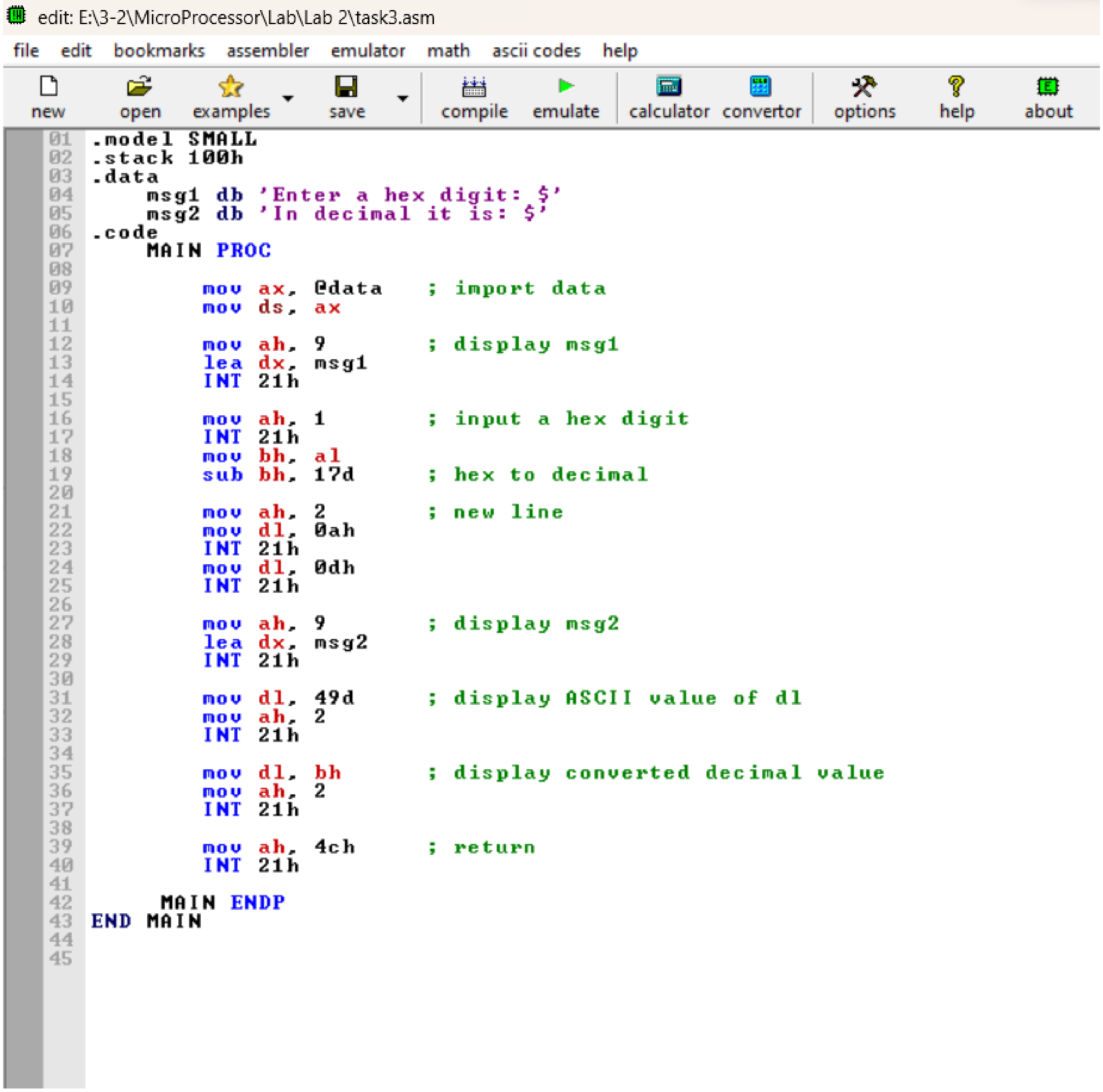
message

PROGRAM HAS RETURNED CONTROL
TO THE OPERATING SYSTEM

```
registers
     H   L
AX  4C  30
BX  30  00
CX  01  66
DX  00  30
CS  F400
IP  0204
SS  0710
SP  00FA
BP  0000
SI  0000
DI  0000
DS  0720
ES  0700
```

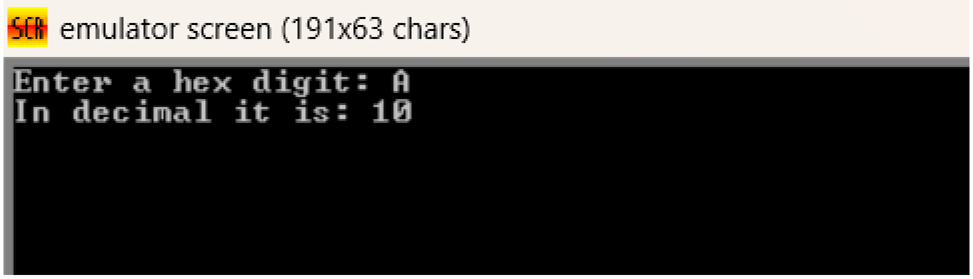
```
F400:0204                 F400:0204
F420F: 00 000 NULL        ADD [BX + SI], AL
F4210: 00 000 NULL        ADD [BX + SI], AL
F4211: 00 000 NULL        ADD [BX + SI], AL
F4212: 00 000 NULL        ADD [BX + SI], AL
F4213: 00 000 NULL        ADD [BX + SI], AL
F4214: 00 000 NULL        ADD [BX + SI], AL
F4215: 00 000 NULL        ...
```

file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor option
original source c...
clear screen change f
file math debug view external virtual devices virtual drive help
Load reload step back single step run step delay ms: 0
OK
screen source reset aux vars debug stack flags
line: 75 col: 36 drag a file here to open



```
.model SMALL
.stack 100h
.data
    msg1 db 'Enter a hex digit: $'
    msg2 db 'In decimal it is: $'
.code
    MAIN PROC

        mov ax, @data    ; import data
        mov ds, ax

        mov ah, 9        ; display msg1
        lea dx, msg1
        INT 21h

        mov ah, 1        ; input a hex digit
        INT 21h
        mov bh, al
        sub bh, 17d      ; hex to decimal

        mov ah, 2        ; new line
        mov dl, 0ah
        INT 21h
        mov dl, 0dh
        INT 21h

        mov ah, 9        ; display msg2
        lea dx, msg2
        INT 21h

        mov dl, 49d      ; display ASCII value of dl
        mov ah, 2
        INT 21h

        mov dl, bh       ; display converted decimal value
        mov ah, 2
        INT 21h

        mov ah, 4ch      ; return
        INT 21h

     MAIN ENDP
END MAIN
```

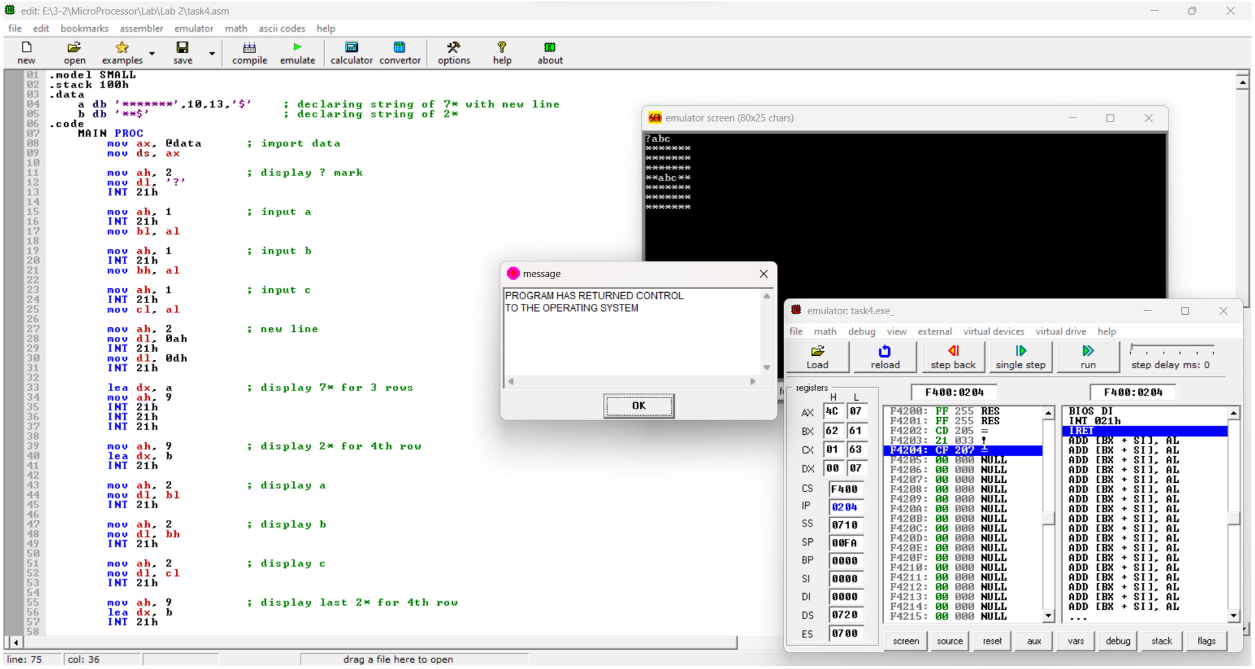
Inputting a hex digit and converting to decimal.



```
Enter a hex digit: A
In decimal it is: 10
```

Input hex digit is A here.

Task 4



edit: E:\3-2\MicroProcessor\Lab\Lab 2\task4.asm
file edit bookmarks assembler emulator math ascii codes help
new open examples save compile emulate calculator convertor options help about

```
.model SMALL
.stack 100h
.data
    a db '*******',10,13,'$'     ; declaring string of 7* with new line
    b db '**$'                   ; declaring string of 2*
.code
    MAIN PROC
        mov ax, @data       ; import data
        mov ds, ax

        mov ah, 2           ; display ? mark
        mov dl, '?'
        INT 21h

        mov ah, 1           ; input a
        INT 21h
        mov bl, al

        mov ah, 1           ; input b
        INT 21h
        mov bh, al

        mov ah, 1           ; input c
        INT 21h
        mov cl, al

        mov ah, 2           ; new line
        mov dl, 0ah
        INT 21h
        mov dl, 0dh
        INT 21h

        lea dx, a           ; display 7* for 3 rows
        mov ah, 9
        INT 21h
        INT 21h
        INT 21h

        mov ah, 9           ; display 2* for 4th row
        lea dx, b
        INT 21h

        mov ah, 2           ; display a
        mov dl, bl
        INT 21h

        mov ah, 2           ; display b
        mov dl, bh
        INT 21h

        mov ah, 2           ; display c
        mov dl, cl
        INT 21h

        mov ah, 9           ; display last 2* for 4th row
        lea dx, b
        INT 21h

```

line: 75 col: 36 drag a file here to open
emulator screen (80x25 chars)

```
?abc
*******
*******
*******
**abc**
*******
*******
*******
```

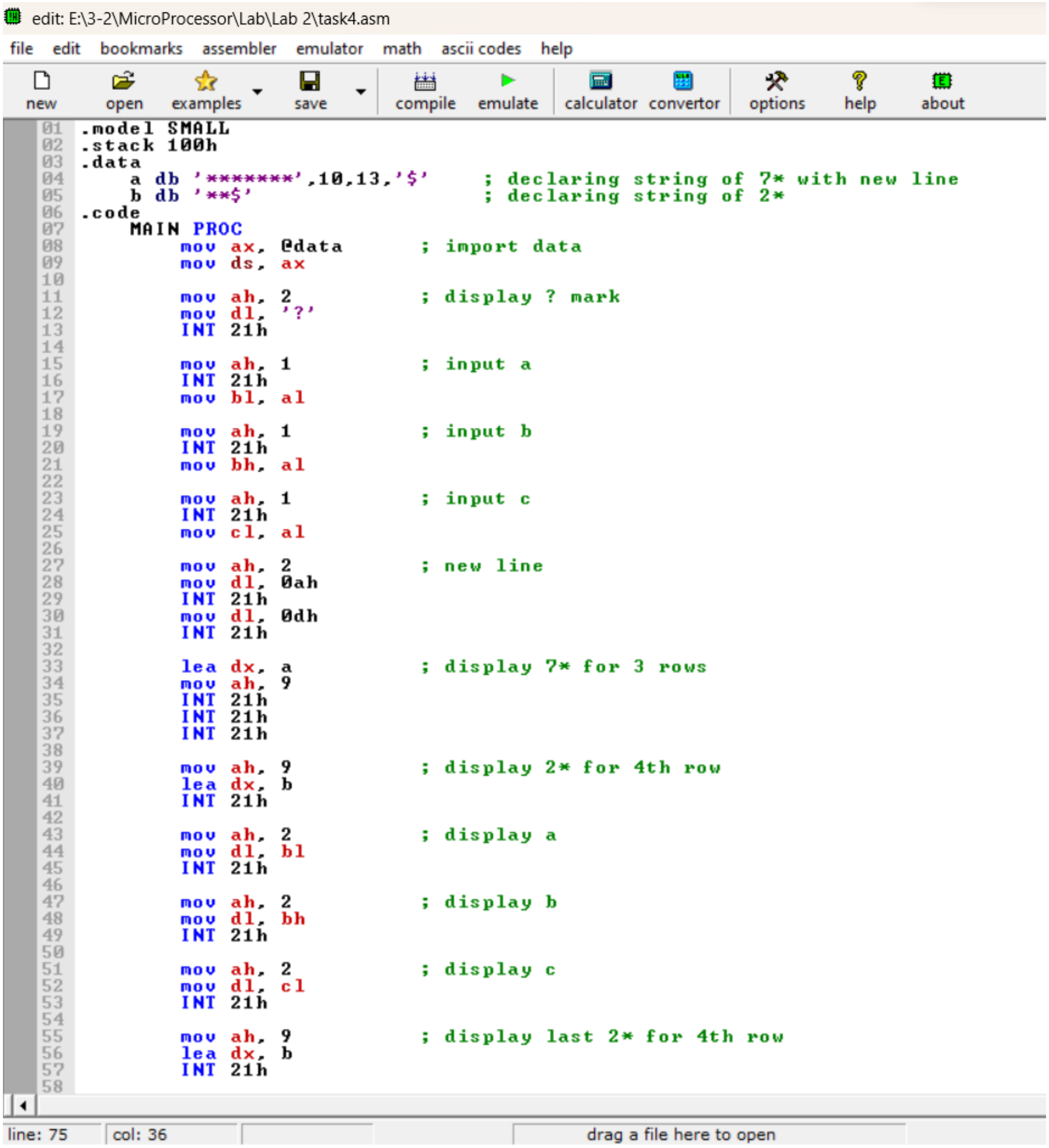
PROGRAM HAS RETURNED CONTROL
TO THE OPERATING SYSTEM

OK
emulator: task4.exe_
file math debug view external virtual devices virtual drive help
Load reload step back single step run step delay ms: 0
registers

	H	L
AX	4C	07
BX	62	61
CX	01	63
DX	00	07
CS	F400	
IP	0204	
SS	0710	
SP	00FA	
BP	0000	
SI	0000	
DI	0000	
DS	0720	
ES	0700	

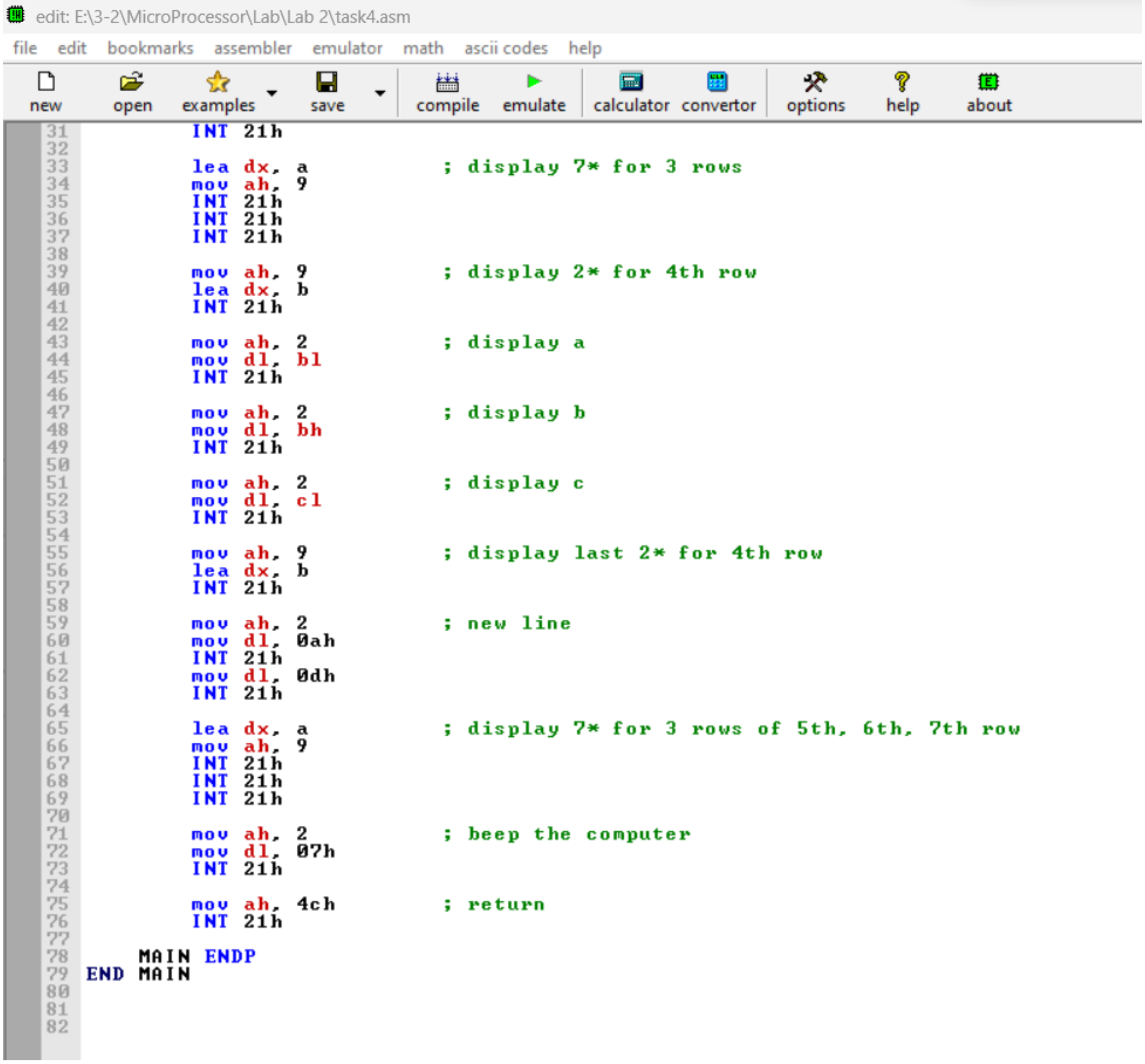
```
F400:0204
F4200: FF 255 RES
F4201: FF 255 RES
F4202: CD 205 =
F4203: 21 033 !
F4204: CF 207 ±
F4205: 00 000 NULL
F4206: 00 000 NULL
F4207: 00 000 NULL
F4208: 00 000 NULL
F4209: 00 000 NULL
F420A: 00 000 NULL
F420B: 00 000 NULL
F420C: 00 000 NULL
F420D: 00 000 NULL
F420E: 00 000 NULL
F420F: 00 000 NULL
F4210: 00 000 NULL
F4211: 00 000 NULL
F4212: 00 000 NULL
F4213: 00 000 NULL
F4214: 00 000 NULL
F4215: 00 000 NULL
```

```
F400:0204
BIOS DI
INT 021h
IRET
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
ADD [BX + SI], AL
...
```

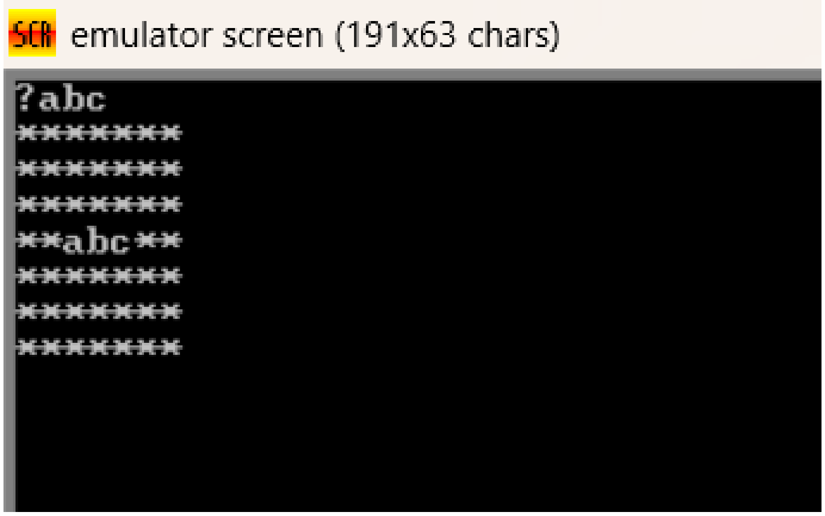


```
.model SMALL
.stack 100h
.data
    a db '*******',10,13,'$'     ; declaring string of 7* with new line
    b db '**$'                   ; declaring string of 2*
.code
    MAIN PROC
        mov ax, @data        ; import data
        mov ds, ax

        mov ah, 2            ; display ? mark
        mov dl, '?'
        INT 21h

        mov ah, 1            ; input a
        INT 21h
        mov bl, al

        mov ah, 1            ; input b
        INT 21h
        mov bh, al

        mov ah, 1            ; input c
        INT 21h
        mov cl, al

        mov ah, 2            ; new line
        mov dl, 0ah
        INT 21h
        mov dl, 0dh
        INT 21h

        lea dx, a            ; display 7* for 3 rows
        mov ah, 9
        INT 21h
        INT 21h
        INT 21h

        mov ah, 9            ; display 2* for 4th row
        lea dx, b
        INT 21h

        mov ah, 2            ; display a
        mov dl, bl
        INT 21h

        mov ah, 2            ; display b
        mov dl, bh
        INT 21h

        mov ah, 2            ; display c
        mov dl, cl
        INT 21h

        mov ah, 9            ; display last 2* for 4th row
        lea dx, b
        INT 21h

```

Input abc and the *formation are of 2 types:
7* and 2*.



```
        INT 21h

        lea dx, a          ; display 7* for 3 rows
        mov ah, 9
        INT 21h
        INT 21h
        INT 21h

        mov ah, 9          ; display 2* for 4th row
        lea dx, b
        INT 21h

        mov ah, 2          ; display a
        mov dl, bl
        INT 21h

        mov ah, 2          ; display b
        mov dl, bh
        INT 21h

        mov ah, 2          ; display c
        mov dl, cl
        INT 21h

        mov ah, 9          ; display last 2* for 4th row
        lea dx, b
        INT 21h

        mov ah, 2          ; new line
        mov dl, 0ah
        INT 21h
        mov dl, 0dh
        INT 21h

        lea dx, a          ; display 7* for 3 rows of 5th, 6th, 7th row
        mov ah, 9
        INT 21h
        INT 21h
        INT 21h

        mov ah, 2          ; beep the computer
        mov dl, 07h
        INT 21h

        mov ah, 4ch        ; return
        INT 21h

    MAIN ENDP
END MAIN
```



```
?abc
*******
*******
*******
**abc**
*******
*******
*******
```